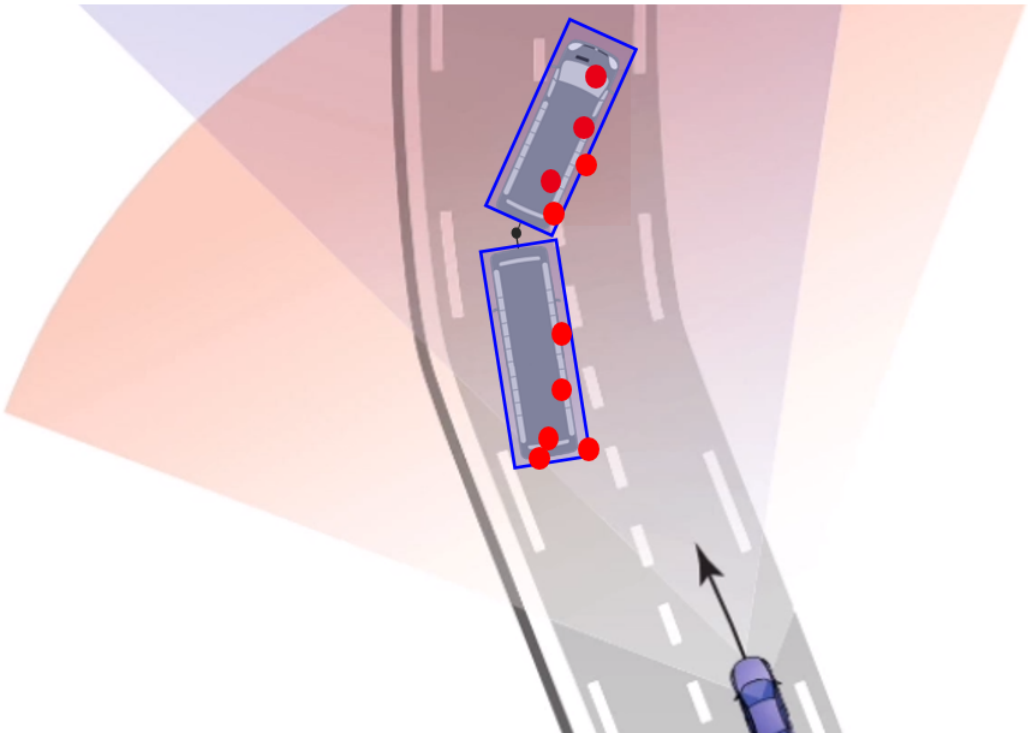


Tracking of articulated vehicles using radar measurements

Master's thesis in Systems, Controls and Mechatronics

FELIX GUSTAVSSON
LUDVIG HAZARD

DEPARTMENT OF ELECTRICAL ENGINEERING

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2021
www.chalmers.se

Master's thesis 2021

Tracking of articulated vehicles using radar measurements

FELIX GUSTAVSSON
LUDVIG HAZARD

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Signal Processing and Biomedical Engineering
Chalmers University of Technology
Gothenburg, Sweden 2021

Tracking of articulated vehicles using radar measurements
FELIX GUSTAVSSON
LUDVIG HAZARD

Supervisor: Elvira Ramle, Aptiv
Examiner: Professor Tomas McKelvey, Department of Electrical Engineering

Master's Thesis 2021
Department of Electrical Engineering
Division of Signal Processing and Biomedical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2021

Tracking of articulated vehiclesusing radar measurements
FELIX GUSTAVSSON
LUDVIG HAZARD
Department of Electrical Engineering
Chalmers University of Technology

Abstract

In this thesis, tracking of an articulated vehicle using radar measurements is investigated. The radar sensors are mounted on an observing host vehicle and the purpose is to estimate the position and motion of a distant articulated vehicle.

The problem was studied in a simulated environment using synthetic radar detections. The articulated vehicle, consisting of a trailer and its towing tractor unit, was simulated using a modified kinematic bicycle model. A tracking algorithm was developed which uses k-means clustering to partition the radar detections to each vehicle unit. The state of each unit was then estimated separately and then merged to give an estimate of the vehicle as a whole, using an Extended Kalman Filter.

The information that can be obtained from the detections depends on the aspect angle of the vehicle. Therefore, different measurement models were developed to handle these situations accordingly.

The developed tracking algorithm shows that it can handle certain scenarios well, while other scenarios are more challenging. The results show that further developments are needed to obtain a robust solution.

Radar, Articulated vehicles, Extended Target Tracking, Vehicle state estimation

Acknowledgements

This thesis was written during the COVID-19 pandemic from the comfort of Ludvig's apartment. Consequently, all supervision was given digitally.

First and foremost, we wish to thank our supervisors Elvira Ramle, Niclas Carlström, Hanna Nyqvist, Deepak Kiran and Jeanette Warnborg at APTIV for providing guidance, feedback and keeping us sane during the project.

We also wish to thank our supervisor and examiner Professor Tomas McKelvey for his support and helping us outline the project.

A special thank you to Mats Jonasson, who provide consultation regarding the dynamics of articulated vehicles.

Felix Gustavsson & Ludvig Hazard, Gothenburg, July 2021

Contents

List of Figures

List of Tables

1

Introduction

In many advanced technical systems there is a need to monitor objects and their behaviour over time. The information of interest depends on the context. For example, an airport control tower is interested in tracking nearby airplanes, or on a smaller scale, a biologist may be interested in tracking the properties of a cell in a microscope [1]. Another area of application is in the automotive industry, where many different vehicles and objects are in play.

To further the feature set and safety of modern vehicles, there is a need for the ability to construct a comprehensive and reliable perception of a vehicle's immediate surroundings. The knowledge is often used in Advanced Driver-Assistance Systems, which are electronic systems designed to aid the driver in different traffic scenarios. To achieve a reliable perception, many vehicle perception systems utilize, amongst other sensors, radars mounted on the vehicle itself to gather data about the vehicle's surroundings [2].

The use of radar to keep track of objects can be traced back to early applications within the air force, where radar systems were used as warning systems [3]. Radar signals are able to easily penetrate many adverse weather conditions such as rain and fog, making the sensor favorable over other types of sensors for relatively long distance sensing. Historically, radars have had a low sensor resolution such that relatively large objects would still only occupy a single resolution cell and in turn yield a single data point. This limitation only allowed tracking of objects modeled as a single point, as seen in Figure 1.1. However, recent advancements in sensor technology have greatly increased the capabilities of the radar and have thus made it possible for relatively small objects to return multiple data points thus giving rise to the concept of *extended object tracking* [4].

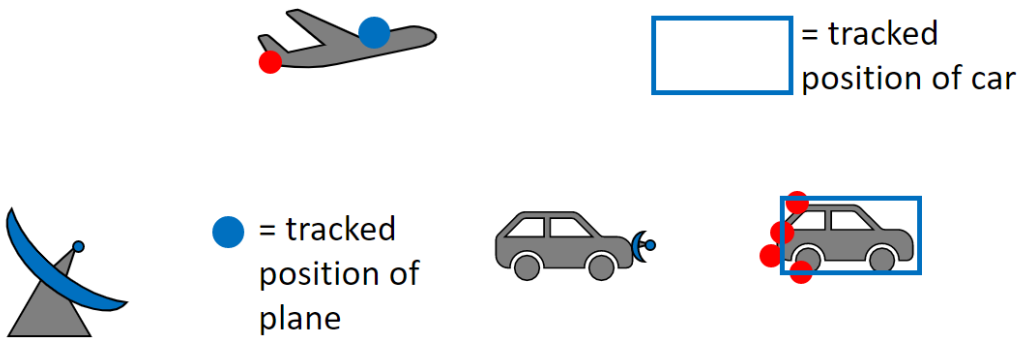


Figure 1.1: Difference between point and extended target tracking. To the left, an airplane occupies a single resolution cell and thus yields a single radar detection (red dot). To the right, a car occupies several resolution cells of the radar and thus yields several detections.

1.1 Extended object tracking

Whenever an object occupies multiple resolution cells of a sensor, the tracking problem is referred to as being of extended type [4]. As each occupied resolution cell can return a data point, an extended object tracking problem generally involves substantially more data which have to be processed in order to extract information about the object. The upside is that it can now be possible to estimate parameters of an object otherwise not possible, e.g. its shape, orientation, and kinematics. This information can help solve harder tracking problems, for example in urban traffic scenarios, where vehicles typically look and move very differently.

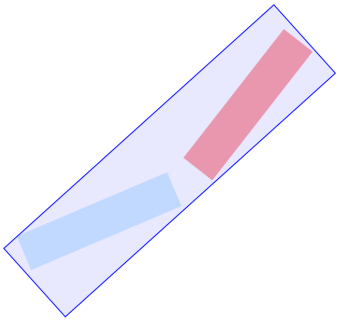
However, in the case with a radar sensor, the parameters are not directly obtainable from the radar measurements but rather have to be derived using various estimators. I.e. it is non-trivial how multiple data points should be used to estimate the movement of the vehicle. In [5], a method to obtain the orientation by fitting a box to the detections is proposed, but this method relies on a well visible contour of the vehicle. When the detections are more sparse, a probabilistic method is more suited as suggested in [6], where a radar response model uses the fact that typically radar measurements originate from certain reflection centers such as the wheelhouses or corners. These are a few of the current methods, however, what they all have in common is that they are specific for a vehicle type that is rigid in shape and currently do not work well when tracking vehicles that do not meet this criteria, e.g. articulated vehicles.

1.2 Articulated vehicle tracking

Articulated vehicles is an umbrella term covering any kind of vehicle that has one, or more, pivot joint(s) that allow the vehicle to turn more sharply than otherwise

possible. Some examples are trams, buses, semi-trailers or cars with trailers. Current state of the art radar tracking algorithms mainly model vehicles as a single, rigid shape (e.g. as a rectangle) and do not consider the kinematics of articulated vehicles. Anyone who has tried to reverse a car with a trailer intuitively understands that this is a great simplification and in a tracking algorithm, the current models can indeed result in a large disconnect between the estimated and actual state of an articulated vehicle as illustrated in Figure 1.2b.

(a) Articulated vehicle



(b) Tracking output

Figure 1.2: A tracking algorithm that models the vehicle in a) as a single rectangle can lead to an inaccurate estimation as shown in b).

Articulated vehicles are addressed in many problems related to control and path planning but currently there exists no public research on improving the tracking result that the authors of this thesis are aware of. In [7], radars are mounted on the articulated vehicle itself in order to estimate the angle to the trailer, but the information is only known to the driver of the articulated vehicle. The angle in a tractor-trailer combination is called the *articulation angle* and is a crucial variable when trying to estimate the motion of an articulated vehicle and is thus a central part of an articulated vehicle tracking algorithm.

1.3 Purpose

The purpose of this thesis is to develop a novel tracking algorithm that is tailored to track and estimate the movement of articulated vehicles. To do such, the aspiration of the algorithm is to incorporate a model of articulated vehicles such that its movement can be estimated at all times although parts of the vehicle may be obscured by itself. Hence the purpose of this thesis is split two-fold, to create a model for articulated vehicles to use in a tracking scenario, and to estimate its states using radar measurements.

1.4 Scope and limitations

The developed algorithm uses radar data from a 2D simulation environment and is a single object tracking algorithm, meaning only one vehicle will be tracked at a time.

The thesis also does not deal with the data association problem, i.e, every generated data point will be assumed to originate from the tracked vehicle. The simulated data resembles post-processed radar data and will not address the challenges with acquiring such data.

The tracked vehicle will be assumed to be identified as an articulated vehicle beforehand and only one pivot joint is allowed. The full dimensions of the vehicle are also considered known. No real-time implementations of the algorithm have been considered and it is limited to run in simulation only.

2

Theory

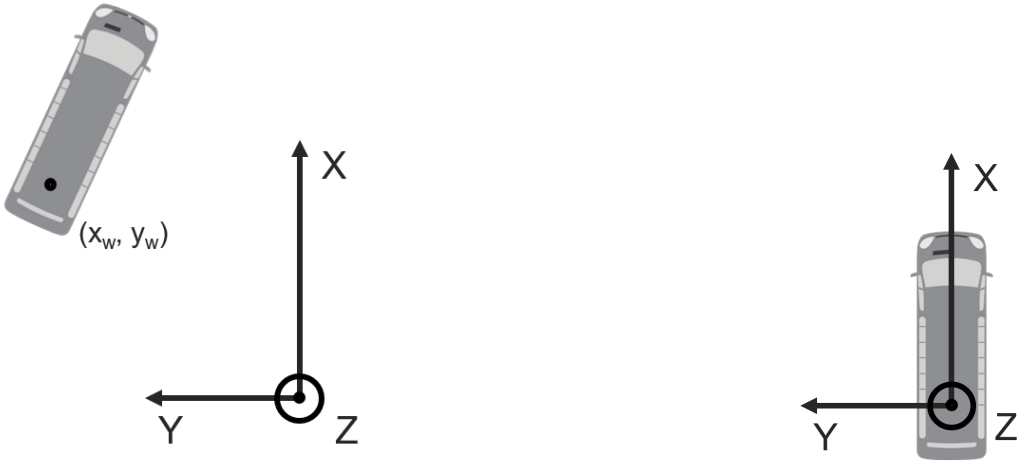
This section aims to present the nomenclature and material necessary to digest methods introduced in later chapters of the thesis.

2.1 Coordinate frames

This thesis is using a 2D world to simulate different scenarios. To ease computations and nomenclature, we introduce a set of interlinked coordinate frames to express 2D positions and vectors in the X/Y-plane. In tracking systems, the X and Y directions are commonly referred to as longitude and latitude respectively. All coordinate frames follow the right-hand rule for orientation and the Z-axis is always pointing up from the ground, regardless of the frame. Different frames can be rotated around the Z-axis and translated in the X/Y-plane relative to each other. Rotation is measured as the angle between coordinate frames' X-axis with positive direction defined as anti-clockwise when looking in the direction of the Z-axis. To prevent ambiguity, all angles are always kept between ± 180 degrees.

With the exception of the target coordinate frame, which is introduced in Section 2.3, and the common coordinate frame, introduced in Section 4.1.2, the most used frames are:

- the **World coordinate frame**, denoted by a superscript $[W]$ as seen in Figure 2.1a. It has an arbitrary, set, origin and orientation of its X/Y-axis. The other coordinate frames are defined relative to the world frame.
- the **Ego (Vehicle) coordinate frame**, denoted by a superscript $[E]$. In the context of this thesis ego vehicle refers to the vehicle on which all sensors are mounted on. The coordinate frame origin is tied to the rear axle of the ego vehicle and is oriented such that the X-axis is pointing forward towards the front axle as seen in Figure 2.1b. Hence the ego vehicle coordinate frame is free to move and rotate as the ego vehicle moves. The orientation of the ego vehicle is referred to as the vehicle's yaw.



(a) Vehicle in world frame **(b)** Ego centered frame

Figure 2.1: The vehicle is free to move around and rotate in the world frame as seen in a). The position is measured to the center of the rear axle marked with a dot. The ego centered frame in b) is tied to the rear axle and aligned with the heading.

- the **Sensor coordinate frame**, denoted by a superscript $[S_x]$ where x denotes the sensor ID. Its origin and orientation is tied to the position of the sensor as expressed in the ego vehicle frame with the X-axis pointing straight forward out of the sensor as can be seen in Figure 2.2. The orientation of a sensor is referred to as the sensor's boresight and is denoted by β_x where, again, x denotes the sensor ID.

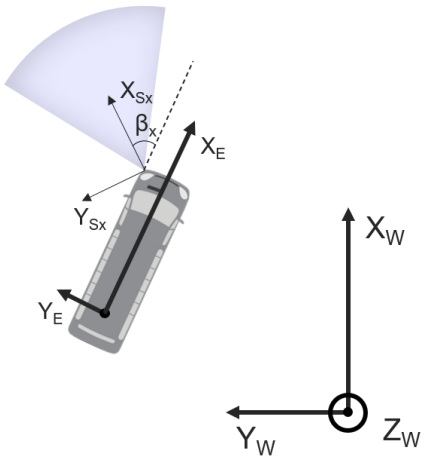


Figure 2.2: All three coordinate frames including the sensor frame.

2.1.1 Conversion between different coordinate frames

Since information expressed in different coordinate frames convey the same information presented from different perspectives, at times it may be useful to transform from one frame to another. To do so, the relative orientation and position of the

coordinate systems needs to be known. Let coordinate frame B be expressed in coordinate frame A, $\theta_B^{[A]}$ denote the rotation of coordinate frame B relative to A, $t_{x,B}^{[A]}$ and $t_{y,B}^{[A]}$ denote the origin of coordinate frame B as expressed in coordinate frame A. Then the transformation from coordinate system A to coordinate system B can be expressed as

$$\begin{bmatrix} x^{[B]} \\ y^{[B]} \end{bmatrix} = \begin{bmatrix} \cos(\theta_B^{[A]}) & \sin(\theta_B^{[A]}) \\ -\sin(\theta_B^{[A]}) & \cos(\theta_B^{[A]}) \end{bmatrix} \left(\begin{bmatrix} x^{[A]} \\ y^{[A]} \end{bmatrix} - \begin{bmatrix} t_{x,B}^{[A]} \\ t_{y,B}^{[A]} \end{bmatrix} \right). \tag{2.1}$$

The process of conversion for positions and velocities is very similar although velocity vectors only need to be rotated whereas positions may need to be rotated and translated. Note, as the transform requires the relative orientation and position between coordinate frames to be known, to transform between e.g. sensor frame and world frame one first has to transform from sensor frame to ego frame and then from ego frame to world frame rather than transforming from sensor to world frame directly.

2.2 Radar basics

Whilst this thesis does not delve deep into the inner workings of radar sensors, it is important to acknowledge the properties of their measurements. Hence this section aims to provide background on radar to motivate some of its basic functionality and its strengths and weaknesses.

2.2.1 Basic range estimation

Fundamentally, radar systems transmit electromagnetic waves which reflect off objects [8]. We will refer to objects of interest as targets. The radar then analyzes the reflected wave to obtain information about the target. A basic system is the pulse radar, which repeatedly emits a short pulse of radio energy as seen in Figure 2.3. The time between transmission and receiving a reflected wave, the time of flight, can then be used to determine the range (denoted r) to the target. The reflected and received wave is commonly referred to as a *detection*.

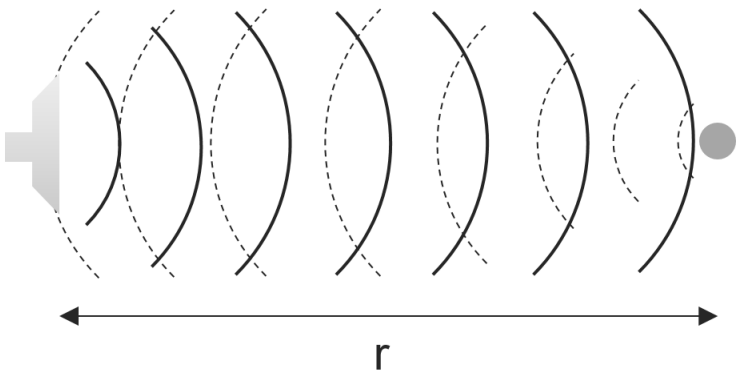


Figure 2.3: Pulse radar obtaining a distance measurement r from the time of flight.

2.2.2 Basic angle estimation

Using a single sensor as shown in Figure 2.3, the radar is unable to differentiate between two targets spaced apart at the same distance. To solve this issue, modern radars typically consist of an array of several receiving antennas as shown in Figure 2.4. Due to the spacing d, the reflected signal will have to travel a slightly longer distance, Δr, between different antennas depending on the origin of the reflected signal. Hence the received signal between antennas will have a phase shift proportional to the additional distance and the signal's wave length. By analyzing the phase shift between antennas, it is possible to compute where the signal originated from [8]. Horizontally spaced antennas can hence solve for the angle of a received signal in the X/Y-plane, commonly referred to as the detection's azimuth which is denoted with θ.

