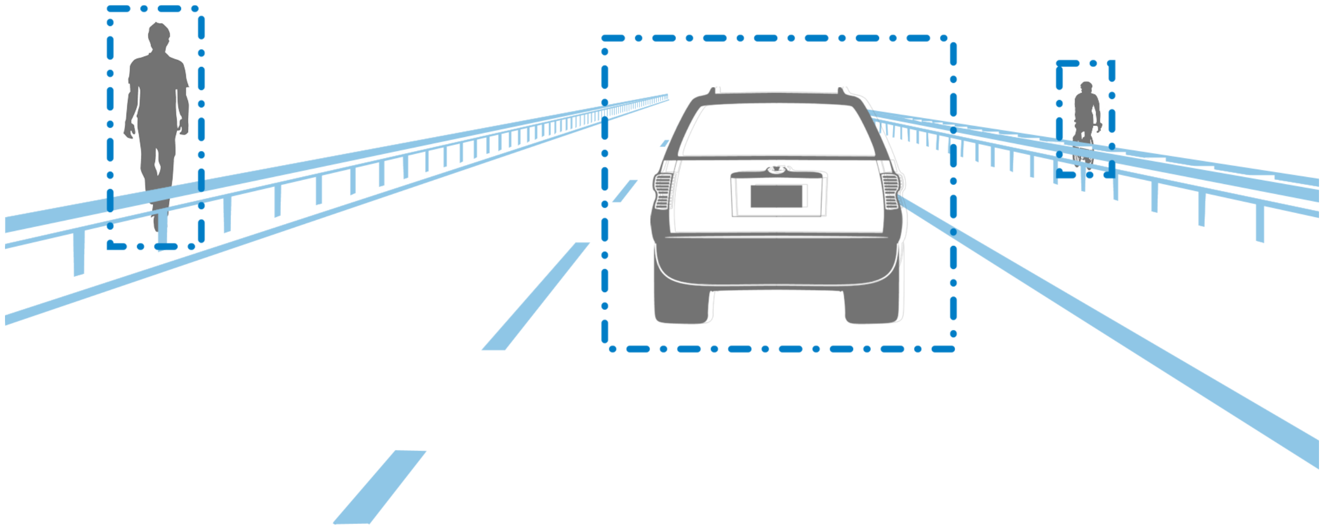
Robust Tracking Approach for Dealing with Classification Uncertainty

Ronald Immerzeel



Robust Tracking Approach for Dealing with Classification Uncertainty

by

Ronald Immerzeel

Student number:	1224581
Master:	Mechanical Engineering
Track:	BioMechanical Design
Faculty:	3Me
Publishing date:	April 23, 2018
Supervisors:	Dr.ir. R. Happee,
	ir. J.F.M. Domhof

TU Delft

Abstract

Every year about 1.25 million people die as a result of road traffic accidents [56]. Besides the traffic on the road increases every day, including the environmental impact due to the corresponding traffic emissions [69]. Autonomous driving could be a unique opportunity to increase these traffic safety, traffic flow efficiency and to reduce emissions in future [30]. In order to operate reliably and accurately, autonomous driving vehicles and autonomous features require an accurate perception of the infrastructure and other road users in the surrounding. Multi-object tracking is the process concerned with the estimation of the states of the objects in the environment, given the noisy measurements from the sensors. Besides the estimation of the states, it is also necessary to estimate the classification of the objects e.g. for a correct situation analysis and path planning. Classifiers are used to detect and classify objects from image frames, however the classification is sometimes incorrect or uncertain. This results in a decrease of tracking accuracy and an incorrect classification in these classification uncertain conditions. The contribution in this work is, by keeping the classification uncertainty in the tracking approach and using it in all steps, to jointly improve the tracking accuracy as well as the classification.

The main challenge with multi-object tracking lies in the correct association of the measurements with the corresponding tracks of the objects. Most traditional multi-object tracking approaches are focused on the explicit association of the measurements and tracks. Due to this explicit measurement-to-track association, these approaches are computational intensive in general [60] [74]. Mahler proposed the Multiple Model Cardinalized Probability Hypothesis Density (MM-CPHD) [47], an object tracking approach based on Random Finite Sets (RFS), which explicitly avoids the association of measurements and tracks. In addition, each type of road user has different motion characteristics and therefore the MM-CPHD uses class-specific motion models for an accurate prediction. However in classification uncertain conditions, the type of road user is misclassified and in that case the incorrect motion model is selected. As a consequence, the tracking accuracy reduces and the misclassification of objects are propagated to the next step of the autonomous driving process.

In this thesis a more robust multi-sensor multi-object tracking approach for dealing with classification uncertainty, called the Robust Classification Aided CPHD (RCA-CPHD), is proposed. Instead of using binary classifications, the RCA-CPHD is making use of the classification probabilities in the filter. In the MM-CPHD, the classification is only used in the prediction step of the filter. In the RCA-CPHD, however, the class-probabilities are used in all steps of the filter. For example, the prediction is based on the classification uncertainty, resulting in a more accurate prediction. In addition, the update of the tracks with the measurements is not only based on the likelihood in localisation of these measurements with respect to the tracks, but also on similarity in classification. A track with a certain class is therefore not influenced by a measurement of another class, what prevents a mix up of tracks e.g. during crossing objects. Also the gating process is not only based on likelihood in localisation, but also in classification. New appearing objects close to another object, but with a different classification, are therefore faster initiated.

The performance of the proposed RCA-CPHD, defined by the tracking accuracy and the classification accuracy, is evaluated in a two step approach; verification and validation. First a verification is done by comparing the performance of the RCA-CPHD with the MM-CPHD and the CMM-CPHD on simulated data from one radar and one camera. The camera data is processed by two different types of classifiers, which differ in the way of dealing with classification uncertainty. Two different scenarios are simulated and the performance is determined by averaging over 100 Monte Carlo runs. Subsequently the performance is validated by comparing the performance of the RCA-CPHD with the MM-CPHD and the CMM-CPHD on six real world data experiments from a stereo camera and the two different type classifiers. Two

experiments are recorded with a stereo camera on the DAVI-vehicle near the TU Delft campus. The recordings have a length of 13 and 22 seconds and detected road users are pedestrians, cyclist and cars, with a maximum of respectively 3 and 4 number of objects at the same time. Subsequently four experiments with an average length of 15 seconds are selected from the KITTI database [22] with a maximum of 9 objects at the same time. The results of both the two simulations as well as all six experiments show that the RCA-CPHD has a significant lower OSPA error and classification MSE during classification uncertain moment, e.g. occlusions, crossings and misclassifications.

Contents

1 Introduction

Every year about 1.25 million people die as a result of road traffic accidents [56]. Besides the traffic on the road increases every day, including the environmental impact due to the corresponding traffic emissions [69]. Autonomous driving could be a unique opportunity to increase these traffic safety, traffic flow efficiency and to reduce emissions in future [30]. Recently the majority of the car manufacturers have been building their own versions of autonomous driving cars [31] and autonomous features like adaptive cruise control and lane assist become more and more standard in new luxury cars. The Delft University of Technology is also contributing in the field of autonomous driving by developing autonomous vehicles suitable for different applications within the Dutch Automated Vehicle Initiative (DAVI). For example the WEPod, a vehicle designed for urban environment at low speed and a Toyota Prius designed for both the urban environment as the highway.

In order to operate reliably and accurately, autonomous driving vehicles and autonomous features require an accurate perception of the infrastructure and other road users in the surrounding. That means every road user and other important objects in the environment of the vehicle have to be detected in time and all the time. This starts by perceiving information about these multiple objects by a series of sensors attached to the car. The process concerned with the estimation of the objects in the environment, given the noisy measurements from the sensors, is called multi-object tracking. The main challenge with multi-object tracking lies in the correct association of the measurements with the corresponding tracks of the objects. This association is challenging due to undetected or false alarms of objects by the sensors, closely packed or crossing objects and dense clutter. Also objects appear and disappear continuously from the vehicle's field of view in traffic situations, resulting in an uncertainty about the number of objects. This unknown number of objects makes the measurement-to-track association even harder.

Most traditional multi-object tracking approaches are focused on the explicit association of the measurements and tracks. Due to this explicit measurement-to-track association, these approaches are computational intensive in general [60] [74]. Mahler proposed the Multiple Model Cardinalized Probability Hypothesis Density (MM-CPHD) [47], an object tracking approach based on Random Finite Sets (RFS), which explicitly avoids the association of measurements and tracks. In addition, each type of road user has different motion characteristics and therefore the MM-CPHD uses class-specific motion models for an accurate prediction. However in classification uncertain conditions, the type of road user is misclassified, and in that case the incorrect motion model is selected. As a consequence, the tracking accuracy reduces and the misclassifications are propagated to the next step of the autonomous driving process.

In this thesis a more robust multi-sensor multi-object tracking approach for dealing with classification uncertainty is proposed. In this approach, called the Robust Classification Aided CPHD (RCA-CPHD), the classification uncertainty is maintained and used in all steps of the filter. The performance of the RCA-CPHD is evaluated on both simulated and real data and compared to the MM-CPHD filter and CMM-CPHD. Chapter 2 starts with the application and an introduction in autonomous driving is given. In chapter 3 the fundamentals of object tracking, object tracking approaches based on random finite sets and the use of classification in these approaches are discussed. Then the proposed approach, the RCA-CPHD, is explained in depth in chapter 4 and the performance of the RCA-CPHD is evaluated on both simulated and real data in chapter 5. Finally, in chapter 6, the conclusion is drawn and recommendations are made for future work.

2 Application

Every year about 1.25 million people die as a result of road traffic accidents [56]. Besides the traffic on the road increases every day, including the environmental impact due to the corresponding traffic emissions [69]. Autonomous driving could be a unique opportunity to increase these traffic safety, traffic flow efficiency and to reduce emissions in future [30]. Right now, the first stage of automated driving is already integrated on the roads. Many new cars across all vehicle segments are nowadays provided with driver assistance systems, such as Adaptive Cruise Control (ACC), Lane Keeping Assist (LKA), Forward Collision Prevention (FCP).

The path towards fully automated driving is not a path of technology only, but also one of e.g. legal challenges and human factors. So in order to introduce automated driving in a safe and legal way on the public roads, investigation, improvement, evaluation and demonstration of automated driving is needed [30]. Addressing these needs is the common goal of the Dutch Automated Vehicle Initiative (DAVI) and the research done in this thesis is part of DAVI. A collaborative initiative between the Delft University of Technology, Connekt, RDW, TNO, Toyota Motors Europe as well as many other Dutch and international partners.

Section 2.1 starts with an introduction in automated driving. The different levels of automation, the challenges of driving on public roads and a system architecture of an autonomous driving vehicle are discussed. Then in section 2.2 the step in the system architecture where this thesis is about, called object tracking, is discussed in more detail. Subsequently two autonomous driving vehicles which DAVI uses to address the challenges of autonomous driving are discussed in section 2.3. The sensors used in these vehicles are discussed in more depth and the used sensor setup in this thesis. Then in section an overview is given in the detection of objects in camera images and the corresponding issue of classification uncertainty. The challenges in object tracking for autonomous driving together with the uncertainty in classification finally results in the problem definition discussed in section 2.5.

2.1 Introduction to Automated Driving

Right now, the first stage of automated driving is already integrated on the roads. Many new cars across all vehicle segments are nowadays provided with driver assistance systems, such as Adaptive Cruise Control (ACC), Lane Keeping Assist (LKA), Forward Collision Prevention (FCP). The path to fully automated driving will be taken step by step. To quantify the steps of automation, the Society of Automotive Engineers (SAE) provided a harmonized classification system for defining the degree of automation [8].

2.1.1 Levels of Automation

The classification system for defining the degree of automation is divided into six different levels of automation, rising from no automation at all towards full automation [10] as shown in figure 2.1.

- at **Level 0** there is no automation at all. The longitudinal control as well as the lateral control is done by the *driver only*. Longitudinal control is the control in longitudinal direction of the car, i.e. tasks like maintaining speed, accelerating and braking. Lateral control is the control in lateral direction of the vehicle i.e. steering and lane changing.
- at **Level 1** One of the *longitudinal or lateral control* is accomplished by the system, while the driver is performing the other task at the same time.
- at **Level 2** the system performs the *longitudinal and the lateral control* in a specific use case. The *driver is continuously monitoring* the system to resume the control of the vehicle immediately when needed.
- at **Level 3** the system performs the *longitudinal and the lateral control* in a specific use case. The *system independently recognizes its limits*, so the driver does not have to monitor the system at all times, but must always be able to resume driving when the systems tells him to do so.
- at **Level 4** the system is able to perform on its own for all situations in a defined use case.
- at **Level 5** the system is able to perform on its own for all situations during the entire journey.

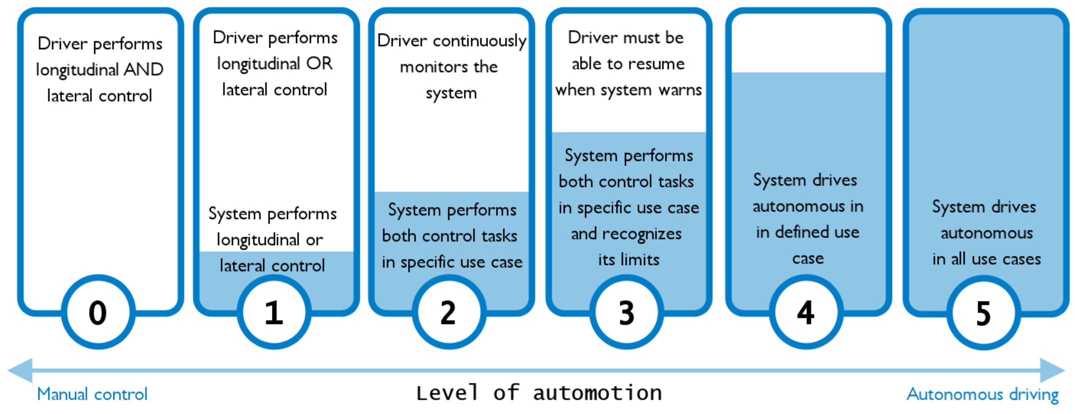

Driver performs longitudinal AND lateral control
Driver performs longitudinal OR lateral control
System performs longitudinal or lateral control
Driver continuously monitors the system
System performs both control tasks in specific use case
Driver must be able to resume when system warns
System performs both control tasks in specific use case and recognizes its limits
System drives autonomous in in defined use case
System drives autonomous in all use cases
0
1
2
3
4
5
Manual control
Level of automotion
Autonomous driving

Figure 2.1: The Society of Automotive Engineers (SAE) provided a harmonized classification system for defining the degree of automation [8], rising from no automation at level 0 to full automation at level 5

Right now and in the near future, the first levels of automation will operate next to each other on the roads. Manually driven cars will be driving in traffic, together with automated vehicles of varying levels of automation. If and when fully automated driving is to be achieved is yet still unknown. Before fully automated could be achieved, some challenges must be addressed first [30].

2.1.2 Challenges of automated driving on public roads

The path towards fully automated driving is not a path of technology only, but also one of for example legal challenges and human factors. Basically the challenges can be subdivided into four different categories [30], namely:

1. **Technological challenges:** An autonomous vehicle must be able to perceive its environment and able to navigate without human intervention. First of all an autonomous vehicle have to be aware of its surroundings. So what is the infrastructure and which other road users are in the environment. This sensing should be reliable for every situation and in every condition the vehicle may be operating in. Besides the sensing technological challenges, reliable control strategies for both longitudinal control and lateral control have to be developed as well.

2. **Challenges related to human factors:** The higher the level of automation of the vehicle, the more the role of the human driver changes. This role changes from manual controller at low level of automation to monitoring the system only at higher level of automation. But is the human driver able to monitor the system at all times and how does other manual drivers on the road react on autonomous vehicles?

3. **Traffic management challenges:** Managing the traffic, like strategies as platooning vehicles or lane specific control, have high potential in increasing traffic flow efficiency, traffic safety and reduction of emissions [30]. Studies must show whether this potential lasts in real situations.

4. **Traffic safety en legal challenges :** The benefits of automated driving on traffic safety, taking into account the technological aspect as well as human factors, must be examined in order to quantify these benefits. Also before commercial autonomous vehicles are able to drive on the road, approval should be given by authorities. Another challenge that needs to be solved is who is responsible and liable when a automated vehicle monitored by a human driver causes an accident?

The focus in this thesis is on technological challenges of automated driving. An autonomous vehicle must be able to perceive its environment and able to navigate without human intervention. This process could be subdivided into several steps, resulting in the system architecture of an autonomous vehicle.

2.1.3 System architecture

How does an autonomous driving vehicles actually work? Let's look first at the way a human is driving a car. When a human is driving, he will perceive the environment with, for the most part, his eyes. The perceived environmental information is subsequently processed in the brain in order to estimate the best control action for that specific situation, e.g. steering or braking. And subsequently that control action will be taken.

The same principle applies to an autonomous driving car. The environment is perceived via numerous of sensors and subsequently processed in order to get an appropriate perception of the environment. This environmental perception is then analysed to get an insight of the current situation and subsequently the best path to take for the vehicle is estimated. Finally the best control action to carry out this path is estimated and the commands for this control action are send to the actuators.

These three steps, the *environmental perception*, *situation analyses* and *planning & control*, are the main steps of the system architecture of an autonomous driving car. These main steps can be subdivided into a series of smaller steps as shown in figure 2.2.

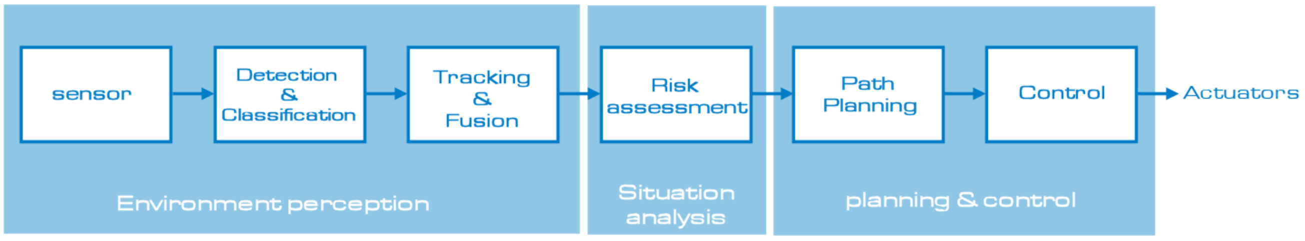


Figure 2.2: The system architecture for an autonomous driving vehicle subdivided into smaller steps

Environmental Perception

The environmental perception is done in three steps. First raw data is received by the sensors, then a detection and classification step is done on this raw data and subsequently the data is fused and relevant objects are tracked.

Sensors In the first step raw data of the environment is perceived from a series of sensors placed around the vehicle. Common used sensors are radar, laser, camera and sonar. Each sensor type has its own strength and weakness and is giving different raw data as output. In this thesis radar and camera are used and are explained in more depth in section 2.3.

Detection/classification In the detection/classification step, the raw data is analysed to get an insight in the important objects in the environment. Pedestrians, cars, cyclists all have different behaviour and require different control actions later on. Therefore the objects are not only detected but also classified. Object classification and the associated challenges are explained in more depth in section 2.4

Tracking/Fusion The detected and classified objects consists not of real objects only, but also of noise and false alarm. Also the predicted classification is not always correct and the measured location may deviate from the actual location. Besides it is also important to get an insight in the movements of these objects in time, in order to correctly estimate the current situation of the vehicle. However to estimate movements, the current measurements have to be associated with the estimated objects from the previous time step. This association could be hard in situations where objects are close to each other or in case of false alarms.

Therefore in the tracking and fusion step, verification of the objects and the estimation of their corresponding movement is done, using the noisy measurements from multiple sources. The focus in this master thesis is on this tracking/fusion step and will be discussed in more detail in section 2.2.

Situation Analysis

Risk assessment When the objects in the environment and their corresponding states are estimated, it possible to analyse the current situation of the car. So in the risk assessment step the objects that might be dangerous for the vehicle, and hence are relevant to focus on, are mapped.

Planning & Control

Path planning After examining the current situation of the vehicle and mapping the relevant objects in the environment, the optimal path for the vehicle is estimated. So what path should the vehicle take in the current situation in order to avoid collision, to change lanes, etc. and what is the required velocity to carry out that path.

Control In the last step, the control step, the planned path is converted to inputs for the actuators controlling the vehicles dynamics. Subsequently these inputs are send to the actuators and the vehicle carries out the path.

Real Time

In order to react fast enough to the changing environment, the entire process from sensor to action has to be real time guaranteed. This means that the received data from the sensor has to be processed before the next new sensor data arrives. For example when the sensors are working on 15 hz, the object tracker receives every 67 milliseconds a new package of measurements. So the first package has to be processed within these 67 milliseconds before the next package arrives. And this has to be guaranteed for every situation without exceptions and should therefore be taken into account when looking for the best algorithm for the DAVI vehicle.

2.2 Object Tracking in Automotive

Situation awareness is crucial for safe navigation in autonomous driving. To achieve this, it is important to identify all the relevant objects in the environment and to determine how they are moving in time. This process starts by continuously gathering information about these objects using multiple and different types of sensors on the vehicle. The measurements obtained by these sensors are however noisy. The process of the verification of the objects and the determination of the corresponding movements, using these noisy measurements, is called *object tracking*.

In the literature a distinction is made between two different types of object tracking, namely *single-object tracking* and *multi-object tracking*. In single-object tracking the system is only concerned with the tracking of only one object, while by multi-object tracking the system is concerned with multiple objects moving in the same surrounding.

2.2.1 Single-object tracking

In single-object tracking, the system is only concerned with the tracking of one object. So if the sensors are detecting more than one object, it is assumed that only one measurement is correct and the others are false alarm [62]. The challenge in the measurement-to-track association is therefore to distinguish the correct measurement generated by the object from the measurements that are false alarms. This fundamentally difference from multi-object tracking techniques where the existence of more than one object simultaneously is taking into account in the measurement-to-track association [62]. In traffic situations numerous of objects could be present in the environment, making single-object tracking techniques not suitable for autonomous driving.

2.2.2 Multi-Object Tracking

In multi-object tracking the system is concerned with the existence of multiple objects in the environment. This results in a number of challenges, such as the common challenges *uncertain data association*, *Unkown number of objects*, *Dense clutter disturbance*, *Manoeuvrable motion* and *Real-time processing requirements* [63].

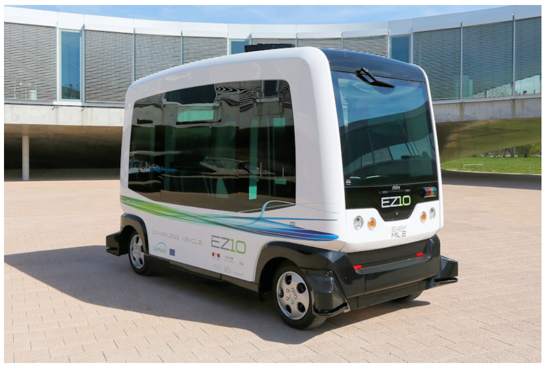
- **Uncertain data association:** One of the main reasons that leads to the complexity of multi-object tracking is the measurement-to-track association. When multiple objects exist in the environment, it is no longer clear which measurement is generated by which object. Especially in situations where objects are closely packed or crossing objects and situations with a lot of clutter measurements.
- **Unknown number of objects** Objects appear and disappear continuously in the environment, for example when a bicycle is leaving the field of view of the camera or a pedestrian gets into a car. Besides false detections or missed detections may be present in the sensor data. When the number of objects is uncertain, it is no longer clear how many measurements have to be associated with how many objects. This makes the measurement-to-track association even harder.
- **Dense clutter disturbance:** The measurements received from the sensor does not only consist real existing data, but also of clutter. For example due to influences like weather conditions and sensor system noise on the background. When the distinctness between a real measurement and this dense clutter is hard, false tracks and missing tracks could arise.
- **Manoeuvrable motion:** In traffic situations, the objects in the environment have manoeuvrable motions. At the same time, the vehicle containing the sensors is moving as well. To model this ego-motion together with the manoeuvrable motions of the objects correctly, sophisticated models are required. However these models turn out to be non-linear and non-Gaussian.
- **Real-time processing requirements:** For safe driving, the system has to react fast enough to the changing environment and real time processing of the data is inevitable. However the computation complexity increases rapidly at large number of objects, which makes it hard to meet the real-time processing requirement.

2.3 The Dutch Automated Vehicle Initiative

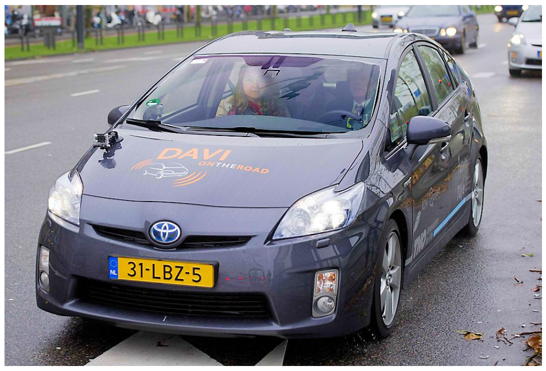
The Delft University of Technology is also contributing in the field of autonomous driving within the Dutch Automated Vehicle Initiative (DAVI), a collaborative initiative between the Delft University of Technology, Connekt, RDW, TNO, Toyota Motors Europe as well as many other Dutch and international partners. The common goal of DAVI is to introduce automated driving in a safe and legal way on the public roads by investigation, improvement, evaluation and demonstration of autonomous driving [30].

2.3.1 DAVI's Automated driving vehicles

One of approaches to address the different challenges in autonomous driving, is by developing and extensively testing automated vehicles. The testing is not only done on a test track, but also on closed roads and on public roads in mixed traffic. Two of these projects are the WEPod and an autonomous driving Toyota Prius.



(a) WEPod vehicle designed for urban environment at low speed [81]



(b) Toyota Prius designed for both the urban environment and highway [1]

Figure 2.3: Two autonomous driving vehicles developed by DAVI

The WEPod is fully automated driverless vehicle, developed to operate in urban environment at low speed. Its application is to transport people from the Wageningen train station to the Wageningen University [29]. The focus in the WEPod is mainly on urban environment at low speed. Next to the WEPod a Toyota Prius is being developed which is, next to the urban environments, also able to operate in highway environments at higher speeds.

2.3.2 Sensor setup

The proposed approach in this thesis is evaluated on data from the Toyota Prius. In order to observe all activities around the vehicle, the vehicle is covered with multiple sensors. In total nine radar sensors and nine cameras are installed on the Toyota Prius of DAVI, respectively three radars and three cameras on front, two radars and two cameras on the back and two radars and two cameras on both sides. A schematically overview of the sensor setup is shown in figure 2.6.

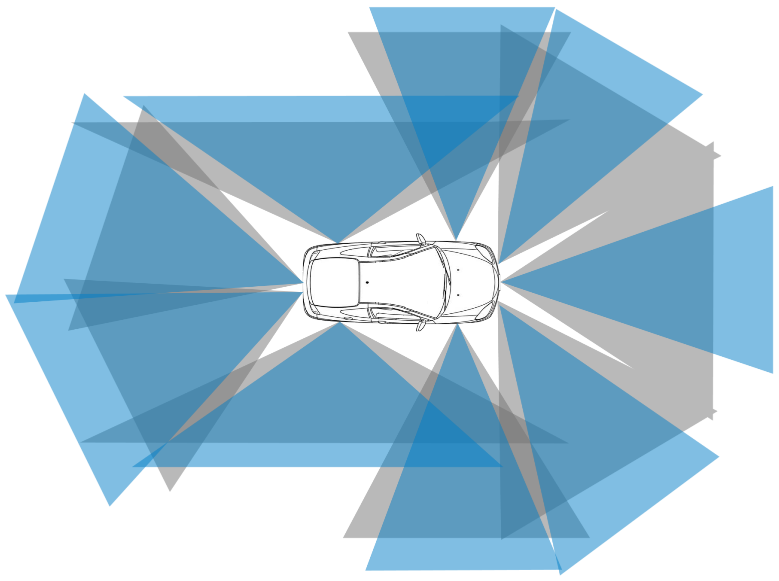


Figure 2.4: schematic overview of sensors on vehicle. The blue areas are visualizing the field of view of the radars and in grey the field of view of the cameras

In this thesis, the proposed approach is evaluated on the combination of one radar and one camera at the front of the vehicle and on a stereo camera also mounted at the front of the vehicle. When using a mono camera, the inaccuracy of the depth perception is compensated by the more accurate depth measurement of the radar. For the stereocamera a more accurate depth perception is obtained using the disparity between the two cameras as explained in more depth in the next paragraphs. The proposed approach is evaluated with a single sensor for each type, but has the possibility to expand to multiple sensors and to use other types of sensor.

Radar

Radars are commonly used for road vehicles for sensing both near and far obstacles. Radars are popular because they are robust mechanically and operate effectively under a wide range of environmental conditions [57]. For instance, radars are in general unaffected by ambient lighting or by the presence of rain, snow, fog, or dust. However due to a fixed output power of the radars for safety reasons, there is a general trade-off between field of view and range [57].

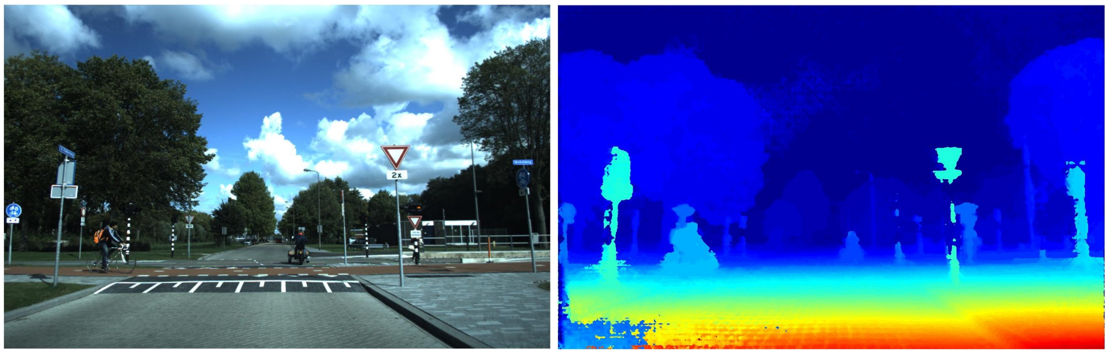
In the DAVI vehicle a long range radar, the Continental ARS 309-2 sensor, and a small range, the Continental SSR sensor, are used. Due to their long range, the Continental ARS 309-2 sensor is used on the front, back and on the front-sides of the car. However the long range benefit is at the expense of a smaller field of view compared to the small range radar.

The small range radar Continental SSR sensors are installed on the backsides of the car due to their larger field of view. However this comes at the expense of a smaller range, but is still within the range required. Both types of radar measure independently the range R, the velocity of this range $\dot{R}$ and the angle α in polar coordinates [9]. The conversion of the radar's polar coordinate system to the vehicle's Cartesian coordinate system is explained in more depth in chapter 4.3 about the applied measurement models.

Camera

Next to the radar, the DAVI vehicle is equipped with nine cameras. A camera is basically a pixel-based sensor giving the position-coordinates u and v of an object in the frame. When using only one camera, the perception of depth is estimated by the height of the object in the frame and is therefore inaccurate. So in order to measure the distance of the object more accurate, at least two cameras are required and the distance could be determined by stereo vision.

In stereo vision two cameras are horizontally displaced from each other, resulting in two different views of the scene. The difference in horizontal coordinates of corresponding image points, called the disparity, gives then a measure of the relative depth of these image points. So when an object is close to the cameras the disparity is large with respect to an object far away from the camera. Using the characteristics of the camera, the depth of an object can then be estimated from the disparity. A visualization of a dense disparity map from a random traffic situation is shown in figure 2.5.



(a) Image frame created by the camera of a random traffic situation *(b) Disparity map of the image frame obtained from stereo vision*

Figure 2.5: Stereo vision gives the possibility to determine depth of objects using the disparity

The camera is perceiving information about the environment by capturing image frames in time. Such an image frame is however only an array of pixel values and does not tell anything about the number of objects in the environment and their corresponding location and classification. Therefore the images are first processed by a *classifier* or *object detector* to detect the relevant objects in the image frame. Common used methods of classification, the challenges in classification and the applied methods in this thesis are explained in more depth in the next section.