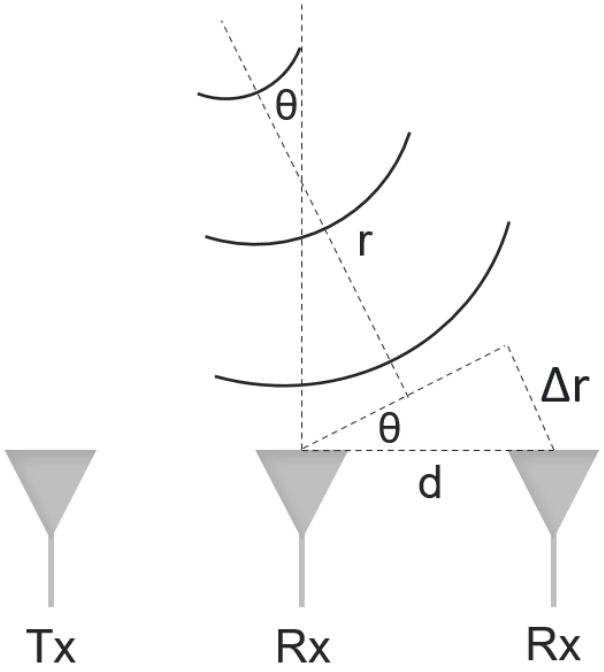


Figure 2.4: Angle estimation using a transmitting antenna, marked Tx and multiple receiving antennas (Rx). The antennas are separated by a distance d resulting in a longer time of flight due to the distance Δr. Consequently, the azimuth θ can be solved for.

2.2.3 Basic velocity estimation

In an automotive setting, it is typically required to have precise information of the target's angle, range *and* velocity at the same time.

A Continuous Wave (CW) radar instead transmits a continuous signal with a known frequency. Due to the Doppler effect, if the target is moving, its reflected signal will have a slight frequency shift compared to the transmitted signal. The difference can then be used to determine the relative velocity between the target and the

radar sensor. This velocity is referred to as the detection's range-rate denoted by $\dot{r}$. The obvious disadvantage of the CW radar is that the target's range can no longer be determined due to the lack of clear separation between received reflections in the signal. This is overcome by the Frequency-Modulated-Continuous-Wave radar (FMCW) which combines the best of the pulse radar and the CW radar to solve for range and range-rate simultaneously [9].

2.2.4 Frequency-Modulated Continuous-Wave Radars

A Frequency-Modulated Continuous-Wave (FMCW) radar is able to continuously transmit a signal with linearly increasing frequency, also known as a chirp signal. The signal is shown in Figure 2.5.

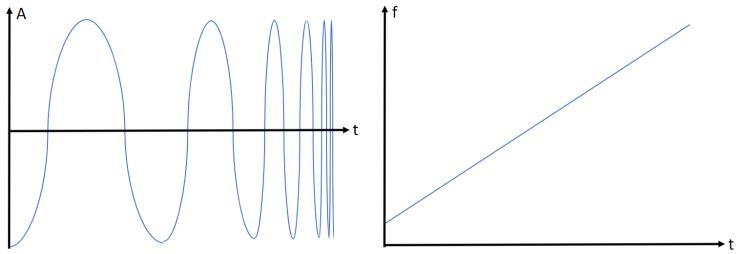


Figure 2.5: A chirp's (blue line) amplitude (A) and frequency (f) over time (t).

The FMCW radar compares the frequency of the transmitted and received chirp, as shown in Figure 2.6, to determine the range and velocity simultaneously. If a target is moving at the same speed as the radar sensor, the received chirp will be identical to the transmitted chirp but with a time delay Δt, resulting in a frequency shift of Δf that is proportional to the range. If the target has a relative velocity to the radar sensor, the received chirp will have a frequency shift due to the Doppler effect. This difference Δf_D is proportional to the velocity of the target. In practice, a different modulation scheme has to be used in order to effectively separate the two frequency differences although the principle remains the same [9].

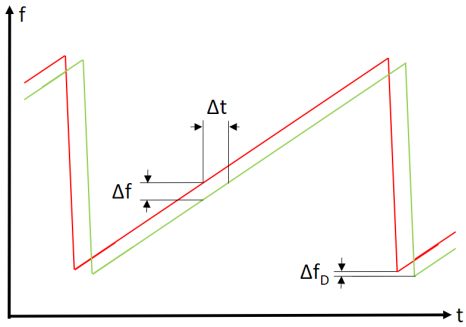


Figure 2.6: FMCW radar signal. Transmitted chirp in red, received chirp in green. The difference in frequency, Δf_D, can be used to determine the velocity of the target.

2.2.5 Radar measurement properties

Due to the nature of a radar detection's components, there are some important properties that should be considered. The positional information from a detection is determined by its range and azimuth. Given a set radar range, and azimuth resolution, the area from which two detections can be separated will form a cone shape with a curved grid. The grid cell size will increase the further away from the sensor origin it is, as seen in Figure 2.7. As a result of this, a distant target will typically receive fewer detections than a target up close to the sensor.

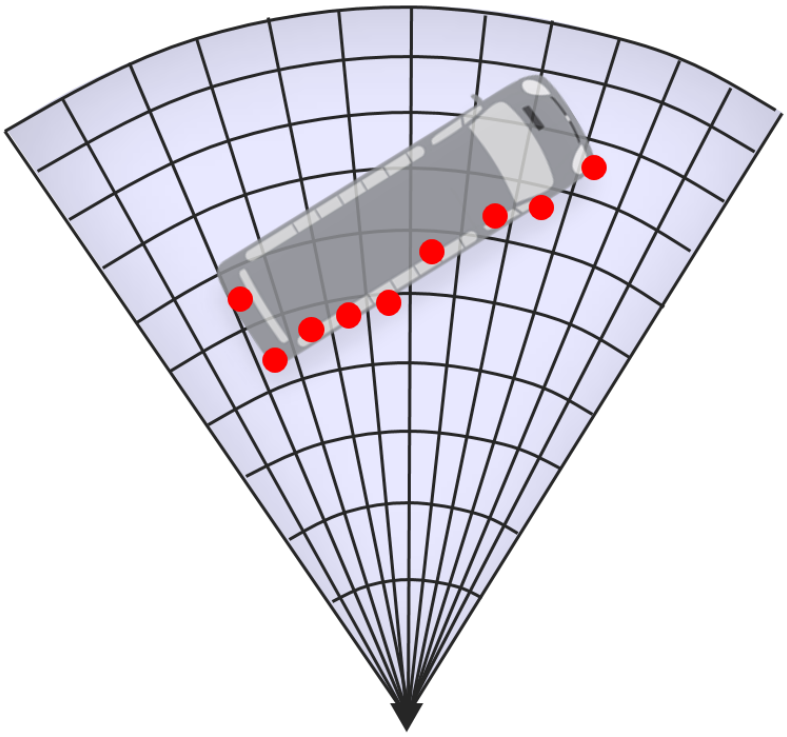


Figure 2.7: The size of the sensor resolution cells increase the further away the target is. At most one detection can be obtained in each cell.

The velocity information of a detection is determined by its positional information in combination with its range-rate. Range-rate is fundamentally a measure of a target's relative velocity in the direction of the sensor, i.e. a *radial velocity* measure. Therefore, the true target velocity is only directly measurable if the target moves linearly (without any rotation) directly towards or away from the sensor as can be seen in the top left in Figure 2.8. If the target is moving while also rotating (e.g. a car in a turn), the range-rate will be a measure of the velocity component that is directed towards the radar sensor as shown in the top right in Figure 2.8. Taken to the extreme, if a target moves perpendicular to the sensor, the velocity component directed towards the sensor will be zero hence the the range-rate measurement will also be zero (or close to) as can be seen in the bottom of Figure 2.8.

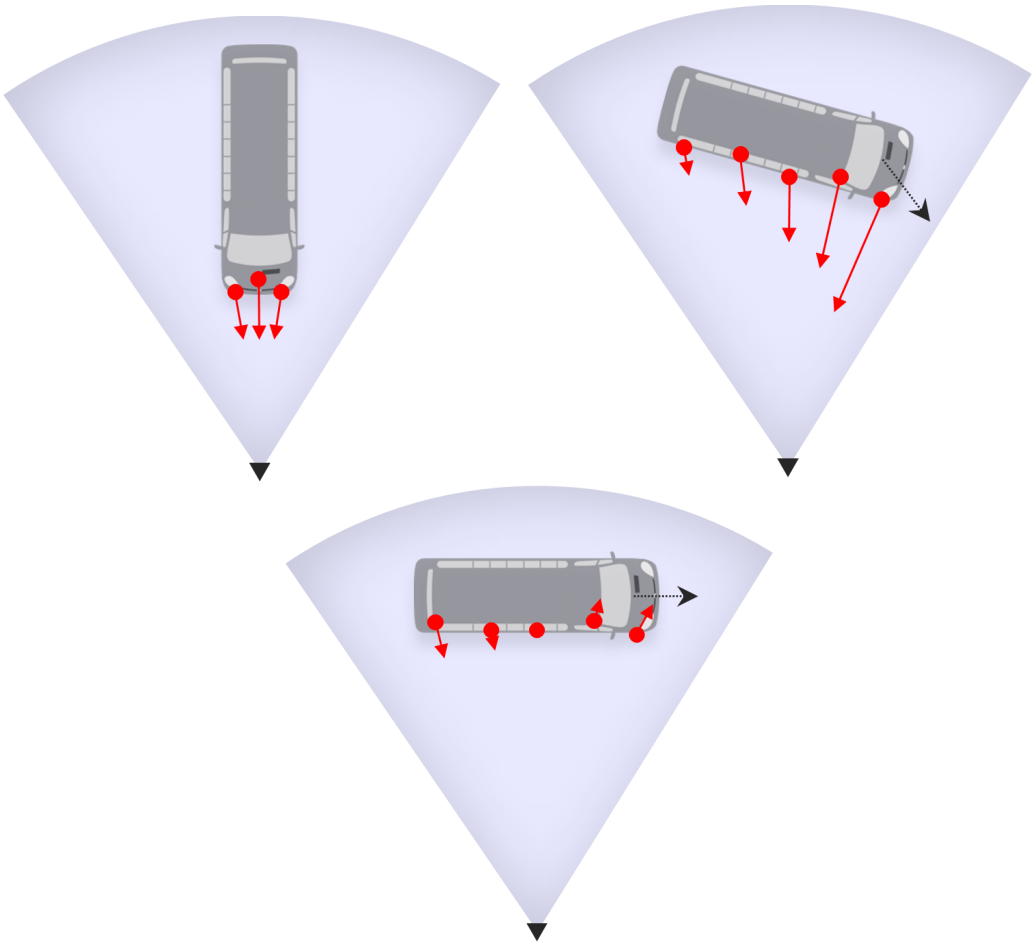


Figure 2.8: The three figures illustrate how the range-rate measurement can differ depending on the movement of the target.

2.2.6 Transform a polar to a cartesian coordinate system

Radar detections in their raw form are commonly expressed in a polar coordinate system using azimuth and range (and range-rate). However, many applications that utilize the information provided by a radar sensor are more suited for use in a cartesian coordinate system. Hence it may be useful to transform radar detections accordingly. Transforming a radar detection expressed in a polar coordinate system yields two components, a position (x, y), and a velocity vector (v_x, v_y). This is given by

$$\begin{bmatrix} x \\ y \end{bmatrix} = r \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} \tag{2.2}$$

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \dot{r} \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} \tag{2.3}$$

where r is the range and $\dot{r}$ is the range-rate. Again, note that the range-rate is the projection of the target's true velocity on to the vector pointing from the radar to the target. The position is simply the combination of azimuth and range as expressed in cartesian coordinates. The velocity vector on the other hand is a vector that is

always directed towards or from the sensor with the length equal to the range-rate as it is a measure of the radial velocity.

2.2.7 Compensation of motion of ego vehicle

In the context of this thesis, the radar sensors will be mounted on the ego vehicle. Hence the sensors will be subject to any motion induced by the vehicle. As range-rate is a measure of the *relative* velocity between a target and the radar sensor, if the ego vehicle is moving at the same speed as the target, the measured range-rate will thus be zero. Typically it is more convenient to express the velocity relative to the ground i.e. as if the sensor is stationary. This can be done through adding a term to the measurement that compensates range-rate induced by ego vehicle motion. To do so, we first compute the velocity of the sensor relative to the ground as induced by the motion of the ego vehicle, denoted by $\boldsymbol{v}_{s_x}^{[S]}$. Let β_x denote the sensor's boresight, $\boldsymbol{v}_E^{[E]}$ the ego vehicle velocity, ω_E the ego vehicle yaw-rate, and L_{s_x} the distance between the sensor placement on the ego vehicle and the vehicle's rotational centre, i.e. the rear axle. Then the velocity of the sensor relative to the ground, in sensor frame, is given by

$$\boldsymbol{v}_{s_x}^{[S]} = \begin{bmatrix} v_{x,s_x}^{[S_x]} \\ v_{y,s_x}^{[S_x]} \end{bmatrix} = \underbrace{\begin{bmatrix} \cos(\beta_x) & \sin(\beta_x) \\ -\sin(\beta_x) & \cos(\beta_x) \end{bmatrix}}_{=\mathcal{R}_{E/S_x}} \underbrace{\left(\begin{bmatrix} v_{x,E}^{[E]} \\ v_{y,E}^{[E]} \end{bmatrix} + \omega_E \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} L_{x,s_x} \\ L_{y,s_x} \end{bmatrix} \right)}_{=\boldsymbol{v}_{s_x}^{E}} \tag{2.4}$$

where $\mathcal{R}_{E/S_x}$ is the rotation matrix from ego to sensor frame and $\boldsymbol{v}_{s_x}^{E}$ is the sensor velocity in ego frame.

Now we can compute how much the sensor is moving in the direction of a detection, i.e. relative to the azimuth θ of the detection. To compute the compensated range-rate $\dot{r}_{comp}$, we simply add this amount to the detection's range-rate denoted $\dot{r}$.

$$\dot{r}_{comp} = \dot{r} + \sqrt{v_{x,s}^{2[S]} + v_{y,s}^{2[S]}} \cos\left(\theta^{[S]} - \tan^{-1}\left(\frac{v_{y,s}^{[S]}}{v_{x,s}^{[S]}} \right) \right) \tag{2.5}$$

2.3 Articulated vehicle model

In this section, we introduce nomenclature to describe the type of articulated vehicle used in this thesis as well as motivate the equations behind its important kinematics. To distinguish the units on different sides of the articulation point we refer to them as *tractor* and *trailer*. The tractor is the unit facing the main (forward) direction of travel whereas the trailer is the unit that trails the tractor. As their kinematics are intertwined, it is useful to be able to exploit the relationship between them in order to use knowledge regarding one of them to draw conclusion in regards to the other. Therefore we seek to ultimately express states of the trailer in tractor states.

2.3.1 Introduction to the articulated vehicle model

The model for articulated vehicles used in this thesis operates under the assumption of Ackermann's steering principle [10] which implies that all wheel pairs as well as wheel axle groups operate without any tire slip and can thus be simplified to a single tire. I.e. each tire's velocity vector is aligned with the respective tire's center-line. E.g. applying this principle to a rigid-body, non-articulated autotruck with a pair of front steering wheels and one, or more, pair(s) of rear wheels would allow the truck's kinematics to be simplified to a bicycle model. This holds true for the tractor of the articulated vehicle however there is a slight difference with the trailer. The trailer does not have a front steering wheel pair of own but is rather connected to the tractor through a friction-free pivot joint at the articulation point. Thus the trailer's kinematics can also be simplified to a bicycle model with the modification that its steering wheel is replaced with the pivot joint such that its direction of travel at the articulation point is determined by where it is connected on the tractor and the tractor's movement. The complete model for the articulated vehicle is visualized in Figure 2.9 with its nomenclature introduced in section 2.3.2

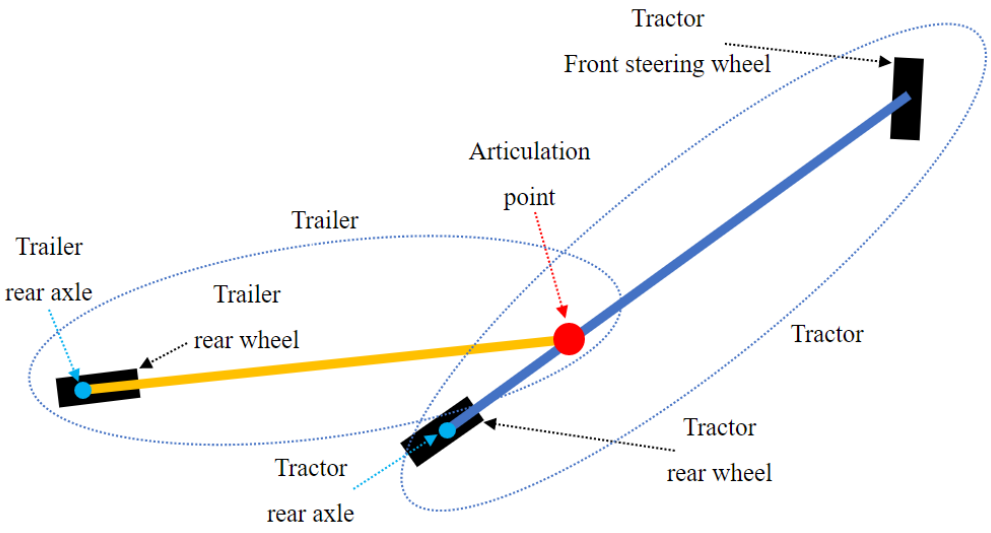


Figure 2.9: Overview of the articulated vehicle model.The solid blue and orange line represent the units' rigid body structure.

2.3.2 Nomenclature

A detailed description of the nomenclature used can be found in Table 2.1. An overview of the model can be seen in Figure 2.11 where subscript 1 denotes the tractor and subscript 2 denotes the trailer with the two units are joined together by a moment-free articulation point denoted p.

L_1	Wheelbase of tractor
R_{1r}	Tractor rear axle turning radius
δ	Steering angle of tractor
(x_1, y_1)	Position of the tractor's rear axle
v_{1r}	Speed of the tractor at its rear axle
ϕ_1	Yaw of the tractor
$\dot{\phi}_1$	Yaw-rate of the tractor
p	Articulation point, where the trailer is connected to the tractor
b	(Signed) Distance between the tractor rear axle & the articulation point p
β_p	Body side-slip of tractor at point p
v_p	Speed at point p
ϕ_A	Articulation angle, difference in yaw between tractor & trailer
$\dot{\phi}_A$	Articulation angle-rate, difference in yaw-rate between tractor & trailer
L_2	Distance between trailer rear axle and articulation point
R_{2r}	Trailer rear axle turning radius
(x_2, y_2)	Position of the trailer's rear axle
v_{2r}	Speed of the trailer at its rear axle
ϕ_2	Yaw of trailer trailer
$\dot{\phi}_2$	Yaw-rate of the trailer

Table 2.1: Table describing the nomenclature used in the articulated vehicle model

For the purposes of this thesis, we refer to the following states as *tractor state*

$$\boldsymbol{x}_1 = \begin{bmatrix} x_1 \\ y_1 \\ v_{1r} \\ \phi_1 \\ \dot{\phi}_1 \end{bmatrix}, \tag{2.6}$$

the following states as *trailer state*

$$\boldsymbol{x}_2 = \begin{bmatrix} x_2 \\ y_2 \\ v_{2r} \\ \phi_2 \\ \dot{\phi}_2 \end{bmatrix}, \tag{2.7}$$

and the following states as *articulation state*

$$\boldsymbol{x}_A = \begin{bmatrix} \phi_A \\ \dot{\phi}_A \end{bmatrix}. \tag{2.8}$$

All states together make up the state of an articulated vehicle. However, it is important to note that by having knowledge of two of them is enough to eventually compute the third using the relationships introduced in the following section.

2.3.3 Kinematic equations

A consequence of Ackermann's steering principle is that wheels will travel in the direction they are pointing. Thus, if the front wheel of a bicycle model is not exactly aligned with its rear wheel, the bicycle model will travel in a circle around a point. This point is referred to as the Instantaneous Center of Rotation (ICR) and can be seen in Figure 2.10. It can be found at the intersection between the two lines that are perpendicular to the front and rear wheel direction of travel.