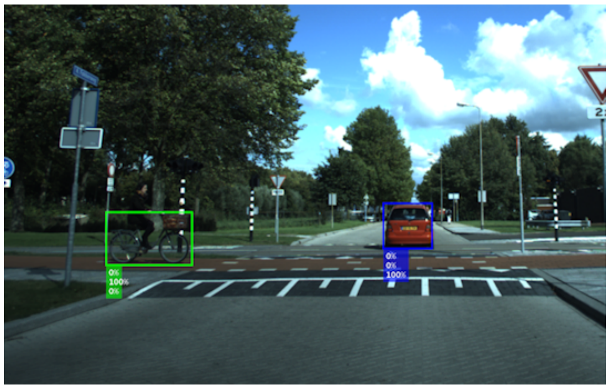


Figure 2.6: A classifier is able to detect relevant objects in an image frame. In this case a bicycle and car is detected

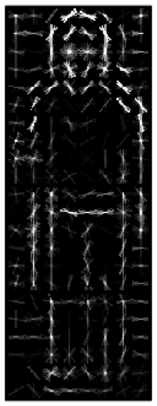
2.4 Object Classification

The camera is perceiving information about the environment by continuously capturing image frames in time. Such an image frame is however only an array of pixel values and does not tell anything about the number of objects in the environment, their location and their classification. For example a colour image with a resolution of 1280x1024 may be described by an 1280x1024x3 array, where each cell describes the amount of red, green and blue in a range of 0 to 255. So before the camera is outputting the information to the tracking filter, a step called *classification* might be done to detect and localize the relevant objects in the environment. Two common used methods of classification are by using Histogram Oriented Gradients (HOG) features with a Support Vector Machine (SVM) detector and Convolution Neural Network (CNN).

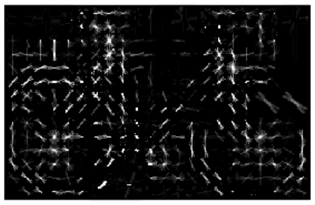
2.4.1 SVM detector using HOG features

As a human we are easily able to identify a pedestrian, car or bicycle in the environment. This is due to the fact that we have learned how a pedestrian, car or bicycle looks like. As a child we have seen i.e. multiple cars and we created in our brain a kind of general model for a car. This general model gives us the possibility to identify a car in different environments. The same principle is done by image detectors, i.e. the SVM detector using HOG features.

The SVM detector learns a model for each type of object in a more general form than the colour pixel values, e.g. gradient vectors. A gradient vector indicates how much pixels change in value with respect to surrounding pixels. The detection process starts by transforming an image to the gradient vector form. Subsequently a sliding window scans over the image and each window is used as an input for a trained classifier, in this case a Support Vector Machine (SVM). The SVM compares each window to the gradients of pre-trained models, in our case a car, bicycle or pedestrian, and classifies each window as being a certain class or not. A visualization of certain pre-trained models is shown in figure 2.7.



(a) pre-learned model of a pedestrian



(b) pre-learned model of a bike

Figure 2.7: pre-learned HOG models are used for classifying objects in an image

The objects in the image could however vary in dimensions in the image, for example when an object appears further away or closer to the camera. So the image is sub-sampled to multiple sizes and each sub-sampled image is scanned for objects. The final result is bounding boxes with different dimensions of all the recognized objects and their corresponding classification. The approach of a SVM classifier using HOG features is, for example, implemented in the classifier of Felzenszwalb et al. [16]. In this approach an object is not described by one model, but is divided into a mixture of deformable part models. The part models could for example be the wheels of a bicycle. In the pre-trained model, all these parts together have a certain configuration and shape. The more these parts deviate from the original configuration, the less likely they will be a certain class.

The proposed approach in this thesis does not only use the detections, but also the classification probabilities of these detections. The SVM classifier itself is not able to calculate class-probabilities, since it is a non-probabilistic binary classifier. Methods such as Platt scaling [61] and Isotonic Regression [82] are however able to give an approximation of the class-probabilities. By Platt scaling, next to the SVM, also parameters of an additional sigmoid function are determined by training for scaling the SVM output scores into probabilities. The assumption of a sigmoid-shaped relation between the output scores and the probabilities of the Platt scaling makes this method more effective than Isotonic Regression when the distortion in the predicted probabilities is sigmoid-shaped [55]. However when the distortion in the prediction probabilities is monotonic, the Isotonic Regression is more powerful [55]. The Isotonic Regression is however more sensitive to overfitting and performs therefore worse than Platt Scaling when the number of training data is low [55].

2.4.2 Convolutional Neural Network

Another approach for object detection and classification is by a Convolution Neural Network (CNN. When we see an image of a traffic situation, we are easily able to identify a car, pedestrian or a cyclist in the image. We achieve this by identifying unique features, like two legs and arms by a pedestrian, which makes an object a certain object. A CNN does this in a similar way. The computer starts the image classification by looking at low level features such as edges and curves and subsequently builds this up to more abstract concepts through a series of layers. This networks of layers can, in some way, be seen as the neural network used in our brain.

CNN principles for classification

The first layer of the network is always a *Convolutional Layer*. This layer extracts low-level features from the image, i.e. edges, lines, and corners. A visualization of some of these low-level features is shown in figure 2.8. The extraction of low-level features is done by "scanning" over the image with a specific filter for each feature, resulting in an *activation map* or *feature map*. This activation map gives a representation of how well each scanned part in the image corresponds to the shape of that specific filter. A visualization of an activation map using a filter for edges between black and white is shown in figure 2.9.

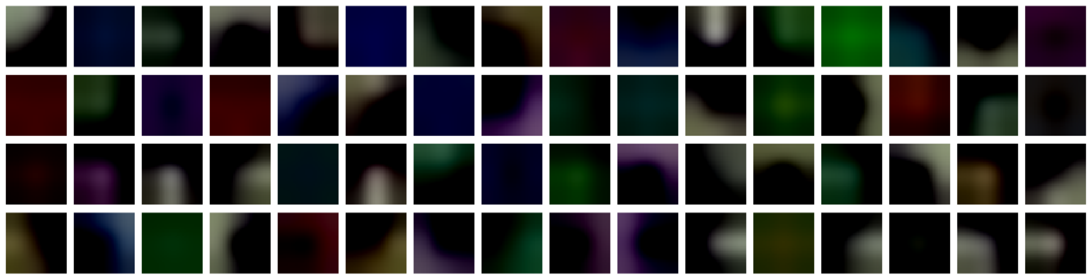


Figure 2.8: visualization of the low level filters used in the first convolution layer of the CNN

The output of the first convolution layer becomes the input of the second convolution layer. This convolution layer is scanning the activation map for higher-level features like for example semicircles or squares, which results in a new activation map. When going deeper and deeper through the network, the filters consist of more and more complex features and finally results in features describing an entire object.

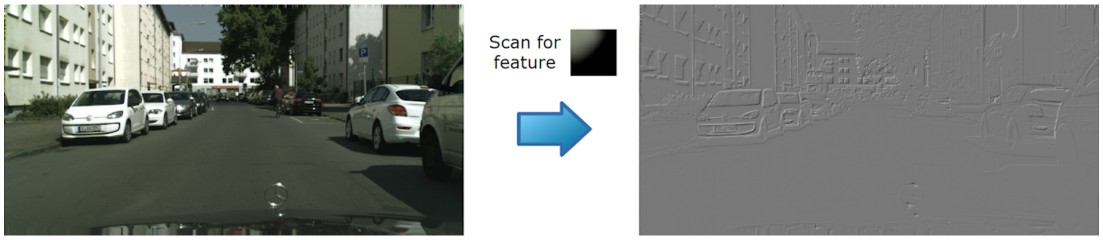


Figure 2.9: Visualization of feature map after using one filter for edges from black to white

To increase the robustness of the neural network, some extra layers may be applied between the convolution layers. Common used layers are the *ReLU layers*, *Pooling layers* and *Dropout layers*.

- *ReLU Layer* The basic purpose of this layer is to introduce non linearity in the network, since most of the real-world data is non-linear. This is achieved by applying the function $f(x) = max(0, x)$ to all the values in the activation map. Or in other words, all the negative pixel values in the activation map are set to zero.
- *Pooling layer* This layer reduces the dimensions of the activation map in order to make it computationally faster and to control over-fitting. Over-fitting describes the problem that a model is too much tuned to its training data, that it is no longer capable to generalize when it comes to test data.
- *Dropout layer* The dropout layer is also for controlling over-fitting and is active during the training phase of the network. It randomly "drops out" a number of values in the activation map by setting them to zero.

At the end of the network there is always the *fully connected layer* or *softmax layer*. This layer looks at the features resulting from the second-last layer and determines which features most correlate to a particular class and the corresponding probability.

2.4.3 Classifiers used for evaluation

The evaluation of the proposed approach is done with two different types of CNN classifiers, namely a detector and a semantic segmentation classifier. Both type of classifiers differ in the way of presenting the recognized classes in the image. This results in a different classification during classification as explained in more depth in the next section and therefore both classifiers are interesting to consider in the evaluation.

Single Shot MultiBox Detector

The first classifier is the Single Shot MultiBox Detector (SSD) developed by Wei Liu et al. [36]. This classifier, as the name suggests, is a detector and is the classifier installed in the DAVI-vehicle. This classifier uses a predefined set of bounding boxes and determines for each bounding box the probability of being a certain class. Bounding boxes with a probability larger than a predefined threshold are subsequently marked as "detections". The output of this classifier is a set of detections consisting of bounding boxes and a corresponding confidence value, basically a probability, of the class.

For the evaluation, the output of this classifier has been adjusted to output the probability of all the observed classes. However the used model in the classifier observes 20 different classes and in the evaluation only pedestrians, cyclists and cars are considered. Therefore the probabilities of all the other classes are considered to be zero and the probabilities are normalized for pedestrians, bicycles and cars. This changes the output to, given a detection, the corresponding probabilities of being a pedestrian cyclist and car and not the probability of the class given by the classifier in the first place.

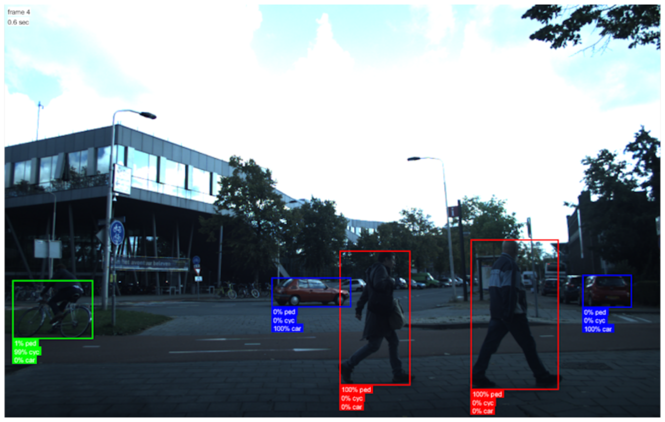
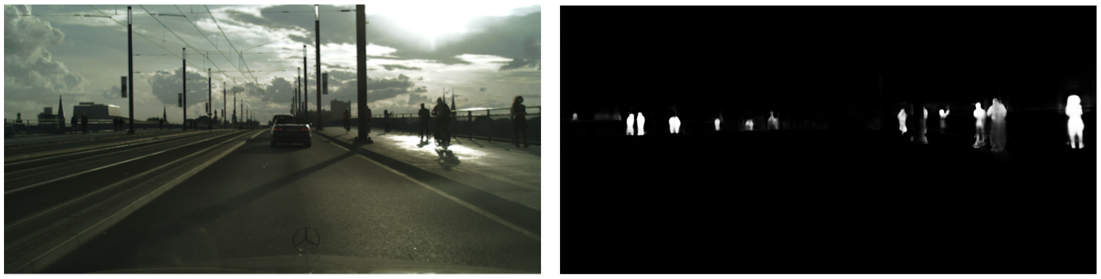


Figure 2.10: A visualization of the detections of pedestrians, bicycles or cars by the SSD detector. The SSD detector is modified in this thesis to output the probability of both pedestrians, cyclists an cars.

Laplacian Pyramid Reconstruction and Refinement

The second used classifier is the Laplacian Pyramid Reconstruction and Refinement (LRR) developed by Ghiasi et al. [25] with a pre-trained network for cityscapes. The difference with respect to the detector is that this classifier outputs the class probabilities per pixel in the image instead of per detection as visualized in figure 2.11 for pedestrians. The pre-trained cityscape model is also capable of classifying riders, making it, combined with bicycle classifications, able to classify cyclists.



(a) image used as input to the CNN trained for cityscapes *(b) visualisation of the probability map of pedestrians*

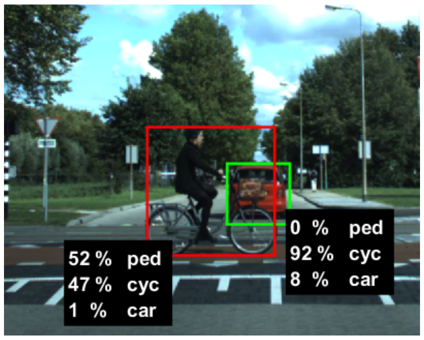
Figure 2.11: The output of the CNN network is a probability map for each observed class.

The proposed approach in this thesis uses however class probabilities per object and not per pixel in the image. Therefore for the evaluation of the proposed approach, the bounding boxes are annotated and the median of the class probabilities of all the pixels in this bounding box are taken as the corresponding class probabilities. Also the class probabilities are normalized for the three considered classes, resulting in the probabilities of being a pedestrian, cyclist and car given the detection.

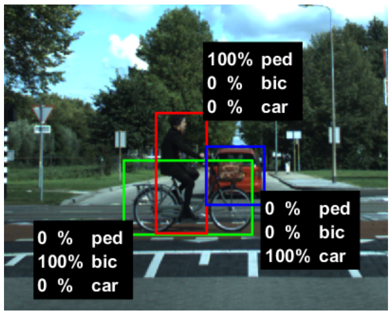
2.4.4 Uncertainty in Classification

The classification from the classifier is, however, sometimes incorrect. A correct classification is required for a correct situation analysis of the autonomous vehicle and the corresponding path planning. In addition, the classification is sometimes also used to improve the object tracking, e.g. with class-specific motion models for the prediction of the object's movement. An incorrect classification can therefore result in the selection of an incorrect motion model and thence in inaccurate predictions, what finally result in tracking errors.

The LRR classifier calculates the classification probability per pixel and the SSD classifier per detected object, and therefore they differ in the way of misclassifying. A common misclassification for road users is a cyclist classified as a pedestrian, i.e. due to the fact that the rider on the bicycle is being seen as a pedestrian. In the case the SSD classifier detects the cyclist as two objects, namely a bicycle and a pedestrian. The LRR classifier with annotation of the objects, on the other hand, classifies the cyclists as a single object with a lower probability of being a cyclist and higher of being a pedestrian. This difference is visualized in figure 2.12. This fundamental difference makes it interesting for taking into account both classifiers in the evaluation, in order to see if the differences in classification also results in a difference in tracking accuracy and classification accuracy.



(a) LRR classifier



(b) SSD classifier

Figure 2.12: In case of cyclist with high class uncertainty, the LRR detects the bicycle with high probability and a false alarm of a pedestrian. The SSD, on the other hand, a single object with lower probability of being a cyclist and higher probability of being a pedestrian

Misclassifying is also not consistent in time. For example a rider on a bicycle can be incorrectly classified as a pedestrian at a certain time instance, while a fraction later it is correctly classified as a cyclist again. This is for example seen in figure 2.13.

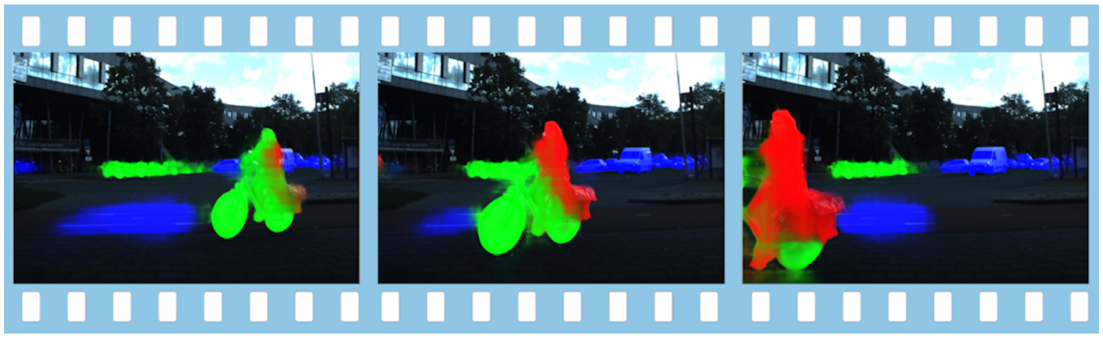


Figure 2.13: Mistakes in classification are not consistent in time. In this example the cyclist is correct classified as a cyclist (green), while a few frames later for most part incorrect classified as pedestrian (red)

2.5 Problem Definition

This thesis is concerned with object tracking for automated driving. The application is an autonomous driving Toyota Prius from Davi, equipped with radars and cameras. In this thesis a setup is used of one camera and one radar and a setup of a stereo camera explained in more depth in section 2.3. The camera frames are processed by a classifier in order to detect the objects in the environment and their corresponding classification, as discussed in more detail in section 2.4.

In traffic situations an autonomous driving vehicle has to deal with multiple road users. So for object tracking in automotive applications, the algorithm has to deal with multiple objects and the corresponding general issues as discussed in section 2.2. Road users appear and disappear continuously in the the environment, resulting in an unknown number of objects. Also road users are crossing or approaching each other, making the measurement-to-track association hard. This is made even more challenging due to dense clutter disturbance.

In order to improve the measurement-to-track association, predictions are made using the motion model corresponding to object. This motion model is different for every type of object, and the selection of the correct motion model is based on classification from the object detector and classifier. This classification is however sometimes incorrect, e.g. during occlusion or a bike rider classified as a pedestrian. This results in an uncertainty in classification and finally in tracking errors due to inaccurate predictions. Besides a correct classification is required for the subsequent steps of the system architecture for an accurate situation analysis and path planning.

So in order to create a object tracking algorithm for automotive applications, the algorithm has to deal with the general challenges of multi-object tracking. Besides it has to take into account the uncertainty in classification to prevent tracking errors and classification errors for the subsequent steps in the systems architecture. That results in the following problem definition:

In what way can classification uncertainty be used to improve tracking accuracy and classification accuracy for object tracking in classification uncertain conditions?

3 Related Work

A good perception of the environment is crucial for autonomous driving and therefore all the relevant objects in the environment have to be identified. Multiple sensors on the autonomous vehicle continuously gathering information about these objects. This measurement data is however noisy and the association between the data and the objects is not always clear. In the previous chapter was explained that the process concerned with the verification of the objects and determination of the corresponding movements, using these noisy measurements, is called *object tracking*.

In this chapter the fundamentals of object tracking and related work in object track are discussed. In the literature a distinction is made between two different types of object tracking, namely *single-object tracking* and *multi-object tracking*. In single-object tracking the system is concerned with the tracking of only one object, while in multi-object tracking the system is concerned with multiple objects moving in the same surrounding.

Section 3.1 starts with the fundamentals of single-object tracking and common used approaches for single-object tracking. This is extended in section 3.2 to multi-object tracking. The individual association of each object with each measurement in most traditional multi-object tracking approaches results, however, in a high level of computation load [62]. This makes these approaches no longer feasible for real-time scenarios when the number of objects increases. Therefore in in section 3.3 approaches based on Random Finite Sets (RFS) are discussed. The use of these Random Finite Sets make it possible to avoid the explicit association between measurements and tracks. In section 3.4 are subsequently related work on RFS multi-object tracking approaches which are using classification in their approach discussed. Finally, in section 3.5 there is a discussion about the different approaches and their pros and cons.

3.1 Single-object tracking

In single-object tracking, the system is only concerned with tracking of one object. This object is described by its *state*, which is a vector **x** containing the location and movement of an object at a certain time instant. In Cartesian coordinates at time instant k this vector is, for example, the position and velocity of the object relative to the autonomous vehicle:

$$\mathbf{x}_k = \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ z \\ \dot{z} \end{bmatrix} \tag{3.1.1}$$

Sometimes the acceleration is included in the state as well, however in this thesis only the position and velocity are taken into account. The camera measurement data is only giving information about the position of the objects. Estimating the acceleration based on the position would not be reliable due to the error caused by second order differentiation.

Measurement Model

The output of each sensor type may differ in format of the quantities used in the state vector. For instance, the radar measures the position of an object in polar coordinates with range r, range velocity $\dot{r}$ and the corresponding angle α of the object with respect to the radar. The camera, on the other hand, after processing the image frame with a classifier, outputs the position by pixel coordinates u and v of the object in the frame and the classification C. When using two cameras with stereovision, the camera also gives the disparity d used for depth estimation.

$$\mathbf{z}_{radar} = \begin{bmatrix} r \\ \dot{r} \\ \alpha \end{bmatrix} \qquad \mathbf{z}_{camera} = \begin{bmatrix} u \\ v \\ C \end{bmatrix} \qquad \mathbf{z}_{stereovision} = \begin{bmatrix} u \\ v \\ d \\ C \end{bmatrix} \tag{3.1.2}$$

Since the object's state and the measurements have different formats, it is not possible to compare them directly. Therefore a *measurement model* or *observation model* is used to make this possible. A measurement model **z** = **z**(x) transforms the quantities of the state to the format of the measurements. The converted format of the state is then equal to the format of the measurements, making it possible to determine the likelihood of each measurement being generated by an object.

So the main goal of the object tracking filter is to estimate the state **x**, giving the measurements **z** received by the sensors. This estimation is very often based on probabilistic methods. For example the Bayesian Filter and the Kalman Filter are using Bayes' rule for combining prior and observation information [12]. These single object filters are outlined in the upcoming sections.

3.1.1 Bayes Filter

The foundation of the Bayesian filter is *Bayes' Rule*. This rule assumes that state **x** and measurement **z** are related as a joint probability $P(\mathbf{x}, \mathbf{z})$. The chain-rule of conditional probabilities tell us that the joint probability can be expanded in two ways.

$$\begin{aligned} P(\mathbf{x}, \mathbf{z}) &= P(\mathbf{x}|\mathbf{z})P(\mathbf{z}) \\ &= P(\mathbf{z}|\mathbf{x})P(\mathbf{x}) \end{aligned} \tag{3.1.3}$$

The goal of the filter is to estimate the state **x**, given the measurements **z**. However due to the uncertainties in these measurements, the state is expressed as likelihood of each measurement being the posterior state. This likelihood $P(\mathbf{x}|\mathbf{z})$ is called the *conditional probability*. Rearranging equation 3.1.3 gives this conditional probability and is called Bayes' Rule.

$$P(\mathbf{x}|\mathbf{z}) = \frac{P(\mathbf{z}|\mathbf{x})P(\mathbf{x})}{P(\mathbf{z})} \tag{3.1.4}$$

So the state, expressed as conditional probability, is now described by the probability functions $P(\mathbf{x})$, $P(\mathbf{z}|\mathbf{x})$ and $P(\mathbf{z})$.

- The probability function $P(\mathbf{x})$ is the *prior probability* and describes what value of **x** is expected from prior beliefs before looking at the measurements.
- The *likelihood* $P(\mathbf{x}|\mathbf{z})$ describes the likelihood of measurement **z** being generated by object **x** and follows from the measurement model. So if **z** favors **x**, the likelihood will be large.
- In the denominator the probability function P(**z**) is present to normalize the posterior distribution.

Multi sensor Bayesian Inference

In the DAVI-vehicle the state of an object is not estimated from one sensor, but from multiple sensors. In this situation it is still possible to use Bayes' Rule. So when considering $\mathbf{z}_1$ the measurements from the first sensor, $\mathbf{z}_2$ the observations from the second sensor and so on, the set of measurements becomes

$$\mathbf{Z}^n = \{\mathbf{z}_1, \mathbf{z}_1, ..., \mathbf{z}_n\} \tag{3.1.5}$$

And the state is now estimated given the measurement set, resulting in the new posterior distribution $P(\mathbf{x}|\mathbf{Z}^n)$ and corresponding Bayes' Rule [11]

$$\begin{aligned} P(\mathbf{x}|\mathbf{Z}^n) &= \frac{P(\mathbf{Z}^n|\mathbf{x})P(\mathbf{x})}{P(\mathbf{Z}^n)} \\ &= \frac{P(\mathbf{z}_1, ..., \mathbf{z}_n|\mathbf{x})P(\mathbf{x})}{P(\mathbf{z}_1, ..., \mathbf{z}_n)} \end{aligned} \tag{3.1.6}$$

This means that the joint distribution $P(\mathbf{z}_1, ..., \mathbf{z}_n|\mathbf{x})$ has to be known, which is not always the case in practice. However when assuming that the measurements from the different sensors are independent of each other, the likelihood simply changes to the product of the individual likelihoods of each sensor;

$$\begin{aligned} P(\mathbf{z}_1, ..., \mathbf{z}_n|\mathbf{x}) &= P(\mathbf{z}_1|\mathbf{x} \cdot \mathbf{z}_2|\mathbf{x}...\mathbf{z}_n|\mathbf{x}) \\ &= \prod_{i=1}^{n} P(\mathbf{z}_i|\mathbf{x}) \end{aligned} \tag{3.1.7}$$

Adding this into equation 3.1.6 gives the updated version of the Bayes Rule' known as *independent likelihood pool* [4].

$$P(\mathbf{x}|\mathbf{Z}^n) = \frac{P(\mathbf{x}) \cdot \prod_{i=1}^{n} P(\mathbf{z}_i|\mathbf{x})}{P(\mathbf{Z}^n)} \tag{3.1.8}$$

The validity of this equation relies totally on the assumption that the measurements from the different sensors are independent. However these measurements are generated by the same object state and are therefore by definition not independent. On the other hand, it is quite reasonable that the only common thing between the sensors is the common state. So when this state is given, the conditional probabilities given this state are independent of each other [11].

Recursive Bayesian Filter

When all the measurements are arrived, the Bayes' Rule estimates the corresponding estimated state. In autonomous driving the goal is not only to estimate the states at a certain time instant, but also to estimate the recursion or $track$ of an object in time. This requires that all the past information is remembered which may be undesirable. Instead of remembering all the past information, it is also possible to update Bayes' Rule in order to only recursive add the new information from the sensors. This is done by adding the new measurements at time t_k into the previous set of observations at t_{k-1}

$$\mathbf{Z}_k = \{\mathbf{z}_k, \mathbf{Z}_{k-1}\} \tag{3.1.9}$$

The chain-rule of condition probabilities tells us

$$\begin{aligned} P(\mathbf{x}, \mathbf{Z}_k) &= P(\mathbf{x}|\mathbf{Z}_k)P(\mathbf{Z}_k) \\ &= P(\mathbf{z}_k, \mathbf{Z}_{k-1}|\mathbf{x})P(\mathbf{x}) \\ &= P(\mathbf{z}_k|\mathbf{x})P(\mathbf{Z}_{k-1}|\mathbf{x})P(\mathbf{x}) \end{aligned} \tag{3.1.10}$$

When setting the first and last equation equal to each other and keeping in mind that $\frac{P(\mathbf{Z}_k)}{P(\mathbf{Z}_{k-1})} = P(\mathbf{z}_k|\mathbf{Z}_{k-1})$, the posterior distribution is obtained

$$P(\mathbf{x}|\mathbf{Z}_k) = \frac{P(\mathbf{z}_k|\mathbf{x})P(\mathbf{Z}_{k-1}|\mathbf{x})P(\mathbf{x})}{P(\mathbf{Z}_k)} = \frac{P(\mathbf{z}_k|\mathbf{x})P(\mathbf{x}|\mathbf{Z}_{k-1})}{P(\mathbf{z}_k|\mathbf{Z}_{k-1})} \tag{3.1.11}$$

This means that all the past information is completely summarized in the posterior likelihood $P(\mathbf{x}|\mathbf{Z}_{k-1})$ only. Since this is the only part to remember, a significant improvement in computational load is obtained compared to the Bayes rule in 3.1.8 where the likelihood of each individual sensor had to be remembered.

Filter recursion

The recursion of the Bayesian filter is done in two steps, a prediction and correction step. First a prediction of the state is made based on the previous measurements and the current state, or in other words the *posterior likelihood*

$$P(\mathbf{x}_k|\mathbf{Z}_{k-1}) = \int \mathbf{P}(\mathbf{x}_k|\mathbf{x}_{k-1}) \cdot \mathbf{P}(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})d\mathbf{x}_{k-1} \tag{3.1.12}$$

Subsequently after receiving a new set of measurements, the posterior likelihood and the new measurements are combined using the Bayes' rule of 3.1.11 resulting in the *posterior distribution*. Then the new likelihoods are calculated resulting in the cycle as shown in figure 3.1.

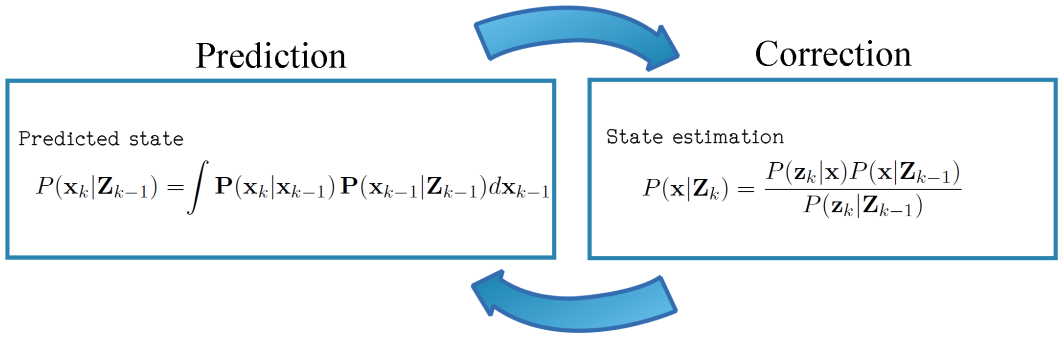

Prediction
Correction
Predicted state
$P(\mathbf{x}_k|\mathbf{Z}_{k-1}) = \int \mathbf{P}(\mathbf{x}_k|\mathbf{x}_{k-1})\mathbf{P}(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})d\mathbf{x}_{k-1}$
State estimation
$P(\mathbf{x}|\mathbf{Z}_k) = \frac{P(\mathbf{z}_k|\mathbf{x})P(\mathbf{x}|\mathbf{Z}_{k-1})}{P(\mathbf{z}_k|\mathbf{Z}_{k-1})}$

Figure 3.1: The recursion of the Bayesian filter

3.1.2 Kalman Filter

Another commonly used filter for estimating the state of an object is the *Kalman Filter*. This filter uses statistical measures to quantify the uncertainty of each sensor on the overall system performance. The higher the uncertainty of a sensor, the less it is taken into consideration in the estimation process of the state.

Instead of calculating the most likely value for the state, as is done in the Bayesian filter, the Kalman filter calculates the conditional mean of the state $\hat{\mathbf{x}} = E\{\mathbf{x}(t)|\mathbf{Z}^t\}$ and the corresponding variance $\mathbf{P}$. This is done in two steps; first a prediction of the posterior state is made based on the current state, and subsequently this posterior state is updated after receiving new measurements.

In the upcoming paragraph the linear variant of the Kalman filter is considered. There exist also non-linear variants, e.g. the extended and unscended Kalman filter, which are linearising about an estimate of the current mean and covariance.

Prediction step

In the prediction step, an estimation of the posterior state and its corresponding covariance is made based on the current state. This prediction is done with a predefined linear system model, which describes how the object is expected to move in time.

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k \mathbf{u} + \mathbf{G}_k \mathbf{v}_k \tag{3.1.13}$$

Where $\mathbf{F}_k$ is the state transition matrix, which describes the effect of each state parameter at time t_{k-1} on the expected state at time t_k. $\mathbf{B}_k$ and $\mathbf{u}$ are respectively the control input matrix and the input vector, which represent the effect of the input on the system. The vector $\mathbf{G}_k\mathbf{v}_k$ models the process noise for each parameter in the state vector. So when the objects in the environment are expected to move with constant velocity and without a control input, the linear system model over a time interval ΔT could for example be

$$\begin{bmatrix} x_k \\ \dot{x}_k \\ y_k \\ \dot{y}_k \end{bmatrix} = \begin{bmatrix} 1 & \Delta T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta T \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ \dot{x}_{k-1} \\ y_{k-1} \\ \dot{y}_{k-1} \end{bmatrix} + \begin{bmatrix} \Delta T^2/2 & 0 \\ \Delta T & 0 \\ 0 & \Delta T^2/2 \\ 0 & \Delta T \end{bmatrix} \begin{bmatrix} v_{x,k} \\ v_{y,k} \end{bmatrix} \tag{3.1.14}$$

The variance associated with the predicted state is given by

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{G}_k \mathbf{Q}_k \mathbf{G}_k^T \tag{3.1.15}$$

Where $\mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^T$ is the process noise covariance matrix associated with the noisy control input.