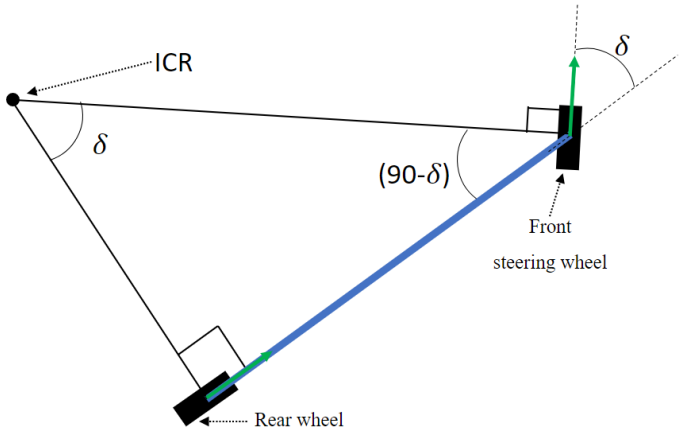


Figure 2.10: A bicycle model with its ICR point for a given steering angle δ. The green arrows indicate the direction of travel at the front and rear wheel respectively.

Utilizing this information, two triangles can be constructed that represent the two ICR points for the tractor and trailer respectively. As the trailer does not have a front wheel of its own, a line that is perpendicular to the direction of travel at the point where it is connected to the tractor (i.e. at the articulation point) is instead used which can be seen in Figure 2.11.

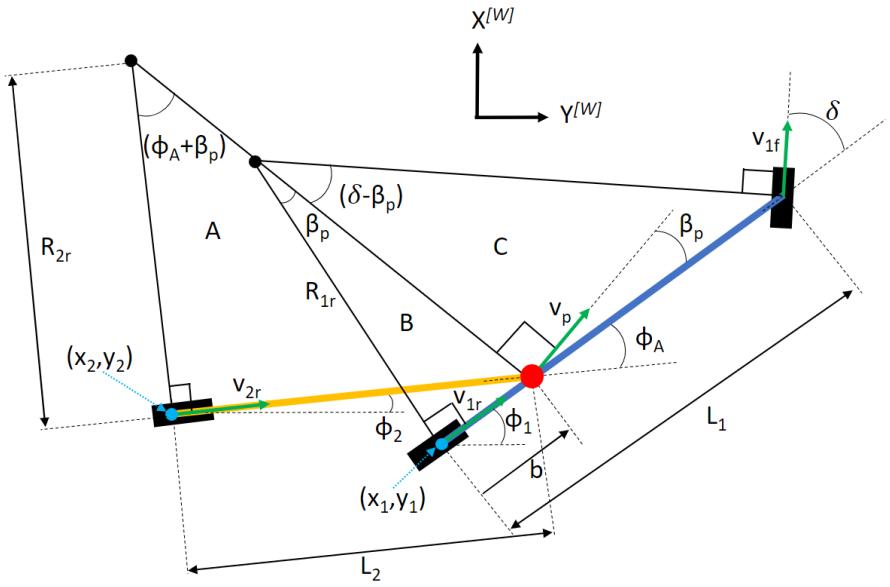


Figure 2.11: Kinematic model of articulated vehicle. The pivot joint at the articulation point is represented by the red dot.

Using triangle B+C in Figure 2.11 we can extract the expression

$$R_{1r} = \frac{L_1}{\tan(\delta)}, \quad 0 < |\delta| < \frac{\pi}{2} \tag{2.9}$$

By combining (2.9) with triangle B we can extract the body-slip angle β_p through

$$\beta_p = \tan^{-1}\left(\frac{b}{R_{1r}}\right) = \tan^{-1}\left(\frac{b\tan\delta}{L_1}\right) \tag{2.10}$$

Using triangle A+B we can extract the expression

$$R_{2r} = \frac{L_2}{\tan(\phi_A + \beta_p)} \tag{2.11}$$

Comparing (2.11) to the tractor steering angle δ in (2.9), the trailer can thus be seen as being steered by the virtual steering angle $(\phi + \beta_p)$.

Now the speed at the articulation point, v_p, can be computed from the geometry

$$v_p = \frac{v_{1r}}{\cos(\beta_p)} \tag{2.12}$$

By computing the velocity component that is perpendicular to the trailer's rear axle at the articulation point we can compute the trailer's speed at its rear axle, v_{2r}, through

$$v_{2r} = v_p \cos(\phi_A + \beta_p) = \frac{v_{1r}}{\cos(\beta_p)} \cos(\phi_A + \beta_p) \tag{2.13}$$

Similarly to the rear axle velocity, the yaw-rate of the trailer, $\dot{\phi}_2$, can be described by computing the velocity component of v_p that is perpendicular to the trailer's body and divide with the length between the articulation point and the trailer rear axle.

$$\dot{\phi}_2 = \frac{v_p \sin(\phi_A + \beta_p)}{L_2} \tag{2.14}$$

$$= \frac{v_{1r} \sin(\phi_A + \beta_p)}{\cos(\beta_p) L_2} \tag{2.15}$$

This can be further simplified by using the rule

$$\sin(a+b) = \sin(a)\cos(b) + \cos(a)\sin(b) \tag{2.16}$$

$$\tag{2.17}$$

to rewrite Equation 2.15 as

$$\dot{\phi}_2 = \frac{v_{1r}}{L_2}\left(\frac{\sin(\phi_A)\cos(\beta_p)}{\cos(\beta_p)} + \frac{\cos(\phi_A)\sin(\beta_p)}{\cos(\beta_p)}\right) \tag{2.18}$$

$$= \frac{v_{1r}}{L_2}\left(\sin(\phi_A) + \cos(\phi_A)\tan(\beta_p)\right) \tag{2.19}$$

$$= \frac{v_{1r}}{L_2}\left(\sin(\phi_A) + \cos(\phi_A)\, b\frac{1}{R_{1r}}\right). \tag{2.20}$$

With

$$v_{1r} = \dot{\phi}_1 R_{1r} \tag{2.21}$$

the expression for the trailer yaw rate is ultimately

$$\dot{\phi}_2 = \frac{v_{1r}}{L_2}\left(\sin(\phi_A) + \cos(\phi_A)\frac{b\dot{\phi}_1}{v_{1r}}\right). \tag{2.22}$$

Through geometry we can express the articulation angle ϕ_A as the difference in yaw between the tractor and trailer.

$$\phi_A = \phi_1 - \phi_2 \tag{2.23}$$

Similarly, the articulation angle-rate can be expressed as the difference in yaw-rate between the tractor and trailer.

$$\dot{\phi}_A = \dot{\phi}_1 - \dot{\phi}_2 \tag{2.24}$$

$$= \dot{\phi}_1 - \frac{v_{1r}}{L_2}\left(\sin(\phi_A) + \cos(\phi_A)\frac{b\dot{\phi}_1}{v_{1r}}\right) \tag{2.25}$$

To describe the trailer's position we introduce the tractor coordinate frame. In this frame, denoted superscript $[T]$, the origin is tied to the tractor's rear axle with the X-axis pointing forward towards the front axle. Then in the tractor frame, the yaw

of the trailer, ϕ_2, is the same angle as the articulation angle ϕ with opposite sign, i.e.

$$\phi_2^{[T]} = -\phi_A. \tag{2.26}$$

By using the articulation angle, the trailer's length L_2, the distance between the tractor rear axle and the articulation point b, we can express the relationship between the tractor's rear axle and the trailer's rear axle through

$$\begin{bmatrix} x_2^{[T]} \\ y_2^{[T]} \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix} + \begin{bmatrix} \cos(-\phi_A) & \sin(-\phi_A) \\ -\sin(-\phi_A) & \cos(-\phi_A) \end{bmatrix} \begin{bmatrix} -L_2 \\ 0 \end{bmatrix}. \tag{2.27}$$

Thus all trailer states have now been expressed in terms of tractor and articulation states.

2.4 Kalman filters

Kalman filters are a type of recursive algorithms that make use of Bayesian statistics combined with several measurements over time of unknown variables in a system to produce an estimate of the variables that tend to be more accurate than using a measurement from a single time-instance. The filter assumes the underlying system

$$\boldsymbol{x}_k = f\left(\boldsymbol{x}_{k-1}\right) + \boldsymbol{q}_k \tag{2.28}$$

$$\boldsymbol{y}_k = h\left(\boldsymbol{x}_k\right) + \boldsymbol{r}_k \tag{2.29}$$

where $\boldsymbol{x}_k$ is the state and $\boldsymbol{y}_k$ is the measurement at time step k. The previous state is linked to the next state by a process or motion model f along with some Gaussian process noise $\boldsymbol{q}_k$. Similarly, the measurement is linked to the state via a measurement function h with some Gaussian measurement noise $\boldsymbol{r}_k$. In this thesis, both of these functions are non-linear and hence a non-linear Kalman filter is used. One of the non-linear extensions of the Kalman filter algorithms is the Extended Kalman Filter (EKF), which uses linearized motion and measurement models. [11]

2.4.1 The Extended Kalman Filter

In the EKF, the estimated state is computed recursively by using the previously estimated state and the current measurement. The filter estimate consists of two parts, an estimate of the state $\hat{\mathbf{x}}_{k|k}$ along with an uncertainty in the estimate $\mathbf{P}_{k|k}$. The double indexing $k|k$ is interpreted as the estimate at time k, given measurements up to and including time k. The reason for this indexing is that the filter is usually split up in to two steps: the prediction and the update step. In the prediction step, the filter propagates the previous state estimate through the motion model to predict the next state estimate as

$$\hat{\boldsymbol{x}}_{k|k-1} = f\left(\hat{\boldsymbol{x}}_{k-1|k-1}\right). \tag{2.30}$$

The covariance, or uncertainty in the predicted state, is computed by the EKF as

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_k \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_k^\top + \boldsymbol{Q} \tag{2.31}$$

where $\boldsymbol{F}_k = \frac{\partial f}{\partial \boldsymbol{x}}\Big|_{\hat{\boldsymbol{x}}_{k-1|k-1}}$ is the jacobian of the motion model and $\boldsymbol{Q}$ is a tuning matrix containing the (unknown) variance of the states. This completes the prediction step. In the update step, the EKF computes a predicted measurement $h\left(\hat{\boldsymbol{x}}_{k|k-1}\right)$ and compares it to the actual measurement $\boldsymbol{y}_k$

$$\boldsymbol{v}_k = \boldsymbol{y}_k - h\left(\hat{\boldsymbol{x}}_{k|k-1}\right) \tag{2.32}$$

to form the measurement residual or *innovation* $\boldsymbol{v}_k$. The covariance of the innovation, $\boldsymbol{S}_k$, is then computed by

$$\boldsymbol{S}_k = \boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^\top + \boldsymbol{R} \tag{2.33}$$

where $\boldsymbol{H}_k = \frac{\partial h}{\partial \boldsymbol{x}}\Big|_{\hat{\boldsymbol{x}}_{k|k-1}}$ is the jacobian of the measurement model and $\boldsymbol{R}$ is another tuning matrix with the variance of the measurement noise. The Kalman gain can then computed as

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^\top \boldsymbol{S}_k^{-1}. \tag{2.34}$$

Finally, the update step is completed by shifting the predicted state towards the measurement and shrinking the predicted covariance using the Kalman gain

$$\begin{aligned} \hat{\boldsymbol{x}}_{k|k} &= \hat{\boldsymbol{x}}_{k|k-1} + \underbrace{\boldsymbol{K}_k \boldsymbol{v}_k}_{shift} \\ \boldsymbol{P}_{k|k} &= \boldsymbol{P}_{k|k-1} - \underbrace{\boldsymbol{K}_k \boldsymbol{H}_k \boldsymbol{P}_{k|k-1}}_{shrinkage}. \end{aligned} \tag{2.35}$$

2.5 Model estimation

In many stages of the tracking algorithm presented in Section 4, there is a need to estimate a mathematical model from measurements affected by noise. The RANdom SAmple Consensus (RANSAC) algorithm does just that.

2.5.1 RANSAC

RANSAC is an iterative algorithm that is used to estimate the parameters of a model given measurements, λ, of the model [12]. As with all real sensor data, measurements will be corrupted by noise. Hence it is very unlikely that any estimated model will fit all measurements perfectly, causing *residuals*. A residual, res_i, is defined as the deviation between the measurement i and the estimated model. The algorithm operates under the assumption that the data will contain some measurements that have been affected by a lot of noise and therefore have a large residual from the true model. Measurements whose residual is such that $res > \tau$, where τ is a hyperparameter, are referred to as *outliers*. The main idea of the method is to estimate model parameters such that the number of outliers is minimized. This

makes RANSAC more robust to data with noisy measurements compared to e.g. minimizing the least square of the residuals, where a single outlier amongst otherwise perfect measurements could skew the entire model.
A difficulty with minimizing outliers is that it is a non-continuous function of λ. To solve this, RANSAC operates as described in Algorithm 1.

Algorithm 1: General RANSAC algorithm

Data: N measurements λ, threshold τ
Result: The best model M_{best} with least amount of outliers
$k = 0$;
$minNumOutliers = N$;
while $k < k_{max}$ **do**
    Randomly sample a minimal subset λ_s out of λ;
    Estimate a model M_s using λ_s;
    Compute $numOutliers$ using M_s;
    if $numOutliers < minNumOutliers$ **then**
        $M_{best} = M_s$;
        $minNumOutlier = numOutliers$;
    $k = k + 1$;
return M_{best};

One of the drawbacks of this approach is that the algorithm is non-deterministic due to the random selection of measurements. However, the idea is that with more iterations of the RANSAC sequence, there is a increase in probability for a suitable subset of measurements to be selected, and in turn, an increase in probability to estimate a suitable model.

2.6 Clustering

As described in Section 2.3, the dynamics between the tractor and the trailer of an articulated vehicle are interlinked. This implies that there may be a difference between the tractor and the trailer states. Therefore, it would be useful to compute these states separately by splitting radar measurements into two groups, measurements originating from the tractor and measurements originating from the trailer. One method of achieving such is to make use of clustering. This thesis utilizes a method named K-means clustering.

2.6.1 K-means clustering

K-means clustering is an algorithm that is used to split a multi-dimensional data set into a set number of clusters such that it minimizes the within-cluster squared Euclidean distances. I.e., it partitions a data set such that the resulting clusters' data points are as close as possible to their respective cluster's mean. A cluster's mean in the context of K-means is commonly referred to as its centroid. Given a desired number of clusters K, the algorithm operates such that it initially randomly selects

K data points and labels these as cluster centroids. The remaining data points are then assigned to each cluster based on the minimum distance to a centroid as given by a distance function. For the purposes of this thesis, the distance function is the squared physical distance (i.e. meters) of the position of a radar detection to the centroid. With all data points now assigned to a cluster, the cluster centroids are recomputed by calculating the mean of all points in each cluster respectively. The process of assigning data points to the closest centroid and recomputation of the centroid is then repeated until convergence. The algorithm is summarized in Algorithm 2.

Algorithm 2: K-means clustering algorithm

Data: A set of data points, desired number of clusters K
Result: Data points split into K clusters
Randomly select K data points to be cluster centroids;
repeat
    Compute squared euclidean distances to centroids for all datapoints;
    Assign data points to the closest centroid/cluster;
    Recompute centroids based on all data points in cluster;
until *convergence*;
return Clustered data;

3
Simulation

A simulation environment is required in order to generate ground truth data for the vehicles along with synthetic radar detections. The ground truth data is the true state of the vehicle at any given time step and can be used for benchmarking purposes whereas the radar detections are used as input to the tracking algorithm introduced in Section 4. The articulated vehicle model derived in Section 2.3 is of relatively low complexity, e.g. it does not incorporate dynamic movement, and thus the Automated Driving Toolbox from MATLAB was suitable. The toolbox has built-in support for generating synthetic radar measurements and simulating movement of non-articulated vehicles. However, as this thesis concerns articulated vehicles, some additions have to be made. Hence this section will first introduce how the toolbox is simulating the measurements of a real radar and then delve into how the toolbox simulates vehicles and the additions made by this thesis.

3.1 Synthetic radar measurements

Synthetic radar measurements are generated by the class `radarDetectionGenerator()` [13]. To use it, a radar object is first instantiated using a set of sensor parameters such as placement on ego vehicle and resolution. An excerpt of some of the important parameters used can be seen in Table 3.1. After the radar object has been instantiated, it can be called on as a function to generate radar detections.

Parameter	Description
'UpdateInterval'	Time between each radar mesurement
'Yaw'	The sensors rotation relative to the ego vehicle.
'FieldOfView'	The area the sensor can cover from its perspective.
'MaxRange'	Maximum range at which a detection can be generated.
'AzimuthResolution'	Minimum resolvable azimuth.
'HasElevation'	Enables elevation measurements. Is set to false.
'DetectionProbability'	Probability a sensor resolution cell yields a detection.

Table 3.1: Table explaining *some* crucial sensor parameters

When the radar object is called to generate radar detections, firstly, it creates a sensor resolution grid, as discussed in 2.7, from the provided azimuth, range and range-rate resolution limits. The coverage of the vehicle inside this grid will determine the amount of active sensor resolution cells. Each cell is then evaluated by a

probability of detection, which is proportional to the range to the target and the target Radar Cross Section (RCS). RCS is a measure of how "visible" the target is for a radar and is dependent on many factors such as the angle a detection is reflected at, material properties of the target and many other factors [14]. Finally, to simulate imperfect detections, Gaussian distributed noise will be added to the positional and range-rate measurements. The noise itself is computed differently and independently for every measurement type (i.e. azimuth, range, range-rate) although the common and underlying factor is the RCS. The end result is a data matrix containing the information about the simulated radar detections in polar frame for a single scan from a single radar. For convenience, the information is also transformed into the Cartesian frame using the transform introduced in section 2.2.6 and stored alongside the polar frame. This is then repeated and aggregated for each radar sensor for each time step. Let t_k denote time step k, j denote radar sensor number j, i denote the i:th measurement, N_j denote the number of detections returned from sensor j in one time step, then the final output can be seen in Table 3.2.

t	$\theta_{i,j}$	$r_{i,j}$	$\dot{r}_{i,j}$
t_k	$\theta_{1,1}$	$r_{1,1}$	$\dot{r}_{1,1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
t_k	$\theta_{N_1,1}$	$r_{N_1,1}$	$\dot{r}_{N_1,1}$
t_k	$\theta_{1,2}$	$r_{1,2}$	$\dot{r}_{1,2}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
t_k	$\theta_{N_2,2}$	$r_{N_2,2}$	$\dot{r}_{N_2,2}$
t_{k+1}	$\theta_{1,1}$	$r_{1,1}$	$\dot{r}_{1,1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
t_{k+1}	$\theta_{N_1,1}$	$r_{N_1,1}$	$\dot{r}_{N_1,1}$
t_{k+1}	$\theta_{1,2}$	$r_{1,2}$	$\dot{r}_{1,2}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
t_{k+1}	$\theta_{N_2,2}$	$r_{N_2,2}$	$\dot{r}_{N_2,2}$

Table 3.2: Data matrix containing the radar detections from time step t_k and t_{+1} for a setup using 2 radars.

3.1.1 Radar setup

Although it is common to utilize several radars to cover a 360 degree area around the ego vehicle for automotive tracking purposes, this thesis will only utilize a radar setup with 2 radars at once. This decision is made solely for the simplicity to be able skip the implementation of the logic needed to handle scenarios where the target vehicle is seen from more sensors than necessary and vice versa.

3.2 Motion simulation

Matlab's automated driving toolbox natively only supports non-articulated vehicles. To enable the simulation of articulated vehicles, two separate vehicles are simulated, one representing the tractor unit and one representing the trailer unit. The tractor is simulated using the toolbox's methods to simulate vehicles. The idea behind the simulation of the trailer unit is to exploit the state of the tractor along with the articulated vehicle kinematic model introduced in 2.3 in each time step to compute the resulting trailer state and feed it manually to the simulation. However, in order to determine the trailer state, the articulation angle between the units has to be known. Hence this section will first introduce what constitutes to a vehicle in the toolbox, how it is simulated using the toolbox's methodology, and then describe the additions made to simulate the articulation angle.