Update step

At time t_k the measurements are received by the sensors. As explained in section 2.2 these measurements include noise or an uncertainty. So the actual measurement is a combination of the object's state plus the noise, resulting in the following definition of the measurements;

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{D}_k \mathbf{w}_k \tag{3.1.16}$$

With $\mathbf{H}_k$ the measurement model and $\mathbf{D}_k$ the measurement noise matrix. For the example of the constant velocity model this results in

$$\begin{bmatrix} z_x \\ z_y \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ \dot{x}_{k-1} \\ y_{k-1} \\ \dot{y}_{k-1} \end{bmatrix} + \begin{bmatrix} w_{x,k} \\ w_{y,k} \end{bmatrix} \tag{3.1.17}$$

Using the received measurements, the state prediction $\hat{\mathbf{x}}_{k|k-1}$ and covariance $\mathbf{P}_{k|k-1}$ are updated. This is done according to

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}) \tag{3.1.18}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_{k|k-1} \tag{3.1.19}$$

Where $\mathbf{K}_k$ is the Kalman gain. This Kalman gain defines how much the predicted state should be updated by the values of the measurements. So in the case the measurements are highly reliable, the Kalman gain will be high and the state estimate will highly rely on the measurements. And if the measurements are highly uncertain, the gain will be low and the state estimate will mostly be based on the prediction. The Kalman gain is calculated using the covariances according to

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T \cdot \left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k\right)^{-1} \tag{3.1.20}$$

Filter recursion

The recursion of the Kalman filter is shown in figure 3.2. So first a prediction is made and after receiving the new measurements the prediction is updated. Subsequently a new prediction is made for the next time step and so on.

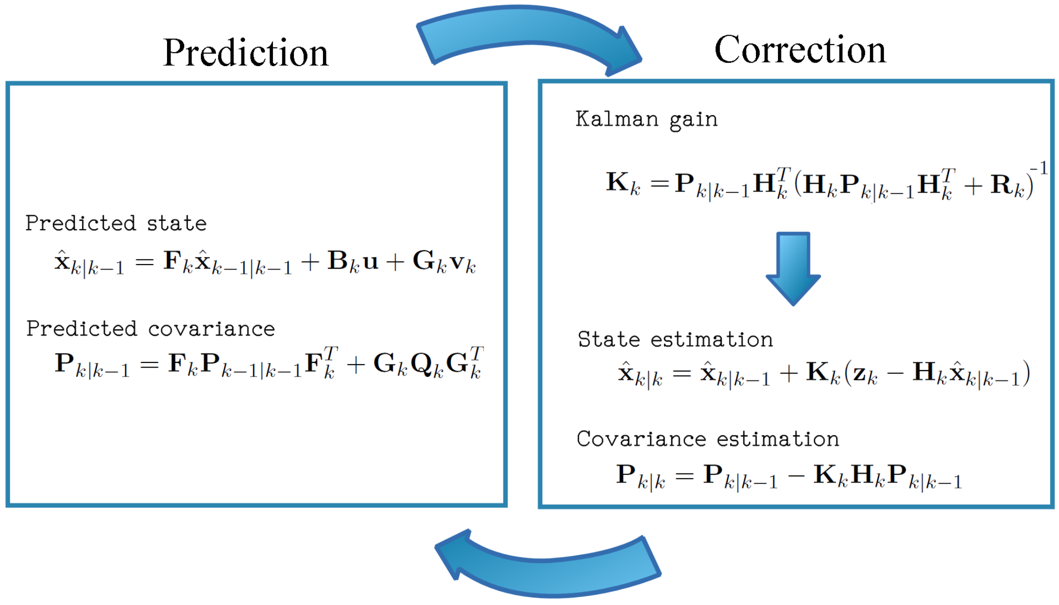

Prediction
Predicted state
$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k\mathbf{u} + \mathbf{G}_k\mathbf{v}_k$
Predicted covariance
$\mathbf{P}_{k|k-1} = \mathbf{F}_k\mathbf{P}_{k-1|k-1}\mathbf{F}_k^T + \mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^T$
Correction
Kalman gain
$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k)^{-1}$
State estimation
$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1})$
Covariance estimation
$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_{k|k-1}$

Figure 3.2: The recursion of the Kalman filter

3.1.3 Alpha-Beta Filter

When the noise matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$ and the transition and observation matrices $\mathbf{F}_k$ and $\mathbf{H}_k$ are time invariant, the gain matrices tend to a constant steady state value. In that case a simplified variant of the Kalman filter may be used. This variant is called the Alpha-Beta filter and uses constant gains in the update step.

Prediction step The constant gains have no influence in the prediction, so the state prediction is the same as the Kalman filter

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1|k-1} + \mathbf{G}_k \mathbf{v}_k \tag{3.1.21}$$

Update step However since the Kalman gain is constant, it is no longer necessary to calculate the gain in the recursion. Instead the Kalman gain **K** is substituted by predetermined constant gains α and β, resulting in the simplified updated state estimate

$$\hat{\mathbf{x}}_{k|k-1} = \hat{\mathbf{x}}_{k|k} + \begin{bmatrix}\alpha\\ \beta\end{bmatrix} [\mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1}] \tag{3.1.22}$$

Filter recursion For both the prediction as the update step, the conditional mean of the estimated state has to be calculated only. It is no longer necessary to calculate the variances and gains as well. This results in the recursion shown in figure 3.3.

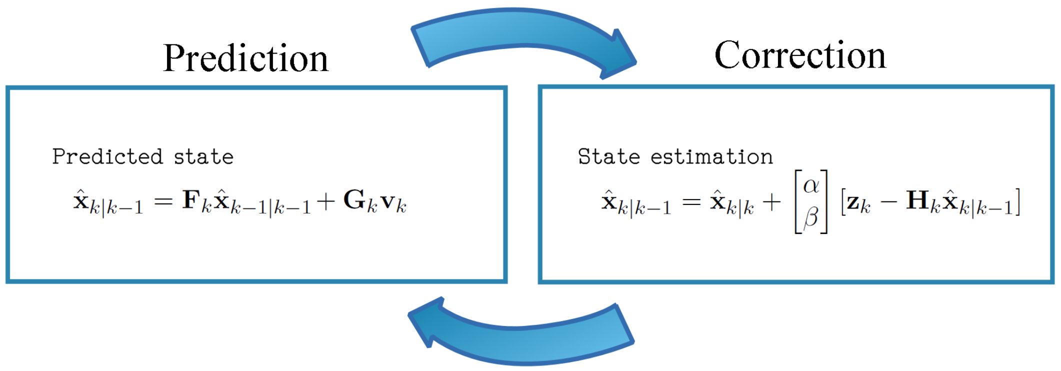

Prediction
Correction
Predicted state
$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1|k-1} + \mathbf{G}_k \mathbf{v}_k$
State estimation
$\hat{\mathbf{x}}_{k|k-1} = \hat{\mathbf{x}}_{k|k} + \begin{bmatrix}\alpha\\ \beta\end{bmatrix} [\mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1}]$

Figure 3.3: The recursion of the Alpha-Beta filter

The advantage of the Alpha-Beta filter compared to the original Kalman-filter is that the gains and variance no longer have to be calculated for each time step. This results in a reduction of the computational load. However this is only possible in systems where the assumption of a constant system is correct.

3.1.4 Multi Sensor Data Fusion

In practise, object tracking is often done with multiple sensors. The use of multiple sensors gives the possibility to cover a larger FOV and different types of sensors often have different advantages and disadvantages. The fusion of the data from multiple sensors allows the system to obtain a more accurate and reliable state estimation, than when only one sensor would be observed [12].

Synchronous Data Fusion

The sensors are discrete systems, outputting their observations at a specific sample rate and latency. In the ideal situation all the overlapping sensors have the same sample rate and latency, and all the data is received at exactly the same time as shown in figure 3.4.

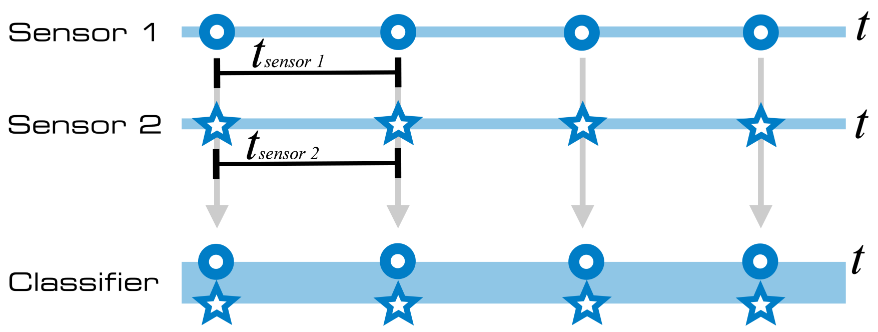


Figure 3.4: In the ideal situation all the overlapping sensors have the same sample rate and latency, and all the data is received at exactly the same time

In that case the classification of objects and subsequently the estimation of their corresponding states can simply be made using the observations of both sensors together as shown in figure 3.5.

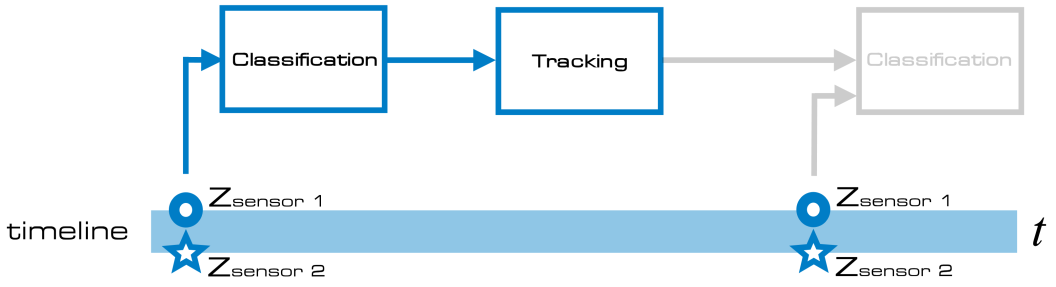


Figure 3.5: Both sensors are used to classify the objects and estimate their corresponding states

Asynchronous Data Fusion

The synchronous data fusion is more of a theoretical matter, because in reality the different sensors are almost never exactly synchronous. The different sensors are operating often at different sample rates and have different time instances. For example in the case when the sensor 1 has a much higher frequency with period $t_{sensor\ 1}$ compared to the frequency of sensor 2 with period $t_{sensor\ 2}$ as shown in figure 3.6.

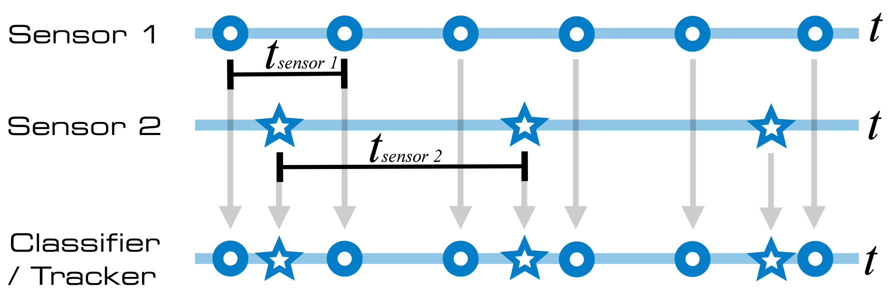


Figure 3.6: Different latencies and sample times of sensors results in data fusion issues

One way to deal with the different frequencies is to update the classification and track asynchronous as shown in figure 3.7. First an estimation of the classification and track is made after receiving data from sensor 1. Subsequently the classification and track are updated after receiving the data from the other sensor 2.

Another possibility when the asynchronicity is only caused by a difference in frequency of both sensors, is to align the data of both sensors. This is done by making an estimation of the high data rate sensor, in this case sensor 1, at the time the measurement of sensor 2 is received. This estimation could for example be done by obtaining a least-squares estimate of the high data rate sensor data at the time the other sensor observation is taken [5]. Subsequently the approach of the synchronous case can be used to fuse the time aligned data.

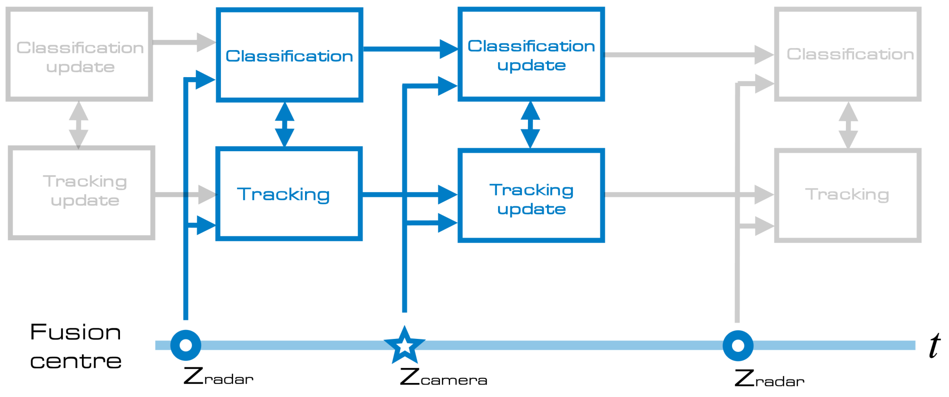


Figure 3.7: Sequential updating of data from an asynchronous sensor couple

3.1.5 Asynchronous Kalman Filter

The previous filters are based on synchronous sensor data. However in practice the data from different sensors arrive often asynchronous as explained in more detail in the previous paragraph. In the case of asynchronous data the Kalman filter is slightly different. First an estimation is made with the sensor data that is available up to time step t_{k-1}.

Prediction at t_{k-1}

$$\begin{aligned} \hat{\mathbf{x}}_{k-1|k-2} &= \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-2|k-2} + \mathbf{B}_{k-1}\mathbf{u} + \mathbf{G}_{k-1}\mathbf{v}_{k-1} \\ \mathbf{P}_{k-1|k-2} &= \mathbf{F}_{k-1}\mathbf{P}_{k-2|k-2}\mathbf{F}_{k-1}^T + \mathbf{G}_{k-1}\mathbf{Q}_{k-1}\mathbf{G}_{k-1}^T \end{aligned} \tag{3.1.23}$$

Update at t_{k-1}

$$\begin{aligned} \mathbf{K}_{k-1} &= \mathbf{P}_{k-1|k-2}\mathbf{H}_{k-1}^T \cdot \left(\mathbf{H}_{k-1}\mathbf{P}_{k-1|k-2}\mathbf{H}_{k-1}^T + \mathbf{R}_{k-1}\right)^{-1} \\ \hat{\mathbf{x}}_{k-1|k-1} &= \hat{\mathbf{x}}_{k-1|k-2} + \mathbf{K}_{k-1}(\mathbf{z}_{k-1} - \mathbf{H}_{k-1}\hat{\mathbf{x}}_{k-1|k-2}) \\ \mathbf{P}_{k-1|k-1} &= \mathbf{P}_{k-1|k-2} - \mathbf{K}_{k-1}\mathbf{H}_{k-1}\mathbf{P}_{k-1|k-2} \end{aligned} \tag{3.1.24}$$

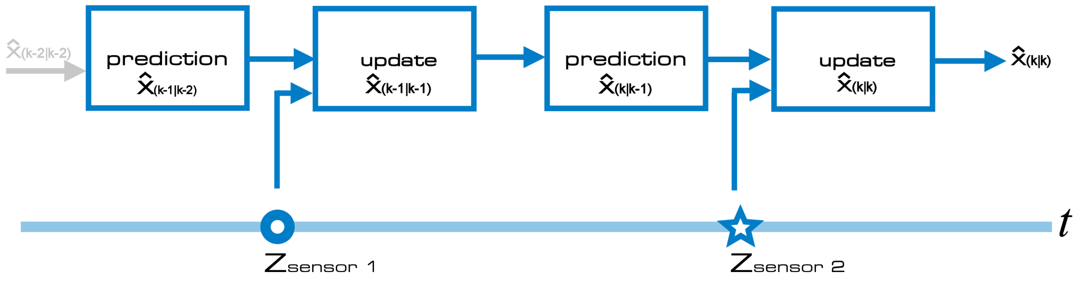


Figure 3.8: The asynchronous Kalman completes the cycle twice, first with the observations from sensor 1 and subsequently with the observations from sensor 2

Subsequently a new prediction is made after receiving the measurements from the other sensor at t_k. So basically the Kalman filter has been run twice; first after receiving the measurements from the first sensor and subsequently after receiving the data from the other sensor. An overview of this time course is shown in figure 3.8.

Prediction at t_k

$$\begin{aligned} \hat{\mathbf{x}}_{k|k-1} &= \mathbf{F}_k\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k\mathbf{u} + \mathbf{G}_k\mathbf{v}_k \\ \mathbf{P}_{k|k-1} &= \mathbf{F}_k\mathbf{P}_{k-1|k-1}\mathbf{F}_k^T + \mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^T \end{aligned} \tag{3.1.25}$$

Update at t_k

$$\begin{aligned} \mathbf{K}_k &= \mathbf{P}_{k|k-1}\mathbf{H}_k^T \cdot \left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k\right)^{-1} \\ \hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}) \\ \mathbf{P}_{k|k} &= \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_{k|k-1} \end{aligned} \tag{3.1.26}$$

The recursion of the asynchronous Kalman filter is shown in figure 3.9

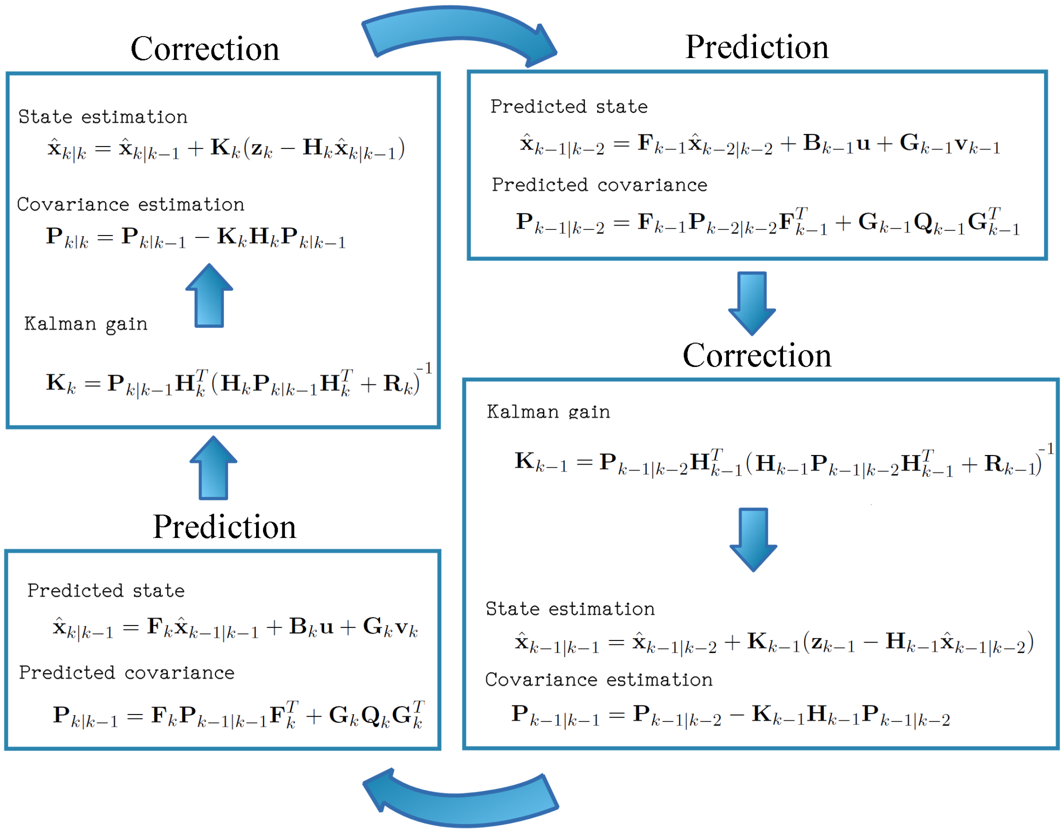

Correction
State estimation
$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1})$
Covariance estimation
$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_{k|k-1}$
Kalman gain
$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k)^{-1}$
Prediction
Predicted state
$\hat{\mathbf{x}}_{k-1|k-2} = \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-2|k-2} + \mathbf{B}_{k-1}\mathbf{u} + \mathbf{G}_{k-1}\mathbf{v}_{k-1}$
Predicted covariance
$\mathbf{P}_{k-1|k-2} = \mathbf{F}_{k-1}\mathbf{P}_{k-2|k-2}\mathbf{F}_{k-1}^T + \mathbf{G}_{k-1}\mathbf{Q}_{k-1}\mathbf{G}_{k-1}^T$
Correction
Kalman gain
$\mathbf{K}_{k-1} = \mathbf{P}_{k-1|k-2}\mathbf{H}_{k-1}^T(\mathbf{H}_{k-1}\mathbf{P}_{k-1|k-2}\mathbf{H}_{k-1}^T + \mathbf{R}_{k-1})^{-1}$
State estimation
$\hat{\mathbf{x}}_{k-1|k-1} = \hat{\mathbf{x}}_{k-1|k-2} + \mathbf{K}_{k-1}(\mathbf{z}_{k-1} - \mathbf{H}_{k-1}\hat{\mathbf{x}}_{k-1|k-2})$
Covariance estimation
$\mathbf{P}_{k-1|k-1} = \mathbf{P}_{k-1|k-2} - \mathbf{K}_{k-1}\mathbf{H}_{k-1}\mathbf{P}_{k-1|k-2}$
Prediction
Predicted state
$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k\mathbf{u} + \mathbf{G}_k\mathbf{v}_k$
Predicted covariance
$\mathbf{P}_{k|k-1} = \mathbf{F}_k\mathbf{P}_{k-1|k-1}\mathbf{F}_k^T + \mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^T$

Figure 3.9: Filter recursion of the asynchronous Kalman filter

3.2 Multi-Object Tracking

In the previous section the focus was on estimating the state of one single object. However in a traffic situation on the road there may be multiple cars, cyclists and pedestrians in the surrounding at the same time. One way for tracking these multiple objects could be by running a series of Kalman filters in parallel. However sometimes it is not clear which measurement from the sensor belongs to which object. So that means the problem is no longer the estimation of the states only, but also the association of the measurements with the objects. Besides the sensors may have missed detections and or false alarms, resulting in an uncertainty in the actual number of objects in the surrounding. Three common used multi-object tracking algorithms are the Nearest Neighbour, Probabilistic Data Association and the Multi-Hypothesis Tracking algorithm.

3.2.1 Nearest Neighbour Filter

The Nearest Neighbour (NN) filter does the association of measurements to the object by assuming that the measurement "nearest" to the prediction is the correct measurement. The nearest measurement is the measurement with the smallest Euclidean or Mahalanobis distance to each object. All the other measurements are subsequently assumed to be false alarm and rejected from consideration [11].

The NN approach is practically used in many applications and usually adequate in situations with high probability of detection and low rate of clutter [11]. However the association is irreversible. So when one association step goes wrong, the association of the next time step will be done incorrect, resulting in a rapidly decrease in performance of the algorithm. The result is that in situations with a high rate of clutter and or low probability of detection the performance will decrease rapidly [65]. Besides the algorithm is based on the assumption that a measurement is generated by one object. For some sensors an object is described by multiple measurements, resulting in an extra preprocessing step instead of using the raw data straight from the sensors.

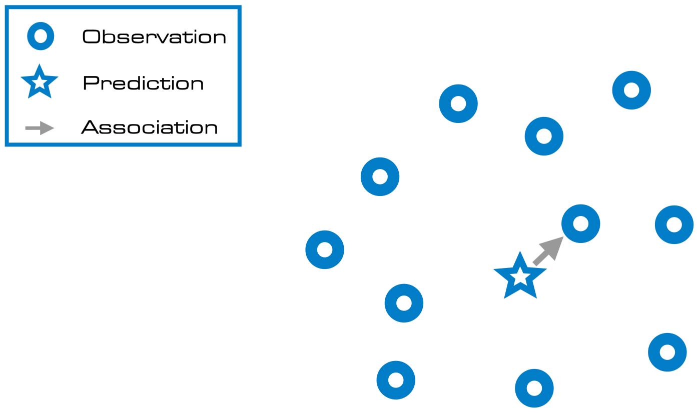


Figure 3.10: The Nearest Neighbour filter assumes that the observation the observations "nearest" to the prediction is the correct observation to associate with the object and subsequently rejects all other observations

3.2.2 Probabilistic Data Association

In situations with a high rate of clutter or false alarms, the Probabilistic Data Association filter (PDA) or Joint Probabilistic Data Association could be used to increase performance [3][12]. Instead of using only the measurement closest to the prediction, these filters are considering the measurements within a certain "confidence ellipsoid" [3]. Then for each of these measurement, a likelihood of being generated by the object is calculated and subsequently these likelihoods are used to approximate the state by a weighted average as shown in figure 3.11.

So when denoting all the measurements m at time k within this confidence ellipse as the set

$$\mathbf{Z}_k = \{z_1, ..., z_m\} \tag{3.2.1}$$

When defining the event $\chi_{k,i}$ that $z_{k,i}$ is generated from the object and $\chi_{k,0}$ the event that none of the measurements is generated from the object. Then association likelihoods $\beta^{i,j}$ for each object i and each measurement j within the confidence ellipsoid becomes

$$\beta_{k,i} = p(\chi_{k,i}|Z_k) \tag{3.2.2}$$

Using the association likelihoods, the approximate conditional mean of the state at time k is obtained by

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + \mathbf{K}_k v_k \tag{3.2.3}$$

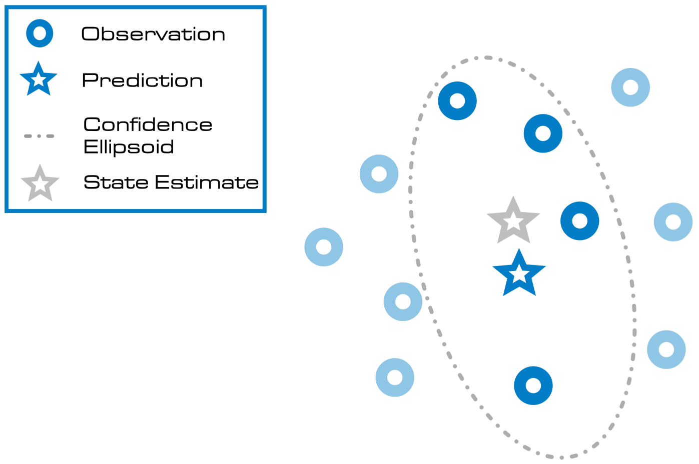


Figure 3.11: The PDA filter only considers the measurements within a certain "confidence ellipsoid" and subsequently takes an weighted average of these measurements to estimate the state

with $\mathbf{K}_k$ the Kalman gain and here v_k the weighted innovation calculated by

$$v_k = \sum_{j=0}^{m} \beta_{k,i}(z_{k,i} - \hat{z}_{k|k-1}) \tag{3.2.4}$$

The covariance associated with the state estimate at time k is a summation of the covariance of the update if there is only one measurement and the covariance at t_k

$$\mathbf{P}_{k|k} = \mathbf{P}^0_{k|k} + \mathbf{P}_k \tag{3.2.5}$$

- $\mathbf{P}^0_{k|k}$ the covariance in case of only one measurement
- $\mathbf{P}_k = \mathbf{K}_k \left(\sum_{j=0}^{m} \beta_{k,i}(z_{k,i} - \hat{z}_{k|k-1})(z_{k,i} - \hat{z}_{k|k-1})' - v_k c'_k \right) \mathbf{K}'_k$

This results in the recursion shown in figure 3.12.

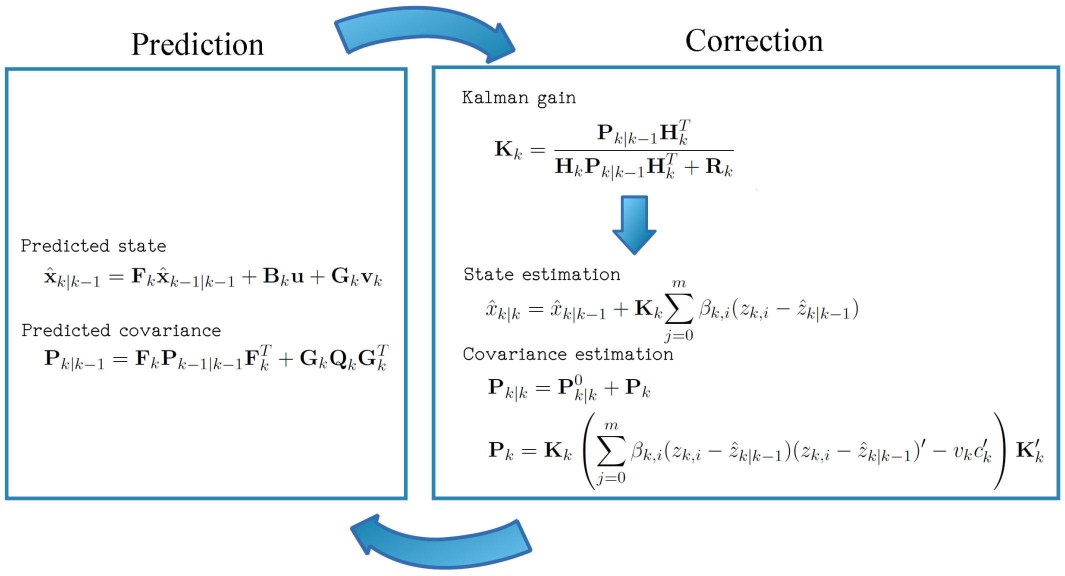

Prediction
Correction
Predicted state
$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k\mathbf{u} + \mathbf{G}_k\mathbf{v}_k$
Predicted covariance
$\mathbf{P}_{k|k-1} = \mathbf{F}_k\mathbf{P}_{k-1|k-1}\mathbf{F}_k^T + \mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^T$
Kalman gain
$\mathbf{K}_k = \frac{\mathbf{P}_{k|k-1}\mathbf{H}_k^T}{\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k}$
State estimation
$\hat{x}_{k|k} = \hat{x}_{k|k-1} + \mathbf{K}_k\sum_{j=0}^{m}\beta_{k,i}(z_{k,i} - \hat{z}_{k|k-1})$
Covariance estimation
$\mathbf{P}_{k|k} = \mathbf{P}^0_{k|k} + \mathbf{P}_k$
$\mathbf{P}_k = \mathbf{K}_k\left(\sum_{j=0}^{m}\beta_{k,i}(z_{k,i} - \hat{z}_{k|k-1})(z_{k,i} - \hat{z}_{k|k-1})' - v_k c_k'\right)\mathbf{K}_k'$

Figure 3.12: Filter recursion of the PDA filter

The consideration of more measurements around the object makes it possible to obtain a better estimate in situations with high clutter. However this comes at the cost of an increased estimation error covariance [65]. In addition, the performance of the PDA is decreasing in situations of overlapping confidence ellipsoids. I.e. when objects are closely packed of crossing each other. In these cases it is possible to have an observation which is located in both gates, resulting in an uncertainty from which object it is originated. The JPDA takes into account these situations and estimates the state using all possible association hypothesis [65].

3.2.3 Multi-Hypothesis Tracking

The states in the NN, PDA and JPDA filters are estimated using the previous time step only. That means that if the state in one time step is estimated incorrect, the performance of the filter decreases rapidly. The Multi-Hypothesis Tracking (MHT) filter prevents this by maintaining all the tracks for each possible associated measurement in the past [64]. Therefore a mismatch at a certain time step can be corrected at a later stage. Where the other filters are looking for the measurement that fits their current track best, searches the MHT for the track that fits all the measurements from the past best.

The MHT filter starts with creating a ”confidence ellipsoid” around the prediction, just like the PDA and JPDA filter. Subsequently all the measurements within this confidence ellipsoid are considered as candidate associations. For every existing track, a new track is created per candidate association. For each of these tracks the likelihood is then calculated, resulting in a ”hypothesis tree”. The track with the highest likelihood is finally considered as the correct track.