3.2.1 Vehicle simulation

Inside the simulation, a vehicle has a state vector

$$\boldsymbol{x} = \begin{bmatrix} x \\ y \\ v \\ \phi \\ \dot{\phi} \end{bmatrix} \tag{3.1}$$

where (x, y) denotes the position of a vehicle's rear axle, v denote the speed of the vehicle, ϕ the yaw of the vehicle, and $\dot{\phi}$ the yaw-rate. It also has the parameters

Parameter	Description
Wheelbase	Distance between the front and rear axles
Front overhang	Distance between the front of the vehicle and the front axle
Rear overhang	Distance between the rear axle and the rear of the vehicle

Table 3.3: Standard vehicle parameters

the graphical representation of which can be seen in Figure 3.1

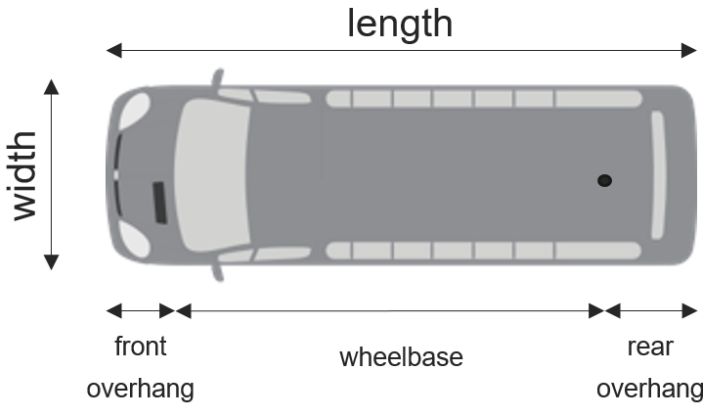


Figure 3.1: Vehicle object parameters in MATLAB's Automated Driving Toolbox.

In addition to the parameters in table 3.3, each vehicle has a number of parameters for its RCS however, for the purposes for this thesis, these will all be kept at the default setting. A vehicle will follow a trajectory that is defined by specifying a set of way-points. The toolbox will then use the way-points to compute a spline for the vehicle's rear axle to follow as seen in Figure 3.2.

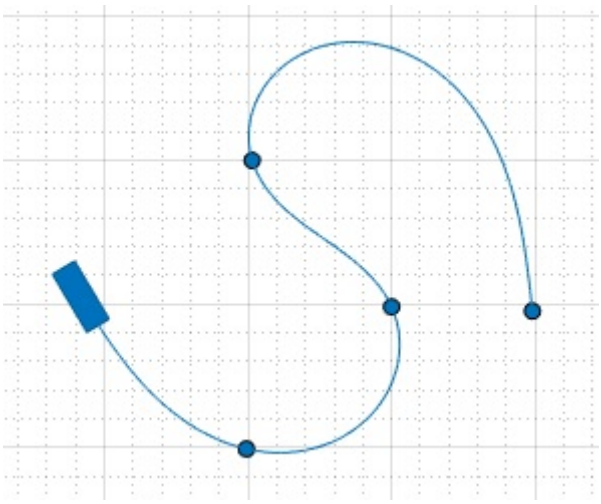


Figure 3.2: Example of a generated vehicle trajectory in the toolbox.

In addition to position, each way-point also contains a set speed for the vehicle. I.e. the vehicle's speed will exactly match the set speed at the waypoint. The toolbox will accelerate vehicles using constant acceleration between different speeds between different way-points. The vehicle's yaw will follow the trajectory's tangent in each time step and the yaw-rate is the result of the vehicle's speed and movement along the trajectory. The resulting ground truth data is hence generated on the format as shown in Table 3.4.

$\boldsymbol{t}$	$\boldsymbol{x}$	$\boldsymbol{y}$	$\boldsymbol{v}_x$	$\boldsymbol{v}_y$	$\boldsymbol{\phi}$	$\dot{\boldsymbol{\phi}}$
t_0	x_0	y_0	v_{x_0}	v_{y_0}	ϕ_0	$\dot{\phi}_0$
			$\vdots$			
t_k	x_k	y_k	v_{x_k}	v_{y_k}	ϕ_k	$\dot{\phi}_k$

Table 3.4: Ground truth data for a single unit where ϕ_k is the yaw and $\dot{\phi}_k$ is the yaw-rate at time step k.

3.2.2 Simulating the articulation angle

To simulate the articulation angle ϕ_A, it and the articulation angle rate $\dot{\phi}_A$ is introduced as their own states and kept track of manually outside the toolbox simulation. To do so, we exploit the equations that models articulated vehicles introduced in section 2.3. The articulation angle rate is given by

$$\dot{\phi}_A = \dot{\phi}_1 - \dot{\phi}_2. \tag{3.2}$$

By expanding the yaw-rate of the trailer $\dot{\phi}_2$, we get

$$\dot{\phi}_A = \dot{\phi}_1 - \frac{v_{1r}}{L_2}\left(\sin(\phi_A) + \cos(\phi_A)\frac{b\dot{\phi}_1}{v_{1r}}\right) \tag{3.3}$$

i.e. an Ordinary Differential Equation (ODE) with the tractor states as input. As the articulated vehicle model is of relatively low complexity (e.g. it does not incorporate any dynamic relationships), a highly accurate solution to the ODE does not yield additional benefits. Given the requirements, a natural choice of solution method is the forward Euler method. Hence given an initial articulation angle, equation 3.3 can be used to compute the articulation angle for every time step. Let t_k denote a variable at time k and T the time step size, then the forward Euler method solves ODEs through

$$x_{t_k} \approx T\dot{x}_{t_{k-1}} + x_{t_{k-1}}. \tag{3.4}$$

Hence, in practice, an initial articulation angle $\phi_{t_0,A}$ is used as an initial condition for simulated scenarios. The articulation angle for all following time steps are computed by first calculating the trailer yaw-rate using the previous articulation angle and the input from the tractor from the simulation through

$$\dot{\phi}_2 = \frac{v_{t_k 1r}}{L_2}\left(\sin(\phi_{t_{k-1},A}) + \cos(\phi_{t_{k-1},A})\frac{b\phi_{t_k,1}}{v_{t_k,1r}}\right) \tag{3.5}$$

and articulation angle rate through

$$\dot{\phi}_A = \dot{\phi_{t_k,1}} - \dot{\phi}_2 \tag{3.6}$$

and ultimately the articulation angle through

$$\phi_{t_k,A} = T\dot{\phi}_A + \phi_{t_{k-1,A}}. \tag{3.7}$$

Then ultimately $\phi_{t_k,A}$ is used to compute all other states such as trailer states and the articulation angle rate.

4
Tracking algorithm

In this chapter, the full tracking algorithm and state estimation techniques are presented. At its core, the algorithm tries to partition the radar detections into two groups based on the origin of the reflection, i.e. the tractor or the trailer, through the use of clustering. The algorithm then labels each cluster as the tractor or the trailer. The detections are then processed using various techniques to estimate the state of each unit separately. The states are then fed to a Kalman filter that combines the information from both units with a motion model to yield a final state estimate of the target vehicle as a whole.

A simplified overview of the algorithm can be seen in Figure 4.1

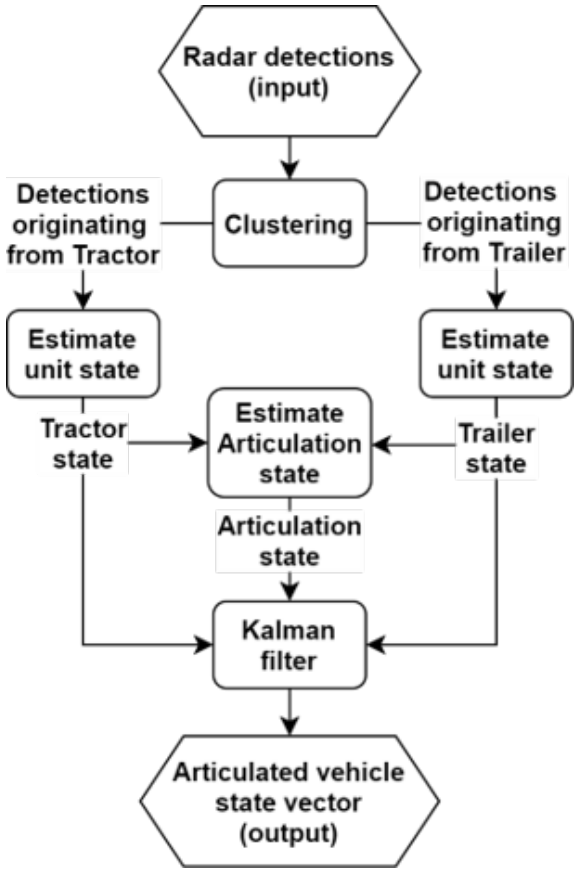


Figure 4.1: Flowchart showing the operation of the tracking algorithm.

where each stage will be explained in the coming sections below.

4.1 State estimation methods

In this section we present the estimation methods used to determine the state of the articulated target vehicle as introduced in section 2.3. For clarity, these are divided into positional as well as motion estimation methods. The methods are the same regardless of the unit that are estimated thus this section is presented assuming the detections originate from a single unit without loss of generality. The methods of partitioning the radar detections are introduced in Section 4.2. However, there are times when the algorithm is unable to partition radar detections. In these scenarios, the same methods are still utilized although with slight modifications as presented in section 4.4.2.

4.1.1 Position and orientation estimation

To estimate the position and orientation of the target, first, a bounding-box representing the outline of the target is fitted to the radar measurements and then, using the bounding-box, the x- & y-position is estimated as well as the yaw of the unit. As the articulated vehicle model assumes Ackermann's condition, a natural choice is to use the center of the rear axle as the reference point for the target's x- & y-position as the velocity at this point is aligned with the heading of the target.

4.1.1.1 Bounding box estimation

A vehicle unit generally gives a distinct detection pattern resembling an L-shape as shown in Figure 4.2. In an edge case, where only one side of the vehicle is seen, the detection pattern is more of an I-shape. The following algorithm tries to fit the seen sides as accurately as possible, and then extrapolates to estimate a complete bounding box of the target vehicle. The underlying principles were first proposed in [15].

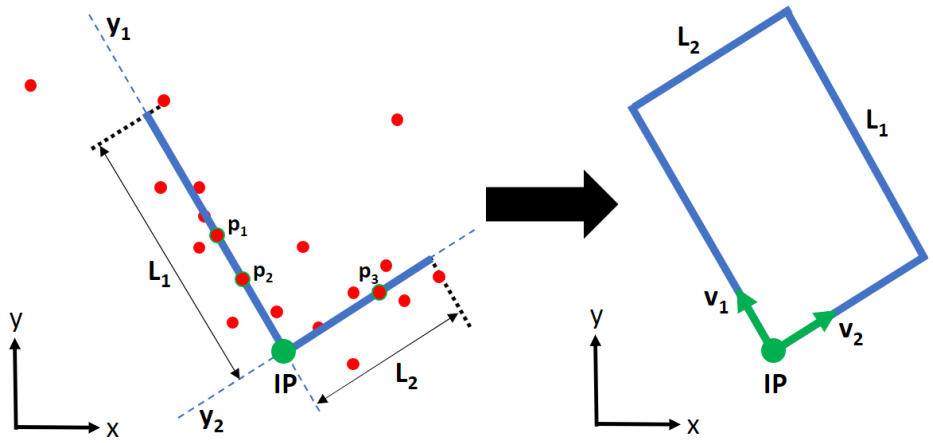


Figure 4.2: Example of L-shape bounding box

The workflow of the algorithm is as follows:

RANSAC is run with 3 sampled points, p_i, where two of the points are used to construct a line y_1 as

$$\boldsymbol{y}_1 = k\boldsymbol{x} + m_1. \tag{4.1}$$

The points that are regarded as inliers to the line y_1 are removed. The remaining sampled point is then used to fit another line y_2, which is perpendicular to y_1 given by

$$\boldsymbol{y}_2 = -\frac{1}{k}\boldsymbol{x} + m_2. \tag{4.2}$$

The intersection point $(x_{\text{IP}}, y_{\text{IP}})$ between the lines is then computed as seen in Figure 4.2, and is an estimation of one of the vehicles corners. Next, the correct side lengths have to be identified and this is done by selecting the inlier furthest away from the intersection point, along both lines respectively. The distances to each of the points are computed as l_1 and l_2 and the largest distance is regarded as being the long side of the vehicle and both sides are set to predefined lengths accordingly. The algorithm finalizes by computing unit vectors $\boldsymbol{v}_1$, $\boldsymbol{v}_2$ pointing away from the intersection point and then assigning them to $\boldsymbol{v}_{\text{long}}$ and $\boldsymbol{v}_{\text{short}}$, to be used in upcoming estimation methods.

In edge cases, where only one side of the vehicle is seen, the line y_1 will perfectly fit the points and thus y_2 can't be computed directly. Instead, the algorithm will save y_1 only and compute the maximum distance between the inliers to the line. If this distance is larger than a threshold value it is regarded as the long side of the vehicle and otherwise the short side. The algorithm then chooses the extreme inlier point furthest away from the ego vehicle, and regards this as a corner point of the vehicle unit. A perpendicular line is then computed through the corner point with appropriate length. The direction of the line is set such that it is pointing away from the ego vehicle. This can be achieved by constructing a vector $\boldsymbol{v}_{EgoToIP}$ from the ego vehicle to the intersection point and then taking the dot product with the line vector s.t

$$\boldsymbol{v}_{EgoToIP} \cdot \boldsymbol{v}_{line} > 0. \tag{4.3}$$

The end result of the bounding box algorithm is shown to the right in Figure 4.2 and the following metrics are saved in world frame

x_{IP}	y_{IP}	$\boldsymbol{v}_{\text{long}}$	$\boldsymbol{v}_{\text{short}}$	width	length

where v_{long} and v_{short} are associated to the long and short side of the vehicle. The

width and length are parameters that are known a priori.

Algorithm 3: Algorithm used to compute bounding box

Data: [x,y] components from radar detections
Result: Bounding box metrics
Sample 3 points;
$\boldsymbol{y}_1, \boldsymbol{y}_2$ = RANSAC(x,y);
if $\boldsymbol{y}_2$ *could not be fit* **then**
  Identify seen side;
  Select corner from extreme inlier on $\boldsymbol{y}_1$;
  Construct $\boldsymbol{y}_2$ through corner point;
Calculate intersection point between $\boldsymbol{y}_1, \boldsymbol{y}_2$;
Assign l_{long}, l_{short} and v_{long}, v_{short} to each line.

4.1.1.2 Reference point estimation

In this section, a method for determining the reference point from the previously computed bounding box is presented.

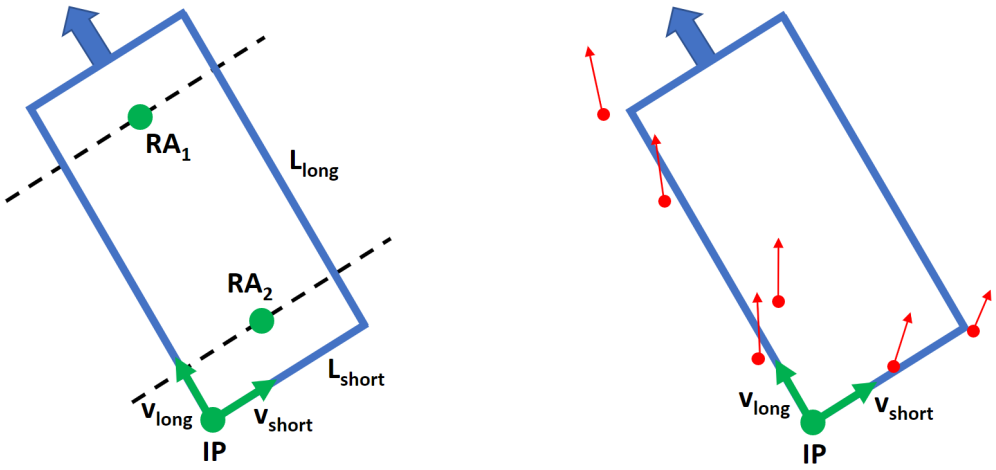


Figure 4.3: The possible locations of the rear-axle (RA_1 and RA_2), given a bounding box.

The rear axle can have one of two possible locations as seen in Figure 4.3, depending on whether the target is moving towards or away from the ego vehicle. To determine this, the measurements are converted to world frame and the average velocity vector $\bar{\boldsymbol{v}}^{[W]}$ is computed and projected onto the long side as seen in Figure 4.3. The projection results in a dot product

$$\boldsymbol{v}_{long} \cdot \bar{\boldsymbol{v}}^{[W]} = \begin{cases} > 0, & \text{if moving away from ego.} \\ < 0, & \text{if moving towards ego.} \end{cases} \tag{4.4}$$

which is used to determine the rear axle location. The rear axle is placed such that it obeys the forward motion of the vehicle, i.e, if the vehicle is moving as in Figure 4.3, the rear axle is placed in location 2. This implies that the vehicle is not allowed

to reverse, as the rear axle will be placed in the wrong location. The location is set such that it aligns with the start of the rear overhang of the vehicle, as seen in Figure 3.1.

4.1.1.3 Yaw estimation

To determine the yaw of the unit, the long side vector $\boldsymbol{v}_{long}$ from the bounding box estimation is used. The heading vector is $\pm\boldsymbol{v}_{long}$ and to determine the correct sign, the heading vector can be computed as

$$\boldsymbol{v}_{heading} = sign(\boldsymbol{v}_{long} \cdot \bar{\boldsymbol{v}}^{[W]})\boldsymbol{v}_{long} \tag{4.5}$$

where the sign function is used to orient the heading vector correctly. The yaw ϕ is then computed by calculating the angle to the x-axis, in world frame as

$$\phi = \cos^{-1}\left(\boldsymbol{v}_{heading} \cdot \begin{bmatrix}1\\0\end{bmatrix}\right). \tag{4.6}$$

4.1.2 Motion estimation algorithm

To estimate the yaw-rate and speed of the target, we utilize a motion estimation method first proposed by Dominik Kellner et al. in [16]. This method computes both the target's yaw-rate as well as its velocity vector in a single time step by only utilizing radar detections' azimuth and compensated range-rate. It does such by analyzing the compensated range-rate as spread over the azimuth, here-on referred to as the target's *velocity profile*. During arbitrary movement, the velocity profile of a rigid target will form a sinusoidal. This can in turn can be exploited to compute the motion of the target. Kellner et al. presents two variations of their motion estimation algorithm that can yield reduced complexity at the expense of accuracy. For the purposes of this thesis, we have implemented what is referred to as the Least-Square Algorithm in a Transformed coordinate system (LSQ-T) which is the middle ground in terms of complexity and accuracy.

4.1.2.1 Transformed coordinate system

The transformed coordinate frame used in LSQ-T will be referred to as the *Common frame* (x', y') which has its origin in one of the sensors and intersects the other sensor, as seen in Figure 4.4. In the following derivations, the information expressed in this frame will be marked with a superscript [C].

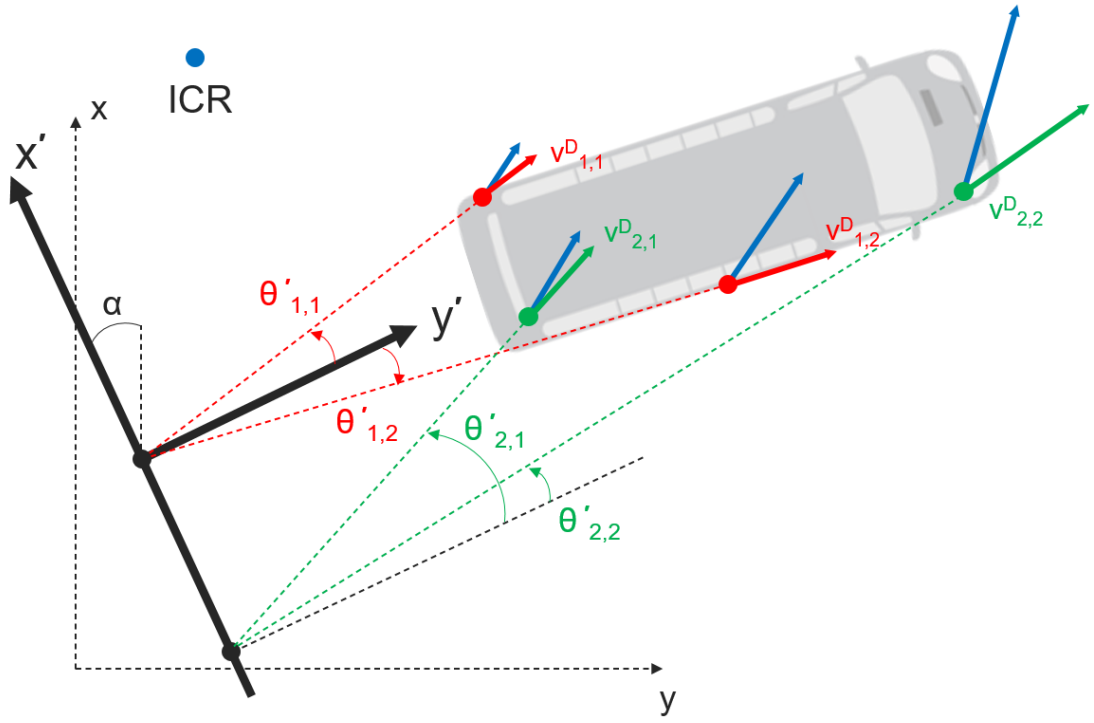


Figure 4.4: The ego vehicle frame (x, y) can be transformed to the common frame (x', y'). Ultimately, the detection velocities and azimuths are expressed in this frame. The red detections belong to the first sensor and the green to the second sensor and the ICR point marks the centre of rotation of the object.

I.e. the sensors' coordinate frame have been rotated such that the two sensors' Y-axis align. This will allow for an accuracy improvement compared to using the two individual sensors' frames, especially if one sensor yields a lot less detections than the other. The reason as to why will be explained in section 4.1.2.2. To convert unprocessed radar measurements to this frame we go through two steps, starting with aligning the sensors' coordinate frame with the ego vehicle coordinate frame. This is achieved through by adding the sensor's boresight to all azimuth measurements. Using the same notation as in section 3.1 where j denotes sensor number j and i denotes the i:th measurements, then

$$\theta_{j,i}^{[E]} = \theta_{j,i}^{[S_j]} + \beta_j \tag{4.7}$$

To align the sensors' Y-axis, we compute the required rotation α using the sensors' position, $(x_{S_1}^{[E]}, y_{S_1}^{[E]})$ and $(x_{S_2}^{[E]}, y_{S_2}^{[E]})$ and subtracting it from the azimuth measurements.

$$\alpha = \tan^{-1}\left(-\frac{x_{S_2}^{[E]} - x_{S_1}^{[E]}}{y_{S_2}^{[E]} - y_{S_1}^{[E]}}\right) \tag{4.8}$$

$$\theta_{j,i}^{[C]} = \theta_{j,i}^{[E]} - \alpha \tag{4.9}$$

Note that the common frame is tied to the ego vehicle frame with origin $(x_{S_1}^{[E]}, y_{S_1}^{[E]})$ and rotation α. Hence to convert Cartesian positions and vectors from the ego vehicle frame to the common frame, we utilize the coordinate frame transform described in section 2.1.1.

4.1.2.2 Velocity profile motivation

As explained in section 2.2.5, the range-rate measurement is a measure of the radial velocity vector, i.e. a component of the true velocity vector of the target. At any point on the target, the true velocity vector is determined by the Instantaneous Center of Rotation (ICR) and the target's yaw-rate $\dot{\phi}$. The ICR is the point which the target is circling around at a specific time instance. The true velocity vector $\boldsymbol{v}_{i,j,P}^{[C]}$ for an arbitrary point $(x_{i,j,P}^{[C]},\ y_{i,j,P}^{[C]})$ on the target is hence given by

$$\begin{bmatrix} v_{x,i,j,P}^{[C]} \\ v_{y,i,j,P}^{[C]} \end{bmatrix} = \dot{\phi} \begin{bmatrix} y_{ICR}^{[C]} - y_P^{[C]} \\ x_P^{[C]} - x_{ICR}^{[C]} \end{bmatrix} = \dot{\phi} \begin{bmatrix} y_{ICR}^{[C]} - r_{i,j,comp} \sin\left(\theta_{i,j}^{[C]}\right) - y_j^{[C]} \\ x_j^{[C]]} + r_{i,j,comp} \cos\left(\theta_{i,j}^{[C]}\right) - x_{ICR}^{[C]} \end{bmatrix}, \quad \dot{\phi} \neq 0 \quad (4.10)$$

It is important to note that Equation 4.10 can only describe the movement of a target which instantaneous movement can be described by a circle, i.e. a target with a yaw rate $\neq 0$. Hence the relationship does not hold for purely linear movement and additional methodology would be required to describe the linear movement exactly. However, in practice, a near linear movement can be approximated by computing a very small yaw rate with an instantaneous center of rotation very far away from the target. It is also noteworthy that in the simulation presented section 3, the path vehicles follows are computed by splines between three, or more, points. Thus the target never travels in a pure linear motion unless its trajectory is defined by exactly two points.

Further, the velocity component that the radar is measuring can be described by

$$\dot{r}_{i,j,comp} = \begin{bmatrix} \cos\left(\theta_{i,j}^{[C]}\right) & \sin\left(\theta_{i,j}^{[C]}\right) \end{bmatrix} \begin{bmatrix} v_{x,i,j,P}^{[C]} \\ v_{y,i,j,P}^{[C]} \end{bmatrix} \quad (4.11)$$

By combining Equation 4.10 and 4.11, we can formulate

$$\dot{r}_{i,j,comp} = \underbrace{\dot{\phi}\left(y_{ICR}^{[C]} - y_j^{[C]}\right)}_{F_j} \cos\left(\theta_{i,j}^{[C]}\right) + \underbrace{\dot{\phi}\Big(\overbrace{x_j^{[C]}}^{=0} - x_{ICR}^{[C]}\Big)}_{G} \sin\left(\theta_{i,j}^{[C]}\right) \quad (4.12)$$

i.e. the equation describing the velocity profile. Note that the velocity profile is a sinusoidal which amplitude and phase is determined by F_j and G, as shown in 4.5.

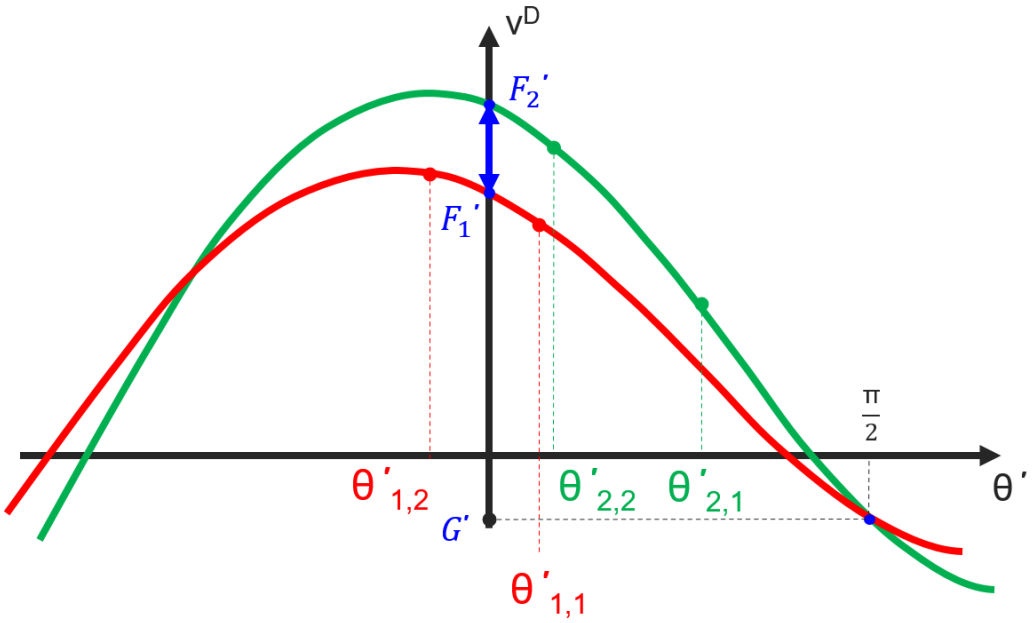


Figure 4.5: The detections obtained from the target when plotted over the azimuth on the x-axis and the range-rate on the y-axis. The detections obtained from each sensor determine unique sinusoidal velocity profiles defined by the parameters F_i and G.

4.1.2.3 Exploiting the velocity profile

By using the equation 4.12, we can form a system of linear equations to solve for each unknown variable. However, the system is clearly over-determined for more than 3 detections on the same unit. Hence to utilize additional detections, we make use of two least squares estimators. The first solves for F_j and G through

$$\min_{F_j,G} \sum_j \sum_i \left(r_{i,j,comp} - F_j \cos(\theta_{i,j}) - G \sin(\theta_{i,j})\right)^2 \tag{4.13}$$

Here we can also observe the reason we gain the benefits from using the common frame. As the two sensors' Y-axis are aligned, $G = G_j = G_1 = G_2$ which allows us to make a single fit in the least squares estimator. The second solves for the full motion through

$$\min_{\dot{\phi}^{-1},x_{ICR}^{[C]},y_{ICR}^{[C]}} \left\| \begin{bmatrix} 0 \\ 0 \\ y_{S_2}^{[C]} \\ 0 \end{bmatrix} - \begin{bmatrix} -F_1 & 0 & 1 \\ G & 1 & 0 \\ -F_2 & 0 & 1 \\ G & 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\phi}^{-1} \\ x_{ICR}^{[C]} \\ y_{ICR}^{[C]} \end{bmatrix} \right\|_2^2 . \tag{4.14}$$

However, using all radar measurements in a least squares estimator makes the method prone to outliers. A single outlier may skew the result significantly. Therefore we utilize RANSAC with the aim to detect any outliers to the velocity profile. To do such, in every RANSAC iteration, we select 3 random radar measurements, using at least 1 from each sensor and compute the exact fit to F_j and G in the

velocity profile as described in equation 4.12. Then, using only measurement that were classified as inliers, deploy the least squares estimators as described above.

Using the aforementioned method, $\dot{\phi}$ is solved for directly. To estimate the velocity vector of any point on the target we combine the estimated yaw-rate with the estimated ICR point. As our state vector models velocity of the target as the speed at the rear-axle of each unit, we estimate the state through

$$v = \left\| \dot{\phi} \sqrt{\left(\begin{bmatrix} x_{ICR}^{[W]} \\ y_{ICR}^{[W]} \end{bmatrix} - \begin{bmatrix} x_{rear-axle}^{[W]} \\ y_{rear-axle}^{[W]} \end{bmatrix} \right)^2} \right\|. \tag{4.15}$$

4.1.3 Estimating articulation angle and angle rate

When the individual states of both units have been estimated, both of them can be used to determine the linkage between them, i.e the state of the articulation angle. The articulation angle is computed by subtracting the tractor yaw from the trailer yaw as

$$\phi_A = \phi_1 - \phi_2 \tag{4.16}$$

and similarly, the articulation angle rate as

$$\dot{\phi}_A = \dot{\phi}_1 - \dot{\phi}_2. \tag{4.17}$$

4.2 Clustering

To partition radar detections into two groups, detections originating from the tractor and detections originating from the trailer, we utilize K-means clustering. K-means tries to create clusters with the least within-cluster squared euclidean distances, i.e., it aims to minimize the total euclidean distance between all data points within each cluster [17]. Hence to partition the radar detections appropriately, we can feed the clustering algorithm data that is based on radar detections and is distributed such that the resulting clusters would correspond to the tractor and trailer respectively. As an articulated vehicle consists of two units, we can exploit how the radar measurement are spatially distributed. If the target vehicle consists of two units of the same length and the radar measurements are distributed equally along the long side of the vehicle, we would be able to split the measurements in two spatially. The clustering with the smallest total euclidean distance between all detections is to partition as shown in Figure 4.6.

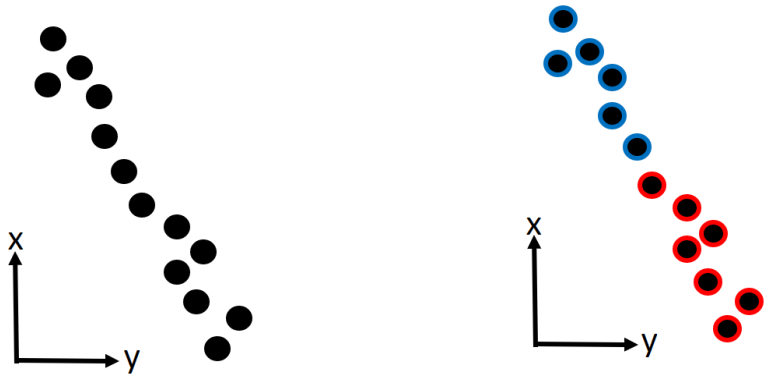


Figure 4.6: A set of samples partitioned such that it has 2 clusters with the smallest total euclidean distance between all samples.

Ultimately the K-means clustering algorithm is set to create two clusters using the cartesian position of the radar detections. I.e.

$$clusters = kmeans\left(\begin{bmatrix}\boldsymbol{x}, & \boldsymbol{y}\end{bmatrix}, \ 2\right) \tag{4.18}$$

such that, ideally, the clustered detections will be partitioned such as shown in Figure 4.7.

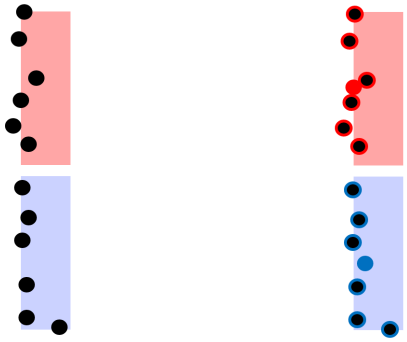


Figure 4.7: The detections before clustering are shown to the left and the desired outcome of clustering shown to the right. The radar detections (black dots) are partitioned (red/blue outline) such it results in 2 clusters where each originate from a single vehicle unit. The mean position of each cluster is shown as a solid red/blue dot.

4.2.1 Cluster labeling

The K-means clustering algorithm is on its own only able to split radar detections into two, unlabeled partitions. In order to be able to take advantage of the clustering, we need to determine which partition corresponds to which unit of the articulated vehicle. To do such, we introduce Algorithm 4 that relies on the assumption that the target vehicle is always moving forward and that the tractor will always be the unit that is facing the direction of travel of the target vehicle. It will choose a cluster,

e.g. cluster A, and compute the dot product between the relative vector of the two clusters, $P_{\Delta AB}$, and the cluster's velocity vector, V_A. The cluster's velocity can be computed from the angular velocity and the instantaneous centre of rotation of the cluster, as seen in Figure 4.8.

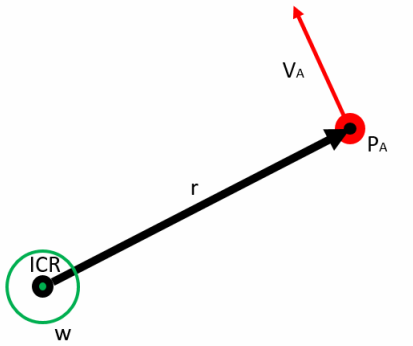


Figure 4.8: The cluster's velocity vector V_A can be computed from the vectors **r** (i.e. the vector between the cluster's ICR point and the cluster's centroid P_A) and the angular velocity vector ω.

If the dot product $V_A \cdot P_{\Delta AB}$ is positive, i.e. the two vectors are oriented such that there are less than 90 degrees separation between them, cluster A must be the trailer. This is shown graphically in Figure 4.9 and the algorithm is introduced in detail in Algorithm 4.

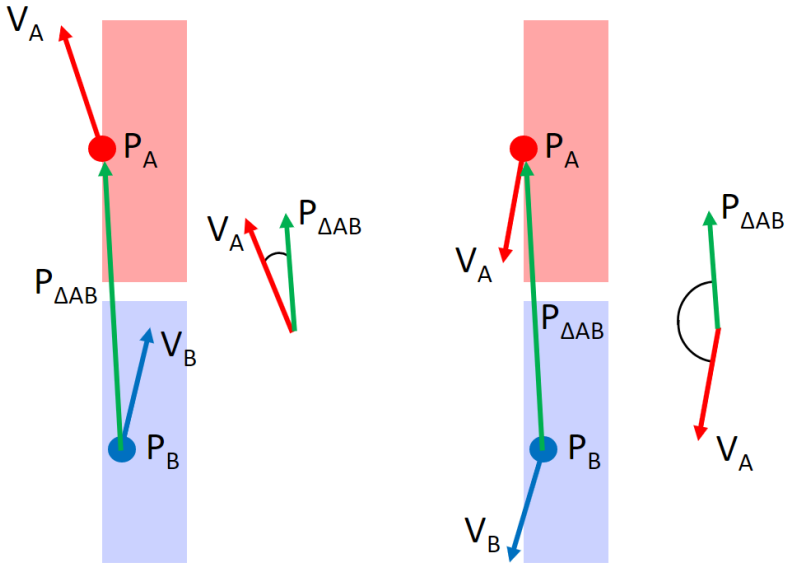


Figure 4.9: Showcase of how the two vectors, v_A, and $P_{\Delta AB}$, are computed.

Algorithm 4: Algorithm used to label clusters

Data: 2 unlabeled clusters $clust_A$ and $clust_B$ of radar detections
Result: Cluster labels
$P_A^{[W]}$ = mean position of $clust_A$;
$P_B^{[W]}$ = mean position of $clust_B$;
$P_{\Delta AB}^{[W]} = P_B^{[W]}$ - $P_A^{[W]}$;
$[\dot{\phi}_A, P_{ICR,A}^{[W]}]$ = EstimateYawrate($clust_A$);
$V_A^{[W]} = \begin{bmatrix} 0 \\ 0 \\ \dot{\phi}_A \end{bmatrix} \times \begin{bmatrix} P_{x,A}^{[W]} - P_{x,ICR,A}^{[W]} \\ P_{y,A}^{[W]} - P_{y,ICR,A}^{[W]} \\ 0 \end{bmatrix}$ From the angular velocity ($\omega \times \mathbf{r}$);
$scalar\ product = P_{\Delta AB}^{[W]} \cdot V_A^{[W]}(1:2)$;
if *scalar product > 0* **then**
    $clust_B$ is the tractor;
    $clust_A$ is the trailer;
else
    $clust_A$ is the tractor;
    $clust_B$ is the trailer;

Although not shown in Algorithm 4, there is a process to determine which of the two clusters is most the most suited to compute the velocity vector of. This process is determined by how many sensors is seeing each cluster and how many detections each cluster consists of with the more, the better. If cluster B were to be chosen instead of A, cluster B is simply switched with cluster A in Algorithm 4.

4.2.2 Increasing clustering robustness

There are scenarios where the aforementioned approach with clustering works poorly, e.g. when there is an imbalance in how the detections are distributed between the tractor and trailer. This happens when one of the two units of an articulated vehicle is seen from two sides and the other unit is seen from a single side as shown to the left in Figure 4.10. The imbalance will cause a shift in the mean position of the clusters and, as a consequence, some detections will be miss-labeled. To minimize the number of detections that are miss-labeled, the measurements that are at the most risk are discarded, i.e. measurements that are close to the midpoint between the clusters, as shown in Figure 4.10.

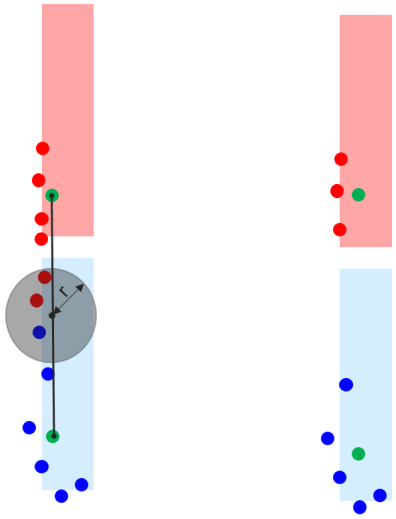


Figure 4.10: The detections can be misclassified as shown to the left, where some of the red detections belong to the other unit. Thus, detections are removed in the circled area giving the detections on the right.

4.2.3 Scenarios where clustering is not possible

A drawback with using K-means clustering is that the algorithm *always* returns two clusters. However, in many traffic scenarios, the articulated vehicle is oriented in a such a way that only a single unit is visible from the perspective of the ego vehicle. For example, when the ego vehicle is driving behind the target vehicle, the detections will typically only originate from the trailer as seen in Figure 4.11.