A downside of the MHT filter is the exponential increase of he number of hypotheses in time, resulting in an substantial increase in required memory and computational power [54]. To reduce this a pruning step is added, which discards the unlikely tracks. The exponential increasing hypotheses in time makes the MHT filter less attractive in environments with high amount of clutter, due to the fact that the clutter creates a lot of hypotheses. However in environments with high track uncertainty, for example due to crossing tracks, maneuvering objects, etc., the MHT filter performs generally good. So the MHT performs generally good in situations with low amount of clutter and high track uncertainty [11].

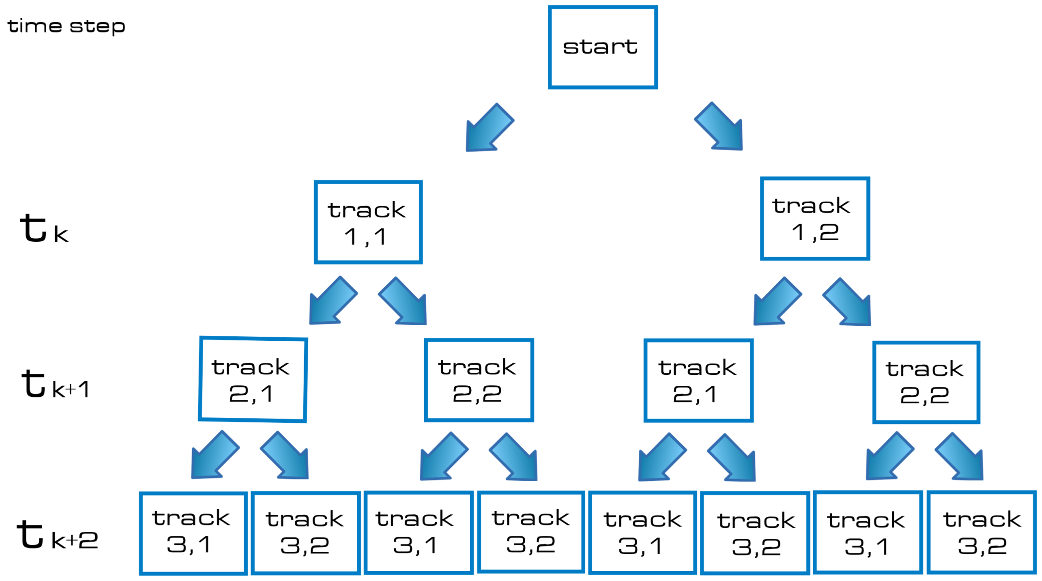


Figure 3.13: An example representation of an hypothesis tree of the MHT over three time steps for two objects. The number of hypothesis increases exponentially in time, like in this example where there are already eight hypothesis after only three time steps with two objects

3.3 Random Finite Set Statistics

In the multi-object tracking approaches discussed in the previous section, the estimation of the number of objects and the state estimation is done separately. Also the individual association of each object with each measurement results in a high level of computation load, which exponentially increases with the number of objects [62]. This makes these approaches no longer feasible for real-time scenarios when the number of objects increases. Therefore an alternative formulation was introduced by Mahler et al. which avoids the explicit association between measurements and objects using Random Finite Set Statistics (FISST) [48][27][42].

The basic idea behind Random Finite Set Statistics is to define the entire collection of individual measurements and states as one *set-valued measurement* and one *set-valued state* [74]. This gives the possibility to use a single object Bayes filter for the multi-object tracking problem and thereby avoids the issues concerned with the unknown number of objects [48][59][71].

Random Finite Set principles

A Random Finite Set (RFS) is a set of variables where both the number of elements and the order of these elements are random. By defining all the state-vectors as the elements in an RFS, the *set-valued state* X is obtained.

$$X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\} \tag{3.3.1}$$

With n the number of objects and $\mathbf{x}_1$,..,$\mathbf{x}_n$ the state vectors of the individual objects. The represented states have no physically inherent order and the number of objects n is variable, which makes these sets fundamentally different from vector representations. The random order and the variable number of objects gives the possibility to deal with the unknown number of objects [74].

The same principle can be applied to the measurements, resulting in the set-valued measurement

$$Z = \{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_m\} \tag{3.3.2}$$

With m the amount of measurements from multiple sensors with values $\mathbf{z}_1$,..,$\mathbf{z}_m$, whose randomness in m is caused by false alarms and missed detections. The definition of the states and measurement in an RFS gives the possibility to use the single object Bayes filter framework in a multi-object environment, resulting in the Multi-Object Bayes filter [48][59][71].

3.3.1 Multi-Object Bayes Filter

In the single object Bayes filter in section 3.1.1 the state of an object was estimated in two steps. First the probability of each measurement given the object' state $P(\mathbf{x}|\mathbf{z})$ was estimated and subsequently the highest probability was assumed to be the object.

The same principle is applied in the Multi-Object Bayes filter, but now using Random Finite Sets. First a prediction is made for the probability that the objects will have state-set X at the time of the next measurements. This prediction is based on the estimated probability of the previous time step $p_{k|k-1}$ and the expected transition of this estimated probability expressed in the Markov transition probability $f(X_k|X_{k-1})$. Using the Chapman-Kolmogorov equation the predicted prior probability set becomes [48][65]

$$P_{k|k-1}(X_k|Z_{k-1}) = \int P_{k-1}(X_{k-1}|Z_{k-1})f(X_k|X_{k-1})\delta X_{k-1} \tag{3.3.3}$$

Subsequently after receiving the measurements from the sensors, the prior probability is updated resulting in the new posterior probability set at time t_k. This is done using the single object Bayes' rule of 3.1.1 extended with the set-valued state X and set-valued measurement Z [48].

$$P_k(X_k|Z_k) = \frac{L_z(Z_k|X_k) \cdot P_{k|k-1}(X_k|Z_{k-1})}{\int L_z(X_k) \cdot P_{k|k-1}(X_k|Z_{k-1})\delta X_k} \tag{3.3.4}$$

Where $L_z(Z_k|X_k)$ is the multi object likelihood in which the likelihood that the objects in X are generated by the measurements in Z is expressed. The integral $\int L_z(X_k) \cdot P_{k|k-1}(X_k|Z_{k-1})\delta X_k$ is a *set integral*, summing over all possible numbers of objects. At last the Bayes-optimal multi object state estimator is used to extract the states from this posterior probability.

Implementations of the Multi Object Bayes Filter

However due to the multiple of integrals to solve when calculating the prior probability set and posterior probability set, the Multi-Object Bayes filter is computationally intractable [48][41][75]. Sequential Monte Carlo implementations are successfully applied for a small number of objects [39], for example in Reuter et al. for tracking pedestrians using two laser range finders [66] [68] [67]. These methods are however still computationally intensive for a large numbers of objects due to the combinatorial nature of the probabilities [75][76].

3.3.2 Probability Hypothesis Density Filter

The Multi Object Bayes Filter is computationally intractable when the number of objects increases, due to the increasing number of integrals [74]. To deal with this computationally intractability, the Probability Hypothesis Density filter (PHD) has been developed. Instead of calculating the entire posterior probability density per object, the PHD filter makes an approximation by calculating the first-order statistical moment of the posterior density [47]. In case of only a single object, the first-order statistical moment of the posterior probability would be the expected state of the object. For a multi-object scenario, with multi-object posterior density P for an RFS X, the first order statistical moment over the region S is a function v called the *intensity* or *probability hypothesis density*.

$$\int |X \cap S| P(dX) = \int_S v(x)dx \tag{3.3.5}$$

An example of an *probability hypothesis density* for the Gaussian Mixture implementation explained in section 3.3.3 is shown in figure 3.14. Each appropriate object is visualized by a Gaussian function and the height of the peak gives the probability of the object being on that location in the number of objects per square meter.

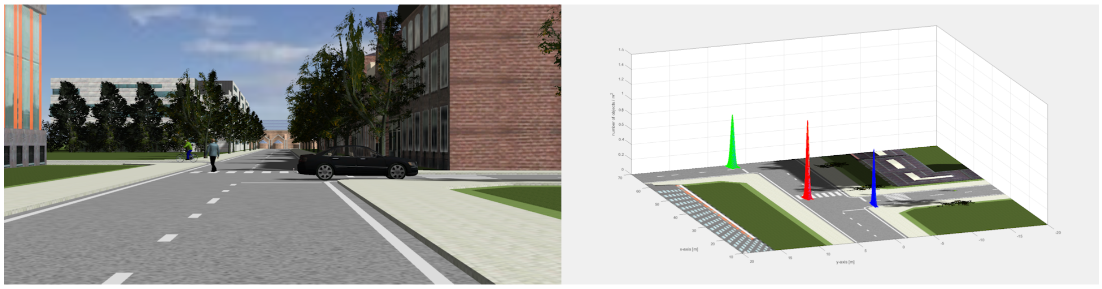


Figure 3.14: A visualization of the probability hypothesis density for a simulated traffic situation. Appropriate objects are represented by Gaussian Mixture components and the height of the peak gives the likelihood of each object.

The recursion of the PHD filter consist of three consecutive steps. Just like the Multi-Object Bayes' filter, the PHD filter starts with a prediction step, subsequently an update step after receiving the measurements and finally a state extraction step.

Prediction step

In the prediction step, a prediction is made what the PHD will be at the time the new measurements arrive from the sensor. This prediction is based on the estimated PHD of the previous time step $v_{k-1}(x)$ and a corresponding transformation model $f_{k|k-1}(x)$. However there is also a probability that an object terminates in the upcoming time step. To facilitate this possibility, the prediction PHD is factorized by a predefined survival probability P_s. Besides there is also a possibility that new objects appear in the upcoming time step, e.g. due to objects entering the FOV or objects spawned from existing objects. To deal with the changing number of objects as a consequence of new appearing objects, an extra PHD for birth objects and spawned objects are added to the predicted PHD.

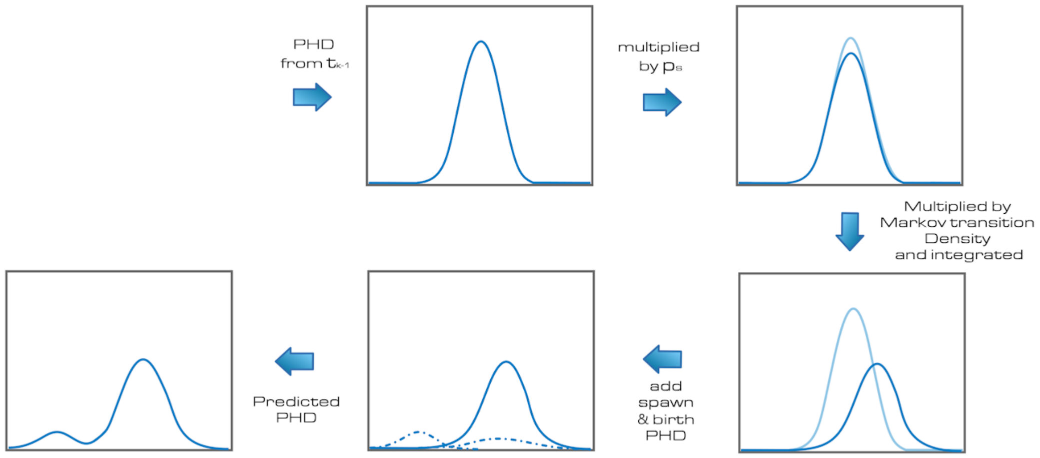


Figure 3.15: Schematic overview of the construction of the predicted PHD

The sum of the PHD of survived objects, together with the PHD of birth objects and spawned objects results in the equation of the predicted PHD [74].

$$\underbrace{v_{k|k-1}(x)}_{\text{predicted PHD}} = \underbrace{\gamma_k(x)}_{\text{PHD of birth objects}} + \underbrace{\int p_{s,k}(x) \cdot f_{k|k-1}(x) \cdot v_{k-1}(x)dx}_{\text{PHD of survived objects}} + \underbrace{\int \beta_{k|k-1}(x) \cdot v_{k-1}(x)dx}_{\text{PHD of spawned objects}} \tag{3.3.6}$$

with

- $\gamma_k(x)$ represents the birth PHD. The construction of this birth PHD is dependent on its application. One way could be to construct this PHD from residual measurements that are most likely not originating from the existing objects. The used birth model in this thesis is explained in depth in section 4.1.3.
- $p_{s,k}(x)$ is the probability of an existing object to survive.
- $f_{k|k-1}(x)$ is the state transition model of the PHD. This model predicts for each object's state x_{k-1} at time t_{k-1} the expected state at x_k at time t_k.

- $\beta_{k|k-1}(x)$ represents the probability that a object will be spawned from object x_k. In PHD filters applied in automotive this part of the equation is often omitted since it is unlikely to occur in most automotive scenarios [40].
- $v_x(\zeta)$ is the estimated PHD from the previous time step t_{k-1}

Update step

In the update step, the predicted PHD is updated with the information resulting from the measurements. The state of every object is compared to each measurement, resulting in a likelihood for each measurement and object $p_k(z|x)$. This likelihood is dependent on the application and could for example be based on similarity in location of measurement and object or in similarity of both classes.

Every object in the predicted PHD is multiplied by its likelihood and a detection probability p_d. This detection probability takes into account the possibility that an object is not detected by a sensor. The result is normalized by the sum of all objects, together with a correction factor $\kappa_k(z)$ for the amount of clutter. This results in the updated prediction PHD. Besides the predicted PHD, factorized by the probability of not detected $(1-P_d)$, is added to this PHD in order to propagate the missed objects by the sensor.

The sum of the missed detections PHD and the updated prediction PHD results in the equation of the updated PHD [74]

$$\underbrace{v_k(x)}_{\text{updated PHD}} = \underbrace{[1-p_{D,k}(x)]v_{k|k-1}(x)}_{\text{PHD of missed detection}} + \underbrace{\sum_{z\in Z_k} \frac{p_{D,k}(x)\cdot p_k(z|x)\cdot v_{k|k-1}(x)}{\kappa_k(z)+\int p_{D,k}(x)\cdot p_k(z|x)\cdot v_{k|k-1}(x)\delta x}}_{\text{update of predicted PHD}} \tag{3.3.7}$$

With the elements:

- $p_D(x)$ the probability of the object x being detected by the sensor
- $\kappa_k(z)$ the intensity of the clutter
- p_k the likelihood function.

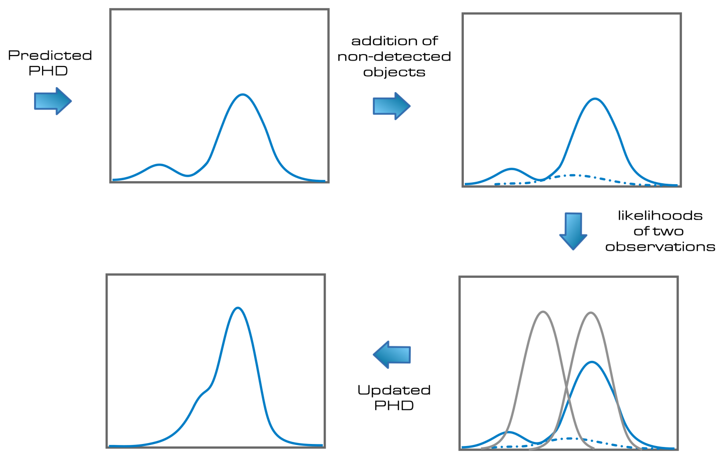

Predicted PHD
addition of non-detected objects
likelihoods of two observations
Updated PHD

Figure 3.16: Schematic overview of the construction of the updated PHD

Number of objects and state extraction

The objects in the PHD are represented in the PHD by peaks with its likelihood in number of objects per square meter. So the integral of the PHD over a specific region S gives the expected number of objects in that region S [74].

$$\int_S v_k(x)dx = \text{expected number of objects in S} \tag{3.3.8}$$

Consequently by taking the integral of the PHD over the entire space of state-set X, the expected number of objects $\hat{N}$ in X is obtained. This number of objects $\hat{N}$ is also called the *cardinality*. Subsequently the peaks with the highest local concentration of the expected cardinality of the PHD, the $\hat{N}$ highest peaks, are taken as the estimated states [74].

The major drawback of estimating the cardinality in this way, is that the estimation is only based on the mean of the cardinality distribution. The estimation by a single parameter makes the cardinality distribution effectively a Poisson distribution, where the mean and variance are equal [78]. So if the number of objects increases, the variance in the cardinality also increases, resulting in a large fluctuation in the cardinality between time steps [78]. One option to get a stable cardinality in practice is to average the number of objects over a time-window [46]. Another option is by estimating the variances in the cardinality as well, using the second-order statistical moment. This is done with the Cardinalized PHD (CPHD) explained in section 3.3.5, however this comes at the cost of a higher computational load [46][38][65].