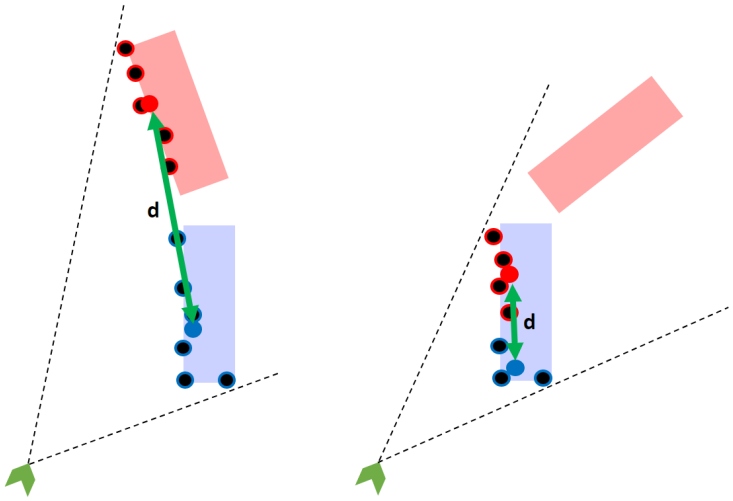


Figure 4.11: A scenario where clustering is possible to the left and a scenario where clustering is not possible to the right. The perspective of the ego vehicle is represented by the green arrowhead and the black dotted lines at the bottom left in the two scenarios.

Another situation is when the vehicle is jack-knifed and one of the units is completely

obscured, as seen to the right. Clustering in these instances would be equivalent to miss-label half of all measurements hence it is of importance to detect when it happens. As this thesis assume the target vehicle dimensions to be known, we can use this knowledge to determine if the distance between the resulting clusters' centroid is reasonable or not.

To determine whether the clustering output should be used, a threshold for the minimum distance allowed between the cluster centroids, d_{min}, is set. If the distance is smaller than this threshold, the clustering output will be ignored in order to prevent a large number of misclassifications. In addition, a minimum amount of detections for each cluster is set to increase robustness against outliers.

Even though clustering is not always available, there is still information to gain in labeling the unit that is seen. To do so we rely on the assumption that the target vehicle is moving forward. This implies that if the unit seen is moving towards the ego vehicle, it must be the tractor that is in view. If it is moving away, then it must be the trailer. This information is used in 4.4.2 to manage the inseparable case as good as possible.

4.3 Kalman filter

It is important to note that the Kalman filter plays two important roles in the tracking algorithm. Firstly, it aids in minimizing the impact of noise in algorithm output. Secondly, it assists in extrapolating the algorithm output when state measurements are not available. Due to the dynamics of the pivot joint in articulated vehicles, at times, parts of the vehicle may be obscured by itself from certain perspectives as can be seen in 4.11. Hence the information the tracking algorithm is able to gather is dependent on the target vehicle state and the relative position of the ego vehicle. E.g. if the ego vehicle encounters a scenario such as seen in 4.11, the algorithm will only be able to measure of the trailer and not the vehicle as a whole. This is where the motion model of the Kalman filter enables the algorithm to still yield a reasonable output of the vehicle as a whole despite not being able to directly measure all its states.

4.3.1 State vector

The purpose of the state vector is to capture the important characteristics of the target using a minimal state representation. As motivated in section 2.3, the kinematics of the tractor and trailer are interlinked such that all states of the trailer can be expressed in terms of tractor states in combination with the articulation states.

Hence the state vector of choice is

$$\boldsymbol{x}^{[W]} = \begin{bmatrix} \boldsymbol{x}_1^{[W]} \\ \boldsymbol{x}_A \end{bmatrix} = \begin{bmatrix} x_1^{[W]} \\ y_1^{[W]} \\ v_{1r} \\ \phi_1^{[W]} \\ \dot{\phi} \\ \phi_A \\ \dot{\phi}_A \end{bmatrix}. \tag{4.19}$$

expressed in the world frame. Using the aforementioned state vector and the kinematic relationships introduced in section 2.3, the state of the trailer can still be completely determined even though the trailer states are not explicitly included yielding a minimal state representation.

4.3.2 Motion model

To predict the movement of the articulated vehicle, a modified Coordinated-Turn (CT) model including the tractor and articulation states is used. The CT model allows for uncertainties in the velocity, yawrate and articulation angle rate and is thus commonly used when tracking objects that are expected to turn. The motion model computes a prediction for the tractor and articulation states, which can be used to compute the remaining states of the trailer as mentioned earlier. The motion model $\boldsymbol{f}$ is given as

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{bmatrix} x_1 + T v_{1r} \cos(\phi) \\ y_1 + T v_{1r} \sin(\phi) \\ v_{1r} \\ \phi + T\dot{\phi} \\ \dot{\phi} \\ \phi_A \\ \dot{\phi}_A \end{bmatrix} \tag{4.20}$$

where the process noise matrix $\boldsymbol{Q}$ is given as

$$\Gamma = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{Q} = \Gamma \begin{bmatrix} \sigma_v^2 & 0 & 0 \\ 0 & \sigma_{\dot{\phi}}^2 & 0 \\ 0 & 0 & \sigma_{\dot{\phi}_A}^2 \end{bmatrix} \Gamma^T. \tag{4.21}$$

The Jacobian $\boldsymbol{F}$ can be computed by

$$\boldsymbol{F} = \frac{d\boldsymbol{f}(\boldsymbol{x})}{d\boldsymbol{x}} = \begin{bmatrix} 1 & 0 & T\cos(\phi) & -Tv_{1r}\sin(\phi) & 0 & 0 & 0 \\ 0 & 1 & T\sin(\phi) & Tv_{1r}\cos(\phi) & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & T & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.22}$$

4.3.3 Measurement model

This thesis utilize various methods that are able to compute the tractor, the vehicle states, and the articulation states separately. However, all measurements may not always be available. Therefore, three separate measurement models are used, one for each of the three state vectors. The tractor measurement model h_1 will consist of direct measurements and is hence given by

$$h_1\left(\hat{\boldsymbol{x}}^{[W]}\right) = \begin{bmatrix} \hat{x}_1^{[W]} \\ \hat{y}_1^{[W]} \\ \hat{v}_{1r} \\ \hat{\phi}_1^{[W]} \\ \hat{\dot{\phi}} \end{bmatrix} \tag{4.23}$$

and has the corresponding measurement model Jacobian is given by

$$H_1\left(\hat{\boldsymbol{x}}^{[W]}\right) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}. \tag{4.24}$$

Similarly, the articulation measurement model h_A is given by

$$h_A\left(\hat{\boldsymbol{x}}^{[W]}\right) = \begin{bmatrix} \hat{\phi}_A \\ \hat{\dot{\phi}}_A \end{bmatrix} \tag{4.25}$$

and the articulation measurement model Jacobian is given by

$$H_A\left(\hat{\boldsymbol{x}}^{[W]}\right) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.26}$$

The trailer measurement model is given by

$$h_2(\hat{\boldsymbol{x}}^{[W]}) = ... = \begin{bmatrix} \hat{x}_2^{[W]} \\ \hat{y}_2^{[W]} \\ \hat{v}_{2r} \\ \hat{\phi}_2^{[W]} \\ \hat{\dot{\phi}}_2 \end{bmatrix} \tag{4.27}$$

which is derived using the relationship between the tractor and trailer as introduced in section 2.3. The measurement model Jacobian, $H_2(\hat{\boldsymbol{x}}^{[W]})$, is derived using Matlab and its symbolic toolbox.

4.3.4 Filter initialization

For the purposes of this thesis, to initialize the filter, the filter is fed the ground truth state of the target vehicle with zero uncertainty. I.e.

$$\hat{\boldsymbol{x}}_{t_0}^{[W]} = \begin{bmatrix} x_{1,t_0}^{[W]} \\ y_{1,t_0}^{[W]} \\ v_{1r,t_0} \\ \phi_{1,t_0}^{[W]} \\ \dot{\phi}_{1,t_0} \\ \phi_{A,t_0} \\ \dot{\phi}_{A,t_0} \end{bmatrix}, \tag{4.28}$$

$$P_{t_0} = \mathbf{0}_{7x7} \tag{4.29}$$

4.4 Tracking algorithm operation

In this section the algorithm at run-time is explained in detail. The exact operation differs somewhat from the simplified overview presented in Figure 4.1 depending on how the target is seen from the perspective of the ego vehicle. If both the tractor and the trailer of the target is seen, similarly to what can be seen to the left in Figure 4.11, then the algorithm will enter what will be referred to as its *ideal* operational state. If only a single unit of the target is seen, similarly to what can be seen to the right in Figure 4.11, then the algorithm will enter what will be referred to as its *alternate* operation state. Ultimately, the main difference between the different states is what Kalman updates steps are performed.

4.4.1 Ideal operation

In the ideal case, the measurements originate from both units of the target vehicle. Now, the ordinary state estimation techniques in 4.1 can be used on each cluster individually to estimate the states of the trailer and tractor respectively. The articulation states are then computed by using the relationships presented in 4.1.3. All 3 are then used to make three separate Kalman update steps, one for each state vector.

4.4.2 Alternate operation

In the alternate case, only a single unit is seen from the perspective from the ego vehicle and thus only information regarding one unit can be derived. The trigger for this operational state is when the algorithm is unable to cluster detections as

presented in section 4.2.3. This section will thus detail how the algorithm make use of the information at hand.

When all detections originate from a single unit as seen in Figure 4.12, the clustering algorithm will flag that the mean position of the two clusters are unexpectedly close, similarly to what can be seen to the right in Figure 4.11.

Figure 4.12: Showcase of the scenario where all radar detections (blue dots) originate from a single unit of the target vehicle.

While the clustering algorithm is unable to partition the detections, it still attempts to label the detections at hand by assuming the target vehicle is always moving forward. Thus, if one unit is obscured and the average velocity of the detections is e.g. heading towards the ego vehicle, the detections will all be labeled as originating from the tractor. Then these will be fed as one to the state estimation techniques which in turn will be used in an appropriate Kalman update step. Note that this means that only a single Kalman update step will be performed.

There are however cases where both units can be seen from the perspective of the ego vehicle and the clustering algorithm is still unable to partition the detections. This happens when the number of detections in one of the clusters is too low to be able to be used in the state estimation techniques as can be seen in Figure 4.13. E.g. at least 3 detections are needed in order to fit a bounding box.

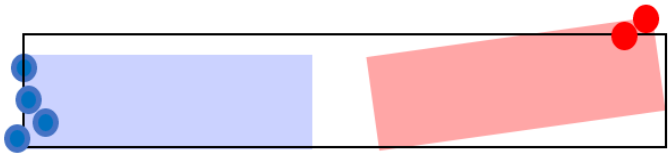


Figure 4.13: Showcase of the scenario where the algorithm is unable to cluster while both units is still visible to the ego vehicle sensors.

In this scenario, the algorithm will utilize the same workflow as when one of the units are completely obscured to derive information in regards to the visible unit. The addition made in this scenario is that the algorithm makes an educated guess that the tractor and trailer is close to parallel. Thus the articulation angle is about 0 degrees. This will then be fed to the Kalman filter with a slight increased measurement uncertainty. As no information in regards to the articulation angle rate can be derived, it will also be fed as 0 to the Kalman filter however with a large measurement uncertainty.

5
Algorithm performance

This section aims to quantify the performance of the articulated vehicle tracking algorithm presented in the thesis. Due to the high number of possible scenarios that can occur, quantifying performance is a long and complicated process. In addition, the tracking algorithm has parameters that need tuning and often the ideal set of parameters varies in different scenarios. As this thesis utilizes a simulation environment, a scenario can easily be re-simulated many times using various settings. However, any movement of vehicles has to be manually specified. Hence each scenario effectively still has to be created by hand. This thesis unfortunately does not have the resources to create such a complete set of scenarios nor pursue a thorough tuning process. Instead, a small set of scenarios will be selected with the aim to highlight some of the algorithm's strengths and weaknesses with algorithm parameters chosen through reasonable guesses.
This section is structured such that it first introduces the evaluation methods, i.e. how the performance is measured. Then the scenarios are presented and ultimately the results are shown.

5.1 Evaluation methods

The algorithm will be evaluated by simulating a set of scenarios and each scenario will be simulated 20 times. Data will be aggregated and analyzed for each of the 20. However, 1 simulation will be selected randomly to show the behavior of the algorithm in operation rather than through aggregated data. The components up for evaluation are split into clustering & labeling, and state estimation.

5.1.1 Evaluation of clustering & labeling

The algorithm's ability to correctly cluster and label detections is measured by comparing the estimated origin of each radar detection to the true origin. As an error in clustering and labeling can affect processes downstream in the algorithm, the result is shown per time step in addition to aggregated data to be able to correlate between errors.

5.1.2 Evaluation of state estimation

The algorithm's ability to estimate the state of the target is analyzed by computing the error by subtracting the ground truth from from the estimated states. As the

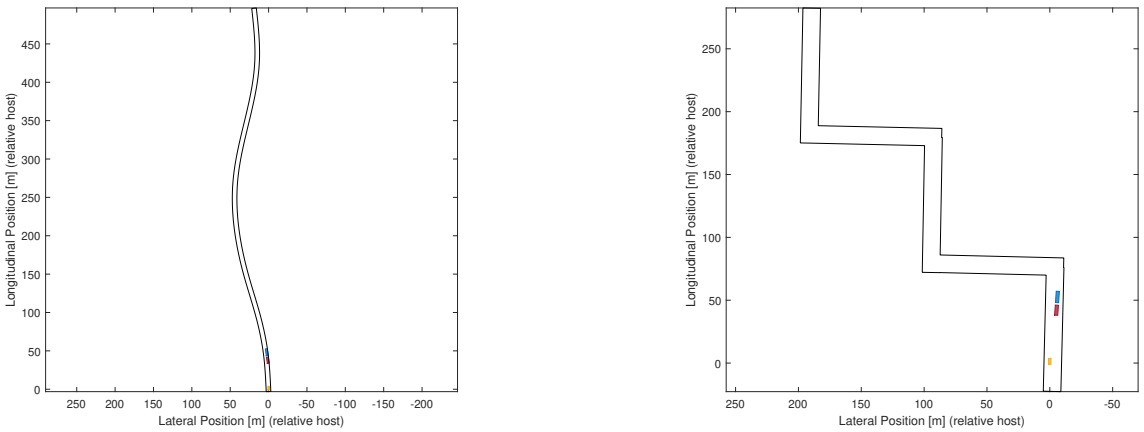
process the algorithm estimates the target's states is situational, the state estimation error is split into three main categories. Clustered, unclustered, and filtered state errors. Clustered and unclustered state errors are thus a measure of instantaneous state estimations whereas the filtered state estimation refers to the Kalman filtered state errors. Each of the three is further split into tractor and trailer errors as applicable.

5.2 Scenarios

To evaluate the algorithm's strengths and weaknesses, two different scenarios were created. Each of which are meant to highlight specific properties of the algorithm. These are

- **Scenario 1**
  - the ego vehicle follows an articulated vehicle down a highway
- **Scenario 2**
  - the ego vehicle follows an articulated vehicle in repeated turns

which are shown in Figure 5.1.



(a) Scenario 1. **(b)** Scenario 2.

Figure 5.1: Two different simulation scenarios.

5.2.1 Scenario 1

This scenario is meant to imitate typical highway driving. The ego vehicle is following the target vehicle in the same lane at high speeds moving relatively straight. Because of this, most of the time, the tractor is obscured from the perspective of the ego vehicle. However, as the target vehicle is modeled as an articulated vehicle, the algorithm still keeps track of the position of the tractor.

5.2.2 Scenario 2

This scenario is meant to imitate typical sharp turns. The ego vehicle is trailing behind the target vehicle as consecutive turns are executed. Again, the tractor will

be mostly obscured from the ego vehicle's perspective and only appear when the target enters a turn. However, as the target vehicle is modeled as an articulated vehicle, the algorithm still keeps track of the position of the tractor.

5.3 Results

The data from the simulations is presented on a scenario basis in the following sections. Note that even though the trailer state vector is not part of the Kalman filter state vector, it is still presented in the results as one. This is extrapolated using the relationships introduced in section 2.3 in combination with the tractor and articulation state vectors of the Kalman filter state vector.

5.3.1 Scenario 1 - State estimation

To first present an overview of the scenario, Figure 5.2 presents the positional ground truth as well as algorithm estimated positions from the randomly chosen simulation out of the total 20.

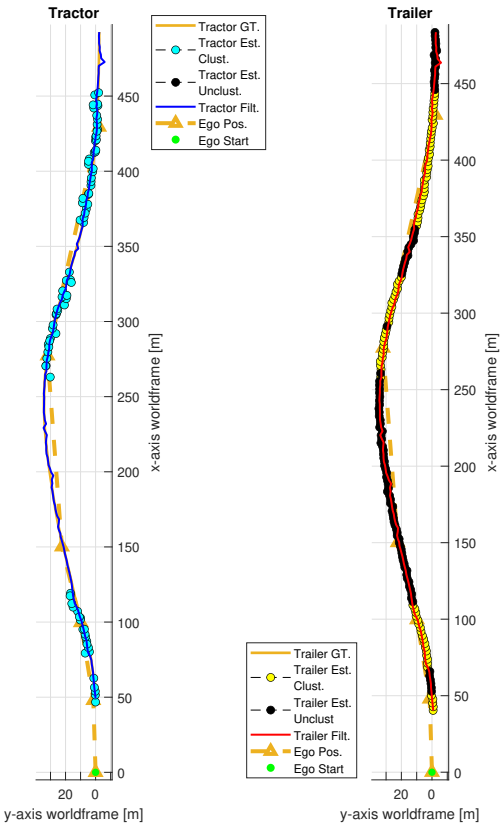


Figure 5.2: The trajectories for the tractor and trailer in the highway scenario. The figure shows the measured and filtered positions as well as the ground truth. The color of the filled dots indicate whether the target vehicle is clustered or not. Note that the ego vehicle is following the same trajectory as the target vehicle at a distance.

The plots shown in Figure 5.3 presents the remaining state vectors for the randomly chosen scenario.

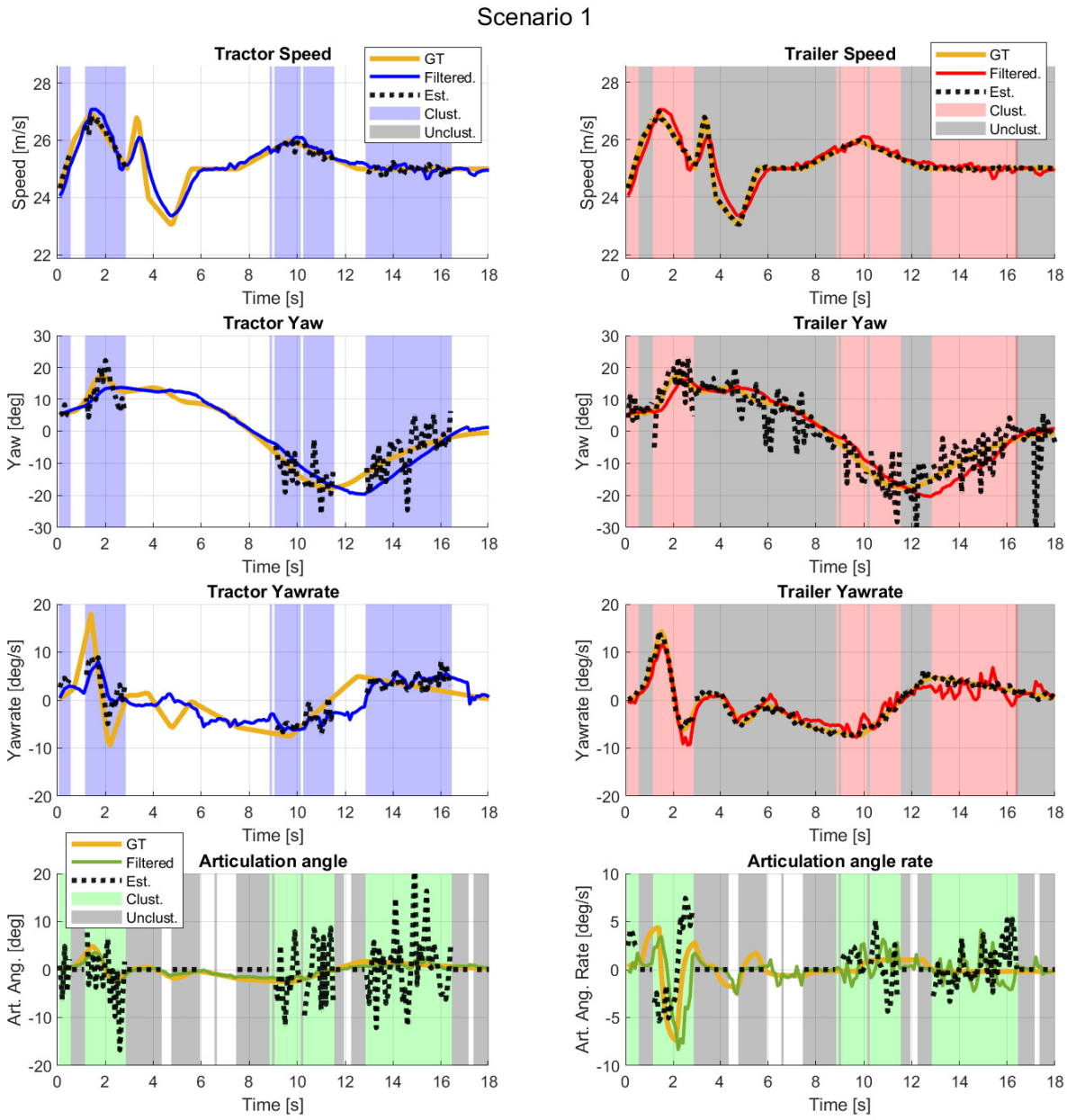


Figure 5.3: An overview of the estimated and filtered kinematic states of the target vehicle versus the ground truth. The background color indicates whether the clustering is turned on or off. A white background means that the unit is obscured and only receives at most a few detections.

The distribution of the estimated state errors from all 20 simulations are shown in the histograms in Figure 5.4.

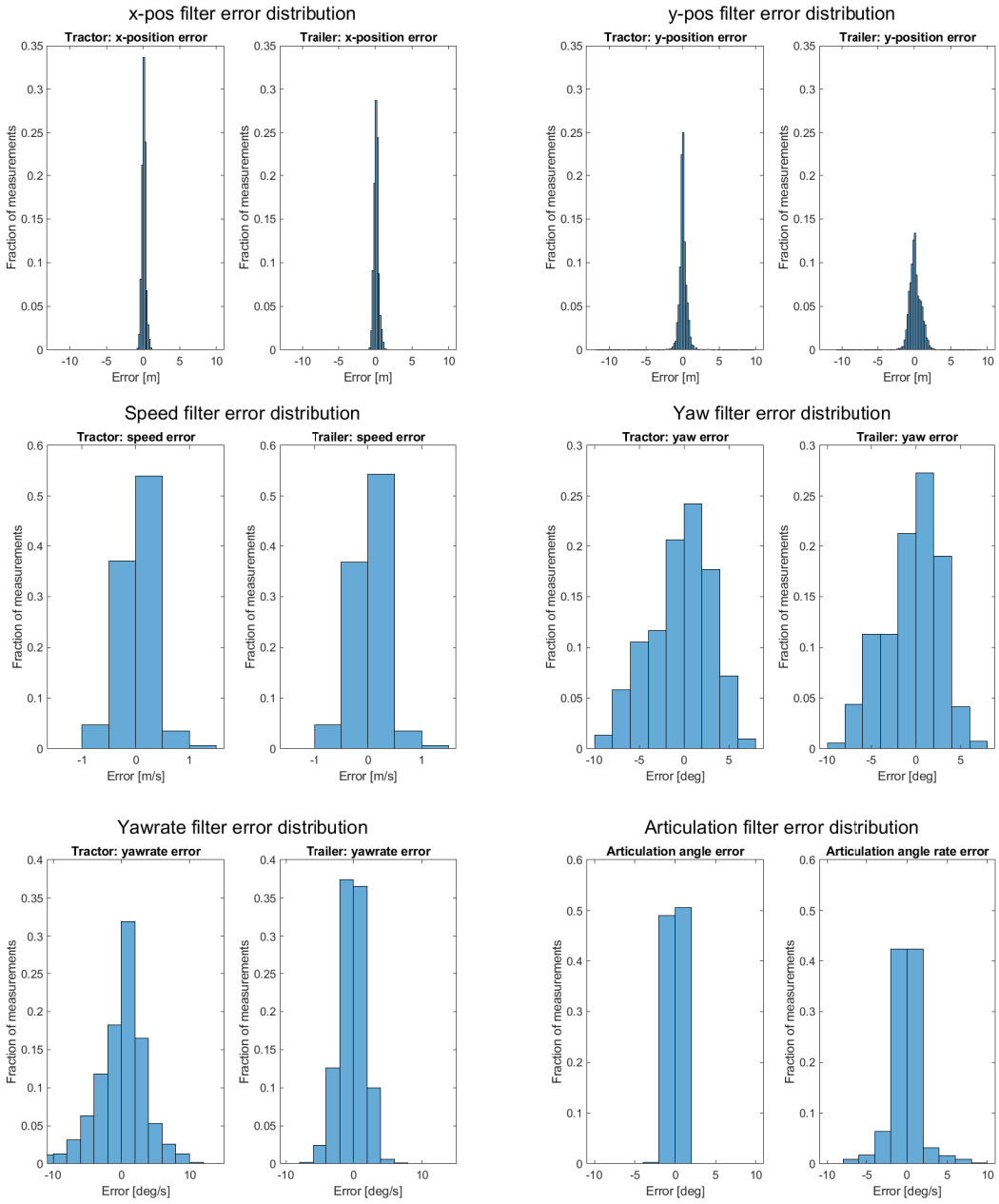


Figure 5.4: Histograms over the filtered state errors from scenario 1 after 20 simulations.

The aggregated estimated state errors from all 20 simulations expressed in terms of mean error μ and standard deviation σ is shown in Table 5.1.

Tractor states error (all iterations)										
	x [m]		y [m]		v [m/s]		ϕ [deg]		$\dot{\phi}$ [deg/s]	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
Filtered	0.22	0.27	0.36	0.66	0.20	0.29	2.71	3.36	2.67	3.56
Est. Clust.	1.51	1.36	0.98	1.10	0.10	0.09	3.49	3.09	2.36	2.03
Est. Unclust.	9.06	0.40	7.71	0.65	0.32	0.08	85.59	2.73	1.45	0.75

Trailer states error (all iterations)										
	x [m]		y [m]		v [m/s]		ϕ [deg]		$\dot{\phi}$ [deg/s]	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
Filtered	0.26	0.31	0.61	0.86	0.20	0.29	2.48	3.08	1.37	1.79
Est. Clust.	0.17	0.13	0.23	0.19	0.03	0.03	3.97	3.35	0.53	0.44
Est. Unclust.	0.12	0.15	0.32	0.36	0.02	0.03	3.13	5.72	0.58	0.47

Articulation states error (all iterations)				
	ϕ [deg]		$\dot{\phi}$ [deg/s]	
	μ	σ	μ	σ
Filtered	0.50	0.63	1.12	1.78
Est. Clust.	5.40	4.48	2.34	2.02
Est. Unclust.	1.03	0.79	0.77	0.80

Table 5.1: Monte Carlo mean μ and standard deviation σ errors for 20 simulations. The data is split between different types of measurements, the filtered states error, the measured states error when clustering is used, and the measured states error when clustering is not available.

5.3.2 Scenario 1 - Clustering & labeling

In Figure 5.5, the number of correctly classified detections in each time step is shown for the randomly chosen simulation.

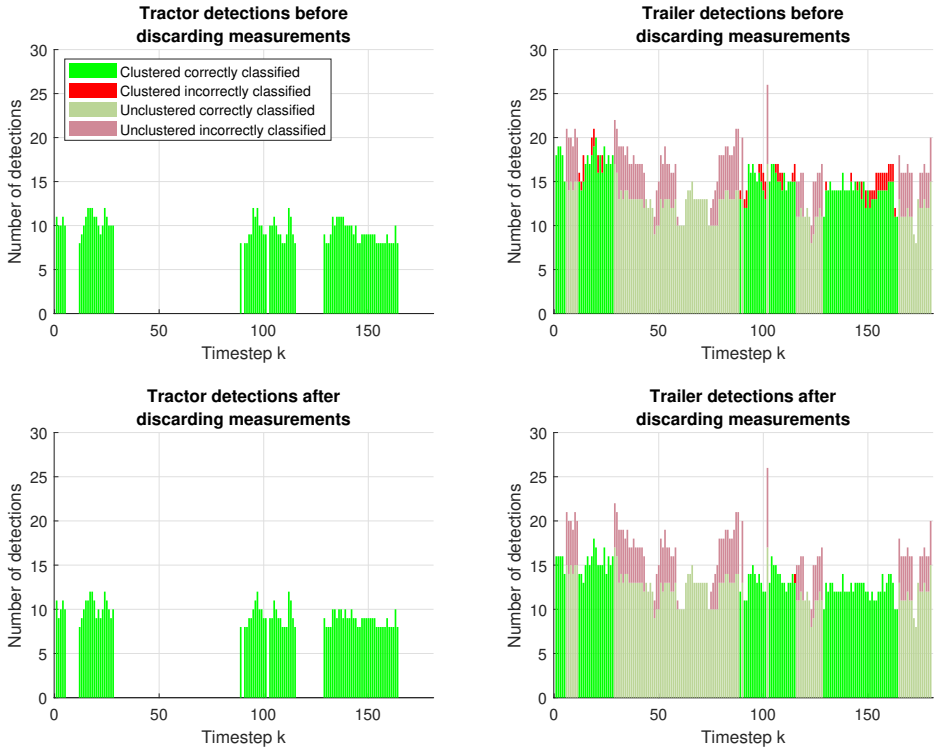


Figure 5.5: An illustration of the clustering performance. In the highway scenario, the target tractor unit is not always seen hence the sparsity in the left plots. In the right plots, the effect of removing detections between the units can be seen.

In Table 5.2, the percentage of correctly classified detections in all 20 simulations combined are shown.

	Figure 5.5 in table data for all iterations (after measurements have been discarded)	
	Timesteps clustering is possible	Timesteps clustering is not possible
	45.3 % (1630 / 3600)	54.7 % (1970 / 3600)
	Detections correctly classified	Detections correctly classified
Tractor	100.0 % (15241/15241)	n/a
Trailer	99.9 % (21846/21860)	74.3% (31468/42360)

Table 5.2: Clustering data for highway scenario

5.3.3 Scenario 2 - State estimation

To first present an overview of the scenario, Figure 5.6 presents the positional ground truth as well as algorithm estimated positions from the randomly chosen simulation out of the total 20.

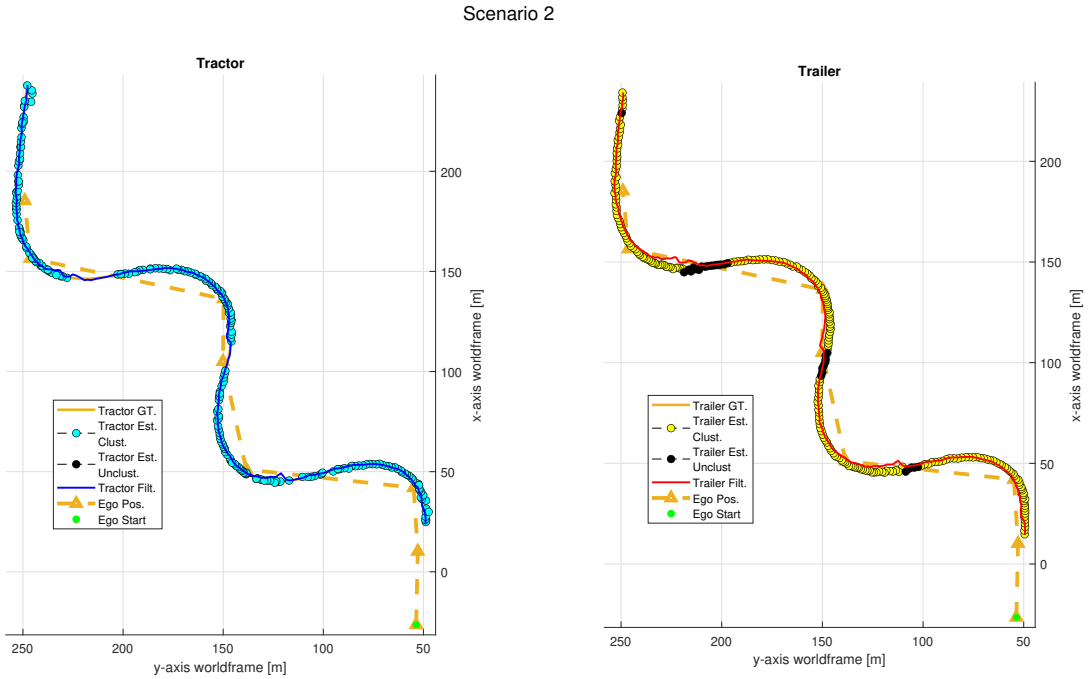


Figure 5.6: The trajectories for the tractor and trailer in the repeated turns scenario. The figure shows the measured and filtered positions as well as the ground truth. The filled dots indicate whether the target vehicle is clustered or not. In this scenario, the target vehicle is tracked as two units most of the time, with only small segments of unclustered tracking. Note that the ego vehicle trajectory is only a rough indication and that the actual movement is smooth.

The plots shown in Figure 5.7 presents the remaining state vectors for the randomly chosen scenario.

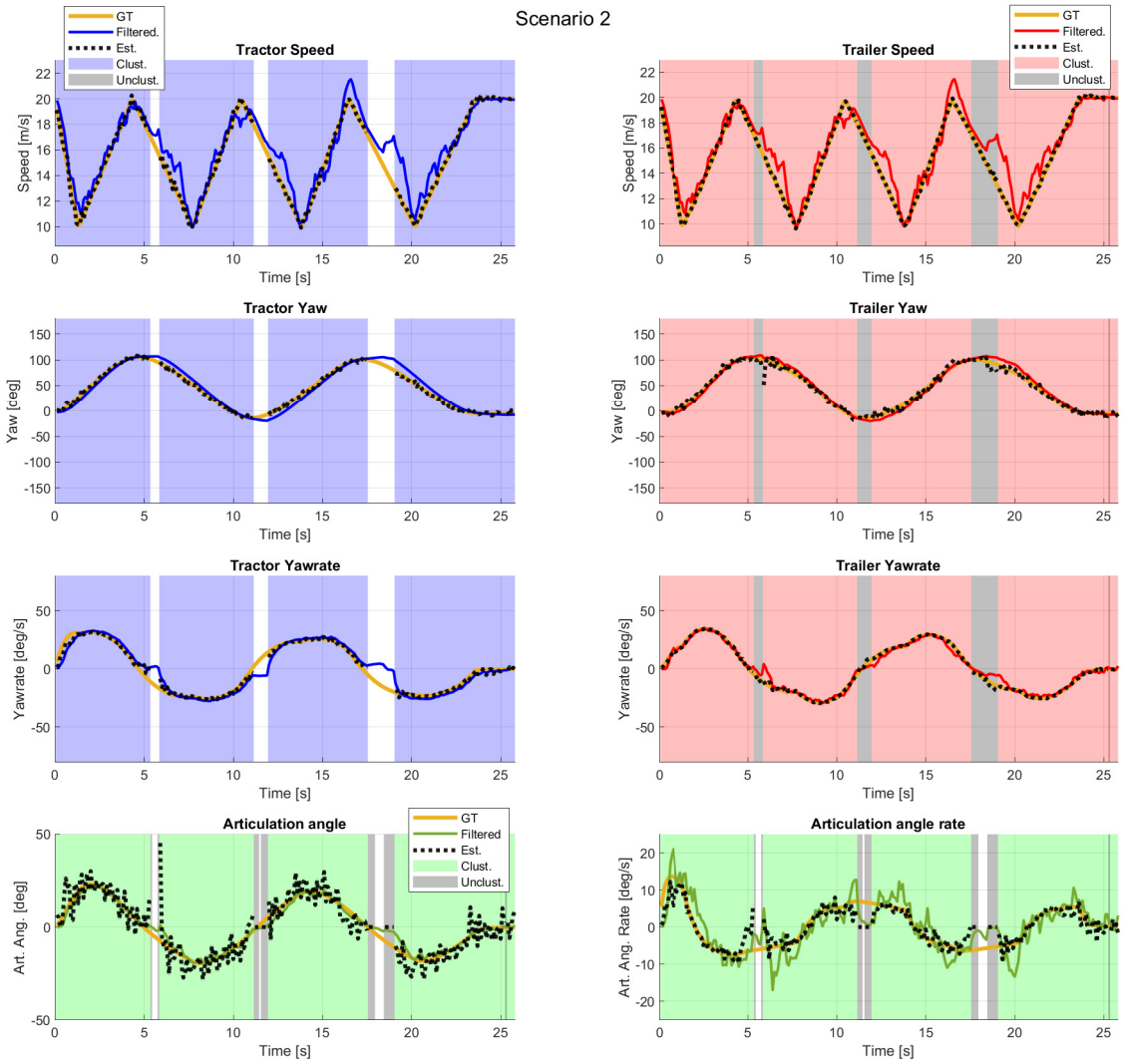


Figure 5.7: An overview of the estimated and filtered kinematic states of the target vehicle versus the ground truth. The background color indicates whether the clustering is turned on or off. A white background means that the unit is obscured and only receives at most a few detections.

The distribution of the estimated state errors from all 20 simulations are shown in the histograms in Figure 5.8.

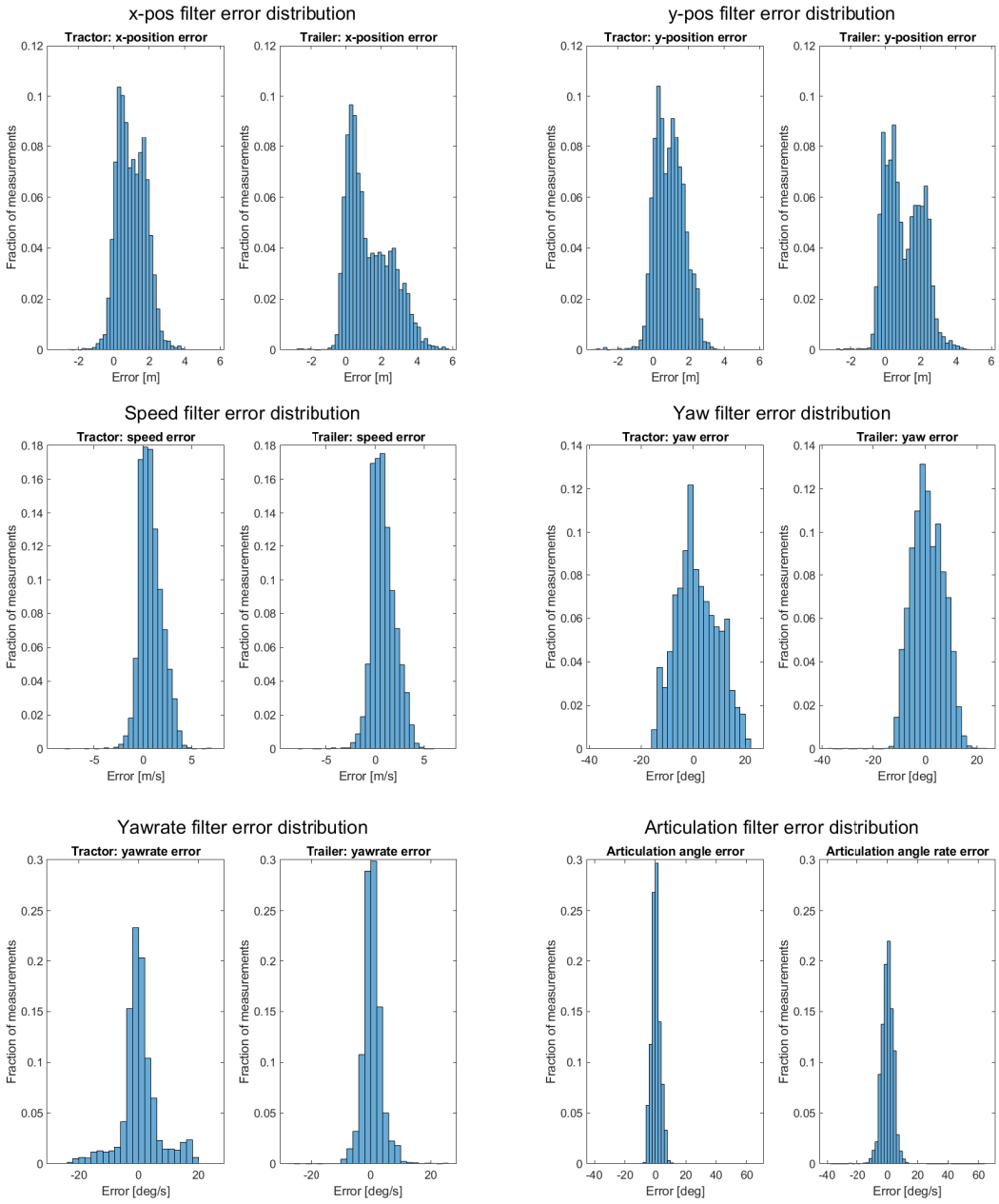


Figure 5.8: Histograms over the filtered state errors from scenario 2 after 20 simulations.

The aggregated estimated state errors from all 20 simulations expressed in terms of mean error μ and standard deviation σ is shown in Table 5.3.

Tractor states error (all iterations)										
	x [m]		y [m]		v [m/s]		ϕ [deg]		$\dot{\phi}$ [deg/s]	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
Filtered	1.05	0.79	1.00	0.81	1.06	1.16	6.64	8.06	4.25	6.35
Est. Clust.	0.59	0.93	0.51	0.78	0.16	0.16	2.50	2.00	1.66	2.32
Est. Unclust*.	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan

Trailer states error (all iterations)										
	x [m]		y [m]		v [m/s]		ϕ [deg]		$\dot{\phi}$ [deg/s]	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
Filtered	1.38	1.24	1.16	1.03	1.11	1.20	4.93	5.97	2.25	3.17
Est. Clust.	0.18	0.27	0.18	0.19	0.08	0.12	3.89	5.96	0.52	0.51
Est. Unclust.	0.31	0.35	0.20	0.32	0.07	0.09	5.71	5.64	2.02	1.35

Articulation states error (all iterations)				
	ϕ [deg]		$\dot{\phi}$ [deg/s]	
	μ	σ	μ	σ
Filtered	2.17	2.91	3.05	4.10
Est. Clust.	4.64	5.08	1.68	2.23
Est. Unclust.	4.49	3.46	5.82	1.38

Table 5.3: Monte Carlo mean μ and standard deviation σ errors for 20 iterations. The data is split between different types of measurements. That is, the filtered states error, the measured states error when clustering is clustering is utilized, and the measured states error when clustering is not available. (*) Note that the algorithm never classifies the seen unit as tractor in this scenario and thus there are no measurements.

5.3.4 Scenario 2 - Clustering & labeling

In Figure 5.9, the number of correctly classified detections in each time step is shown for the randomly chosen simulation.

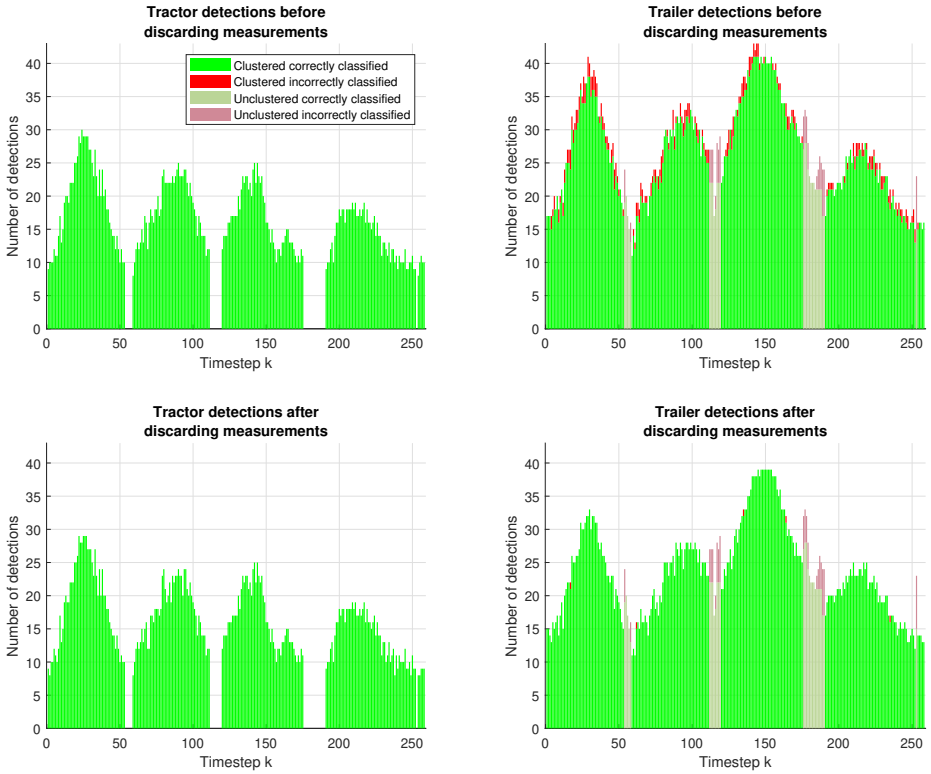


Figure 5.9: An illustration of the clustering performance. In the repeated turns scenario the clustering is on most of the time and the importance of removing measurements between the units can be seen to the right.

In Table 5.4, the percentage of correctly classified detections in all 20 simulations are shown.

	Figure 5.4 in table data for all iterations (after measurements have been discarded)	
	Timesteps clustering is possible	Timesteps clustering is not possible
	88.8 % (4582 / 5160)	11.2 % (578/5160)
	Detections correctly classified	Detections correctly classified
Tractor	100.0 % (74774/74774)	0 %
Trailer	99.9 % (109393/109453)	82.6% (13609/16481)

Table 5.4: Clustering data for repeated turns scenario

6
Discussion

As first introduced in Section 2.3, an articulated vehicle can be modeled as two units, the tractor and the trailer, which are linked through the articulation states. Each unit can be modeled individually using the state vector $(x, y, v, \phi, \dot{\phi})$ which is a common choice of state vector for vehicles in many non-articulated vehicle trackers. It is used in e.g. [18], [19], and a very similar variant in [4]. Hence estimation methods for these states are a well researched area, i.e. there are many methods to estimate and use the tractor and trailer states individually without any regards to their connection.
Therefore, the decision was made to to first build a non-articulated vehicle tracker that models vehicles using the $(x, y, v, \phi, \dot{\phi})$ vehicle state vector and estimates the corresponding states through existing techniques.
Hence, the contributions made by this thesis have mostly been made in regards to how to combine the state knowledge of the tractor and trailer to get a better estimate of an articulated vehicle as a whole. However, as a non-articulated vehicle tracker first had to be built, the thesis had to cover a lot of ground to get to its starting point. Because of this there are many areas that could easily be improved although left as-is simply due to lack of time. In addition, as the some of the foundation is built on components derived from non-articulated vehicle tracker algorithms, many areas that can be improved can be attributed to components that not necessarily need to be designed specifically for articulated vehicles.
Hence this chapter will mainly focus on the challenges of articulated vehicle tracking although some of the non-articulated vehicle tracking challenges will still be brought up as they are closely connected.

The following sections are structured such that, first the properties of the simulation are discussed, then the state estimation methods followed by the clustering performance and finished with a discussion focusing on the Kalman filter performance along with some overarching discussion.

6.1 Simulation

As mentioned in section 5, quantifying the performance of a vehicle tracking algorithm is non-trivial due to the high number of possible scenarios and, in turn, random parameters. Whilst this thesis utilize a simulation environment to simulate various scenarios, the environment is only able to generate random radar detection distributed according to set radar sensor parameters. Hence the movement of the

involved vehicles is the exact same between repeated runs of the same scenario although the radar detections may differ. The consequence of this is that even though e.g. scenario 1 was simulated 20 times over, the data is far from representative of all scenarios involving highway driving. Hence it is not possible to draw any definitive conclusion on the results presented from the simulations. Although what can be extrapolated is indications of the strengths and weaknesses of the articulated vehicle tracking algorithm.

6.2 Clustering

The need to partition radar detections based on the unit they are reflected of arose early to enable the state estimation of the tractor and trailer individually. Hence this challenge distills to very similar challenges as the s.c. data association problem, the problem of linking a radar detection to the vehicle it was reflected of, brings up. However, as the data association problem is outside the scope of this thesis such that it can be assumed that all radar detections originate from the articulated vehicle, the area of data association techniques was not thoroughly explored. Instead, more suited to the premise of the thesis, effort was instead focused on building a specialized solution that would exploit the properties of articulated vehicles to partition the radar detections.

The purpose of the k-means clustering techniques in this thesis is to exploit the positional distribution of radar detections on the articulated vehicle. As the shape and size is considered known, both can be used to make assumptions on how clusters will formed depending on the perspective the target is seen from. This can then be compared to the clusters returned by the k-means algorithm to determine if the radar detections have been partitioned correctly.

As presented in Table 5.2 and Table 5.4, nearly all detections are correctly classified when the clustering is possible. However, there are still many detections that gets incorrectly classified when clustering is not possible. This is especially visible in Table 5.2, where about every fifth detection is misclassified.

Although the approach shows very promising results, it is important to note that it may be an artifact created by the simulation environment. E.g. in a real world context, there are many properties that are likely detrimental to this approach that are unaccounted for in the simulation. The target vehicle may be obscured by other objects or the radar detections may not reflect as expected from all parts of the target vehicle to mention a few scenarios that would invalidate the approach of known cluster formation.

It is worth mentioning that the original idea to partition the radar detections was by exploiting the properties of the units' different velocity profiles during a turn manoeuvre. The velocity profile is a relatively robust measure as it can be estimated for rigid objects of arbitrary size. Due to the dynamics between the tractor and trailer, when initiating a turn, there will be a difference in yaw-rate between the

tractor and trailer, i.e. the velocity profiles would be different. This can then be exploited to separate the detections accordingly. However, the approach was eventually abandoned due two main challenges. First and foremost, the premise for this method is not present when the target is not turning. It could still be used as an addition during turn maneuvers although challenges with merging the result from k-means and this approach lead the idea to be abandoned completely.

6.3 State estimation methods

As the radar detections will either be partitioned into tractor and trailer detections or contain mostly detections from one of the units (i.e. when clustering is not possible), the state estimation methods can be designed very similarly to existing, non-articulated vehicle state estimation methods. However, as the tractor and trailer are closer together than other, separate, non-articulated vehicles, some particular edge-cases will arise for the state estimation methods.

As the strengths and weaknesses of the state estimation techniques have already been explored in the literature they were first introduced in, the discussion in the following sections will focus on the edge-cases introduced by articulated vehicles and areas of improvement rather than the techniques specific performance.

6.3.1 Position estimation

The positional measurements, i.e, the reference point and the yaw of the vehicle, rely heavily on the quality of the bounding box estimation. This estimation is vulnerable in certain orientations where the radar detection pattern does not clearly consist of an L-shape but rather an I-shape. This is unfortunately always the case for an articulated vehicle. If clustering is possible, the radar detections of one unit will form an L-shape most of the time whereas the radar detections of the other unit resembles the I-shape as can be seen to the left in Figure 6.1.

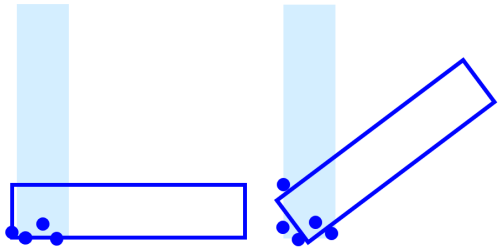


Figure 6.1: Two situations where the bounding box estimation is vulnerable.

These scenarios can cause the bounding box to take the wrong orientation. Much of this can be attributed to the fact that the bounding box estimation does not become more robust over time. Ideally, knowledge from previous time steps should be used to improve the estimate over time. In addition, the current bounding box algorithm does not utilize knowledge of the known tractor-trailer geometry, i.e, the bounding

boxes are fit independently of each other. Implementing additions that incorporate these two suggestions would likely mitigate the presented issues.
In addition, this estimation technique relies on having an appropriately tuned RANSAC threshold to correctly fit the bounding box. This thesis utilize a set threshold however the ideal threshold varies with the range to the target as the accuracy of radar detections decrease. A possible consequence of using an unsuitable threshold is simply a poor fit of the bounding box as shown to the right in Figure 6.1. Hence, a varying threshold is likely preferred over the current implementation.

6.3.2 Motion estimation

The full motion estimation algorithm used in this thesis is very suited for its use case when radar detections are partitioned into tractor and trailer. It is robust by design as it is independent of the target's shape and size, and it is able to handle outliers due to the use of RANSAC. The use of RANSAC is especially useful as outliers are the same as incorrectly classified radar detections in the eye of the motion estimation algorithm.

6.4 Kalman filter

The Kalman filter plays a fundamental role in the articulated vehicle tracking presented in this thesis. It is the link between the tractor, trailer and articulation states such that any of the three can be fed to the filter independently with the aim to create a better estimate of the articulated vehicle as a whole. However, there are many flaws in the use of the Kalman filter in the approach of this thesis that are detrimental to its performance.

The Kalman filter models the measurements obtained through the state estimation techniques (i.e. the input to the Kalman filter) as corrupted by Gaussian noise. The challenges is that while the raw radar detections can be modeled more accurately as affected by Gaussian noise, the state estimation methods heavily processes the radar detections in many, even non-deterministic (RANSAC), steps such that the noise is highly non-gaussian. To be able to yield an accurate output by efficiently compensate for the noise contained in the input, the Kalman filter need to precisely model how the Gaussian noise is processed. If the filter is performing nominally, the output should be close to zero mean Gaussian distributed. However, by observing the histograms of the Kalman filter output (see e.g. Figure 5.8), biases and non-Gaussian distributions are present. Thus the filter used in this thesis is not performing well. The Kalman filter is thus unable to model the noise in the input accurately and hence cannot compensate enough for such, giving biased estimates. This is especially an issue as the heavy processing of radar detections can introduce outlier state estimations. As an outlier does not fit the model the Kalman filter uses to compensate noise, an outlier is likely to throw the filter off completely. A consequence of the aforementioned flaws, the current implementation of the filter does not always produce a better filtered output than the measurements obtained through the state estimation.

It is also important to note that as the articulated vehicle state vector, consisting of tractor, trailer, and articulation states, is under-determined by knowing just one of the three state vectors. Hence in the scenario when only one unit is seen, the information that can be extracted is not enough to determine the state of the whole articulated vehicle. This is especially visible for the tractor yawrate in Figure 5.7 at around 6, 12, and 17 seconds in. At these times, it is only the trailer that is visible from the perspective of the ego vehicle sensors and the algorithm is thus unable to partition the radar detections. E.g. the yawrate of the trailer cannot alone determine the articulation anglerate *and* the tractor yawrate. The filter motion model can compensate for some time however, it is not enough for extended periods of time.

7
Conclusion

The purpose of the thesis was two-fold, to create a model for articulated vehicles suitable for tracking purposes and to estimate its states. A minimal state representation was derived to describe the state of an articulated vehicle which was used in an Extended Kalman Filter. The states of the tractor and trailer were estimated separately to gain full state knowledge of the articulated vehicle through the use of clustering and existing state estimation techniques.

It was hard to quantify the performance of the articulated vehicle tracking algorithm. In part due to limited test data, part due to challenges with how to define algorithm performance. The complexity of the algorithm was ultimately limited to fit the scope of the project. Thus trade-offs had to be made with some performance degradation as a result. However, the presented approach still demonstrates it is able to track some of the key kinematics of an articulated vehicle, indicating that the approach may be feasible for tracking purposes although further research is needed.

Bibliography

[1] K. E. G. Magnusson, "Segmentation and tracking of cells and particles in time-lapse microscopy," Ph.D. dissertation, KTH Royal Institute of Technology, Stockholm, 2016.

[2] M. E. B. E. N. Yassine Ruichek, Fadi Dornaika, "Sensors technologies and methods for perception systems in intelligent vehicles," *Journal of Sensors*, vol. 2016, no. 7241243, p. 1, 2016.

[3] R. Watson, *Radar Origins Worldwide: History of Its Evolution in 13 Nations Through World War II*. Trafford Publishing, 2009. [Online]. URL: https://books.google.se/books?id=Zup4V2wSZtMC

[4] K. Granström, M. Baum, and S. Reuter, "Extended object tracking: Introduction, overview, and applications," *Journal of Advances in Information Fusion*, vol. 12, 12 2017.

[5] F. Roos, D. Kellner, J. Klappstein, J. Dickmann, K. Dietmayer, K. D. Muller-Glaser, and C. Waldschmidt, "Estimation of the orientation of vehicles in high-resolution radar images," in *2015 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*, 2015, pp. 1–4.

[6] L. Hammarstrand, L. Svensson, F. Sandblom, and J. Sorstedt, "Extended object tracking using a radar resolution model," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 3, pp. 2371–2386, 2012.

[7] K. Olutomilayo and D. R. Fuhrmann, "Estimation of trailer-vehicle articulation angle using 2d point-cloud data," in *2019 IEEE Radar Conference (RadarConf)*, 2019, pp. 1–6.

[8] M. Parker, "Chapter 18 - radar basics," in *Digital Signal Processing 101 (Second Edition)*, 2nd ed., M. Parker, Ed. Newnes, 2017, pp. 231–240. [Online]. URL: https://www.sciencedirect.com/science/article/pii/B9780128114537000184

[9] C. Wolff. (2021) Continuous wave radar. [Online]. URL: https://www.radartutorial.eu/02.basics/Continuous%20Wave%20Radar.en.html

[10] W. Norris, *Modern steam road wagons*. London, New York, Bombay, Longmans, Green, and co., 1906, pp. 63–64. [Online]. URL: https://archive.org/details/modernsteamroadw00norrrich/page/62/mode/2up

[11] P. Lytrivis, G. Thomaidis, and A. Amditis, *Sensor Data Fusion in Automotive Applications*, 02 2009.

[12] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, p. 381–395, Jun. 1981. [Online]. URL: https://doi.org/10.1145/358669.358692

[13] (2021) Radar detection generator. [Online]. URL: https://se.mathworks.com/help/driving/ref/radardetectiongenerator.html

[14] H.-J. Li and Y.-W. Kiang, "10 - radar and inverse scattering," in *The Electrical Engineering Handbook*, W.-K. CHEN, Ed. Burlington: Academic Press, 2005, pp. 671–690. [Online]. URL: https://www.sciencedirect.com/science/article/pii/B9780121709600500475

[15] F. Roos, D. Kellner, J. Dickmann, and C. Waldschmidt, "Reliable orientation estimation of vehicles in high-resolution radar images," *IEEE Transactions on Microwave Theory and Techniques*, vol. 64, no. 9, pp. 2986–2993, 2016.

[16] D. Kellner, M. Barjenbruch, J. Klappstein, J. Dickmann, and K. Dietmayer, "Instantaneous full-motion estimation of arbitrary objects using dual doppler radar," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, 2014, pp. 324–329.

[17] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14. Oakland, CA, USA, 1967, pp. 281–297.

[18] M. Schuster, J. Reuter, and G. Wanielik, "Tracking of vehicles on nearside lanes using multiple radar sensors," in *2014 International Radar Conference*, 2014, pp. 1–6.

[19] D. Kellner, M. Barjenbruch, J. Klappstein, J. Dickmann, and K. Dietmayer, "Tracking of extended objects with high-resolution doppler radar," *Intelligent Transportation Systems, IEEE Transactions on*, vol. PP, pp. 1–13, 12 2015.

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

CHALMERS
UNIVERSITY OF TECHNOLOGY