Implementations of the PHD

The PHD recursions contains multiple integrals that have no closed-form solution in general. Therefore two closed-form PHD implementations are developed in general, the Sequential Monte Carlo implementation (SMC-PHD), sometimes called the particle filter, and the Gaussian Mixture implementation (GM-PHD). The Gaussian Mixture is less computationally demanding and provides a true closed-form algebraic solution [74]. However the GM-PHD works with the assumption that each object follows a linear-Gaussian motion and measurement model and works with the assumption that all objects motions are statistically independent. If this is not the case, the SMC-PHD could be used, which is able to deal with highly non-linear motion and measurement models. This comes however at the cost of a higher computational load [18].

The Matlab code for both SMC-(C)PHD and GM-(C)PHD has been implemented by Vo et al. [73]. The GM-CPHD implementation of this code is in this thesis adapted to the radar and camera setup and the stereo camera setup. Besides a GUI is developed to visualize the PHD in time and to browse through the tracking results of the filter in time. Besides this code is used as a basis for the implementations done in this thesis for MM-CPHD and CMM-PHD explained in more depth in respectively section 3.4.1 and 3.4.2

The PHD filter is also implemented multiple times in automotive application, for example by Garcia et al. (2014) for tracking vehicles using data from cameras attached to the ego-vehicle [21], Maehlisch et al. (2006) using a camera, lidar and ESP system attached to the ego-vehicle [40].

3.3.3 Gaussian Mixture Implementation of the PHD

In the Gaussian Mixture implementation (GM-PHD) every candidate object j is represented by a Gaussian with a mean $m^{(j)}$, covariance $P^{(j)}$ and weight $w^{(j)}$ representing the probability of each Gaussian. So the PHD at time t_{k-1} is mixture of Gaussians of the form

$$v_{k-1}(x) = \sum_{j=1}^{J_{k-1}} w_{k-1}^{(j)} \mathcal{N}(x; m_{k-1}^{(j)}, P_{k-1}^{(j)}) \tag{3.3.9}$$

Each of the objects follows a linear Gaussian motion model and the sensor has a linear Gaussian measurement model. So the GM-PHD is based on the assumption that both the motion model as well as the measurement model model are linear-Gaussian. When denoting the Gaussian density of an object as $\mathcal{N}(\cdot; m, P)$ with m the mean and P the covariance, the motion model and measurement model become:

$$\begin{aligned} f_{k|k-1}(x) &= \mathcal{N}(x; F_{k-1}x, Q_{k-1}) \\ g_k(z|x) &= \mathcal{N}(z; H_k x, R_k) \end{aligned} \tag{3.3.10}$$

With F_{k-1} is the state transition matrix, Q_{k-1} the process noise covariance, H_k the observation matrix and R_k the observation noise covariance. Another assumption of the GM-PHD is that the survival and detection probabilities are state independent and thus constant over time.

$$\begin{aligned} p_{s,k}(x) &= p_{s,k} \\ p_{d,k}(x) &= p_{d,k} \end{aligned} \tag{3.3.11}$$

Prediction step

When the new appearing birth PHD and spawned objects PHD are assumed to be linear Gaussian Mixtures, the sum of the surviving, birth and spawned PHD is a Gaussian Mixture too. This results in the adapted predicted PHD equation of 3.3.6 to the Gaussian mixture form.

$$\underbrace{v_{k|k-1}(x)}_{\text{predicted PHD}} = \underbrace{v_{S,k|k-1}(x)}_{\text{Survivors PHD}} + \underbrace{v_{\beta,k|k-1}(x)}_{\text{Spawned PHD}} + \underbrace{\gamma_k(x)}_{\text{Birth PHD}} \tag{3.3.12}$$

with the PHD of the survivors, spawned and birth objects

- Survivors PHD: $v_{S,k|k-1}(x) = p_{S,k} \sum_{j=1}^{J_{k|k-1}} w_{k-1}^{(j)} \mathcal{N}(x; m_{S,k|k-1}^{(j)}, P_{S,k|k-1}^{(j)})$

$$\begin{aligned} m_{S,k|k-1}^{(j)} &= F_{k-1} m_{k-1}^{(j)} \\ P_{S,k|k-1}^{(j)} &= Q_{k-1} + F_{k-1} P_{k-1}^{(j)} F_{k-1}^T \end{aligned}$$

- Spawned PHD: $v_{\beta,k|k-1}(x) = \sum_{j=1}^{J_{k|k-1}} \sum_{l=1}^{J_{\beta|k}} w_{k-1}^{(j)} w_{\beta,k}^{(l)} \mathcal{N}(x; m_{\beta,k|k-1}^{(j,l)}, P_{\beta,k|k-1}^{(j,l)})$

$$\begin{aligned} m_{\beta,k|k-1}^{(j,l)} &= F_{\beta,k-1}^{l} m_{k-1}^{(j)} + d_{\beta,k-1}^{(l)} \\ P_{\beta,k|k-1}^{(j,l)} &= Q_{\beta,k-1}^{(l)} + F_{\beta,k-1}^{(l)} P_{\beta,k-1}^{(j)} (F_{\beta,k-1}^{l})^T \end{aligned}$$

- Birth PHD: $\gamma_k(x) = \sum_{i=1}^{J_{\gamma,k}} w_{\gamma,k}^{(i)} \mathcal{N}(x; m_{\gamma,k}^{(i)}, P_{\gamma,k}^{(i)})$

Update step

The same principle counts for the correction step, but now using the linear-Gaussian measurement model and a constant detection probabilities. This results in the update equation of 3.3.7 in Gaussian Mixture form.

$$v_k(x) = \underbrace{(1 - p_{d,k}) v_{k|k-1}(x)}_{\text{PHD of missed detection}} + \underbrace{\sum_{z \in Z_k} \sum_{j=1}^{J_{k|k-1}} w_k^{(j)}(z) \mathcal{N}(x; m_k^{(j)}(z), P_k^{(j)})}_{\text{update of predicted PHD}} \tag{3.3.13}$$

with

$$\begin{aligned}
w_k^{(j)}(z) &= \frac{p_{d,k} w_{k|k-1}^{(j)} q_k^{(j)}(z)}{\kappa_k(z) + p_{d,k} \sum_{l=1}^{J_{k|k-1}} w_{k|k-1}^{(l)} q_k^{(l)}(z)} \\
q_k^{(j)}(z) &= \mathcal{N}(z; H_k m_{k|k-1}^{(j)}, R_k + H_k P_{k|k-1}^{(j)} H_k^T) \\
m_{k|k}^{(j)}(z) &= m_{k|k-1}^{(j)} + K_k^{(j)}(z - H_k m_{k|k-1}^{(j)}) \\
P_{k|k}^{(j)} &= [I - K_k^{(j)} H_k] P_{k|k-1}^{(j)} \\
K_k^{(j)} &= P_{k|k-1}^{(j)} H_k^T (H_k P_{k|k-1}^{(j)} H_k^T + R_k)^{-1}
\end{aligned} \tag{3.3.14}$$

So for the recursion of the mean and covariance as well as the likelihood of the GM and measurement, the Kalman equations are used.

Number of objects and states

As described in the previous chapter, the number of objects could be calculated by taking the integral of the posterior PHD. However since the posterior PHD is a gaussian mixture, an estimate of the cardinality is obtained by just summing up the appropriate weights [74]. Consequently the mean of the cardinality is

$$\hat{N}_{k|k-1} = \hat{N}_{k-1}(p_{s,k} + \sum_{j=1}^{J_{\beta,k}} w_{\beta,k}^{(j)}) + \sum_{j=1}^{J_{\gamma,k}} w_{\gamma,k}^{(j)} \tag{3.3.15}$$

and the mean of the updated cardinality is

$$\hat{N}_k = \hat{N}_{k|k-1}(1 - p_{d,k}) + \sum_{z \in Z_k} \sum_{j=1}^{J_{k,k-1}} w_k^{(j)}(z) \tag{3.3.16}$$

The states can be estimated by the peaks of the highest local concentrations in the posterior PHD v_k. In the Gaussian mixture representation this is straight forward, since the highest local concentrations are simply the GM components with the highest weights. The states can therefore be found by selecting the means of the $\hat{N}_k$ highest weight GM components.

Merging and Pruning

During each recursion of the filter, a GM is created for every predicted GM with each measurement and a corresponding weight is estimated for the likelihood of each GM component. Gaussian components with low weights will therefore survive in time, resulting in an exponential grow of components in time. To restrain the number of GM components, highly unlikely components are pruned and similar GM components are merged. This is done by merging all the GM components with a weight below a certain threshold and merging GM with similar states.

3.3.4 Sequential Monte Carlo Implementation of the PHD

The GM-PHD is only applicable in situation with a linear-Gaussian measurement and motion model. The Sequential Monte Carlo PHD (SMC-PHD) is however able to deal with non-linear measurement and motion models [13][75]. Instead of expressing a candidate object in the PHD as a linear-Gaussian distribution function, the distribution function is simulated by a number of samples. These samples are called particles and the likelihood of each particle being an object is expressed with a weight. The higher this weight, the more likely the object will be at the location of that particle.

When using the expression of candidate objects as numbers of samples, the PHD of N particles with each particle a weight $w^{(i)}$ and states $x^{(i)}$ will be simulated by the set [75]

$$\begin{aligned} v_k(x) &= \{w_k^{(i)}, x_k^{(i)}\}_{i=1}^{N_k} \\ &= \sum_{i=1}^{N_k} w_k^{(i)} \delta_{k-1}^{(i)}(x) \end{aligned} \tag{3.3.17}$$

Where $\delta_{k-1}^{(i)}(x)$ is a delta function centered at x.

Prediction step

The prediction step is based on the same principle as with the GM-implementation. However where in the GM-implementation a prediction is done for each candidate object described by a GM component, is in the SMC implementation a prediction made for each particle [75].

$$v_{k|k-1} = \sum_{i=1}^{N_{k-1}+J_k} w_{k|k-1}^{(i)} \delta_{x_k^{(i)}}(x_k) \tag{3.3.18}$$

Where the weights are calculated by

$$\begin{aligned} \underbrace{w_{k|k-1}^{(i)}}_{\text{predicted weight}} &= \underbrace{\frac{\gamma_k^{(i)}(x_k)}{J_k p_k(x_k^{(i)}|Z_k)}}_{\text{weight of birth objects}} + \underbrace{\frac{\sum_{i=1}^{N_{k-1}} w_{k-1}^{(i)} \cdot p_{s,k|k-1}(x_{k-1}) \cdot f_{k|k-1}(x_k|x_{k-1})}{q_k(x_k^{(i)}|x_{k-1}^{(i)}, Z_k)}}_{\text{weight of survived objects}} \\ &+ \underbrace{\frac{\sum_{i=1}^{N_{k-1}} w_{k-1}^{(i)} \cdot \beta_{k|k-1}(x_k|x_{k-1})}{q_k(x_k^{(i)}|x_{k-1}^{(i)}, Z_k)}}_{\text{weight of spawned objects}} \end{aligned} \tag{3.3.19}$$

J_k is the number particles describing the birth objects and q_k the previous number of particles.

Correction step

In the correction step the particle presentation of the predicted PHD is modified with new weights using the new observation set. The particle presentation of the posterior PHD is expressed as [75]

$$v_k = \sum_{i=1}^{N_{k-1}+J_k} w_k^{(i)} \delta_{x_k^{(i)}}(x_k) \tag{3.3.20}$$

where the weights are calculated according to

$$w_k^{(i)} = \underbrace{[1 - p_{D,k}(x)] \cdot w_{k|k-1}^{(i)}}_{\text{weights missed detections}} + \underbrace{\sum \frac{P_{D,k}(x) \cdot p_k(z|x_k) \cdot w_{k|k-1}^{(i)}}{\kappa_k(z) + \sum_{j=1}^{L_{k-1}+J_k} P_{D,k}(x) \cdot p_k(z|x_k) \cdot w_{k|k-1}^{(i)}}}_{\text{weight update prediction}} \tag{3.3.21}$$

Number of objects and resampling

The numbers of objects is approximated by the total sum of the weights:

$$L_k \cong \sum_{j=1}^{L_{k-1}+J-k} w_k^{(j)} \tag{3.3.22}$$

In every timestep the number of particles increases with $L_k = L_{k-1} + J_k$, even in the case where the number of objects does not increase. This results in exploring particles in a state space with no object present and is therefore computational intractable. However when the number of particles L_k is kept constant, then the ratio of particles to objects would fluctuate as the number of objects changes [75]. Therefore the particles are resampled by eliminating particles with low weights multiplying particles with high weights to keep the focus on exploring the important regions.

3.3.5 Cardinalized Probability Hypothesis Density Filter

The PHD is a computationally tractable, in contrast to the Multi Object Bayes filter, due to the approximation of the probability density by its first-order statistical moment. The cardinality in the PHD is estimated by the mean of the cardinality distribution only and is therefore effectively approximated by a Poisson distribution. Consequently the variance is equal to the cardinality itself, resulting in large fluctuations in the cardinality when the number of objects in the environment increases [78]. This phenomenon is also called the target death problem [24][14].

The CPHD addresses the target death problem by propagating the variance of the cardinality as well, using the second-order statistical moment [44]. This, however, also affects the prediction and update equations of the PHD since the intensity functions and the cardinality are thereby coupled. As a consequence the CPHD filter shows a dramatic reduction in the variance of the cardinality [77], however at the cost of a higher computational load [46][38][65].

The CPHD is proposed by Mahler [43][44] and a closed form Gaussian Mixture implementation is derived by Pasha et al. [60]. The prediction and the update step of the Gaussian Mixture implementation is discussed in the next paragraph.

Prediction step

The prediction of the PHD is equivalent to the prediction in the PHD filter, however in the original formulation of the CPHD spawning of other objects is no longer taken into account and the clutter is assumed to be generalized Poisson [77]. Lundgren et al. however implemented a generalization of the CPHD with an explicit spawning model [37]. In automotive application spawning occurs rarely and is in general not considered in automotive application [65]. Therefore in this thesis the original CPHD implementation without spawning is considered in this thesis, the equation for the PHD prediction becomes [78]

$$\underbrace{v_{k|k-1}(x)}_{\text{predicted PHD}} = \underbrace{v_{S,k|k-1}(x)}_{\text{Survivors PHD}} + \underbrace{\gamma_k(x)}_{\text{Birth PHD}} \tag{3.3.23}$$

with the PHD of the survivors, spawned and birth objects

- Survivors PHD: $v_{S,k|k-1}(x) = p_{S,k} \sum_{j=1}^{J_{k|k-1}} w_{k-1}^{(j)} \mathcal{N}(x; m_{S,k|k-1}^{(j)}, P_{S,k|k-1}^{(j)})$

$$m_{S,k|k-1}^{(j)} = F_{k-1} m_{k-1}^{(j)}$$
$$P_{S,k|k-1}^{(j)} = Q_{k-1} + F_{k-1} P_{k-1}^{(j)} F_{k-1}^T$$

- Birth PHD: $\gamma_k(x) = \sum_{i=1}^{J_{\gamma,k}} w_{\gamma,k}^{(i)} \mathcal{N}(x; m_{\gamma,k}^{(i)}, P_{\gamma,k}^{(i)})$

In the PHD filter only the PHD is propagated in time. However in the CPHD also the cardinality distribution $p_{k|k-1}(n)$ is propagated, with n the number of objects. The cardinality is in the prediction step dependent on the cardinality of the surviving objects and the cardinality of new appearing objects. However, for example a cardinality of 4, may be due to 1 birth object and 3 surviving objects. But also due to 2 birth objects and 2 surviving objects. The probability of 1 birth object and 3 surviving objects is the product of probability of 1 birth object and the probability of 3 surviving objects

$$p(\text{1 birth + 3 surviving objects}) = p(\text{1 birth object}) \cdot p(\text{3 surviving objects}) \tag{3.3.24}$$

So the probability of having exactly a cardinality n is estimated by the sum of all the "combination probabilities" that make exactly n objects. This results in the equation of the predicted cardinality distribution [78]:

$$\underbrace{p_{k|k-1}(n)}_{\text{predicted cardinality}} = \underbrace{\sum_{j=0}^{n}}_{\substack{\text{sum of all}\\ \text{combinations}\\ \text{that make n}}} \underbrace{p_{\Gamma,k}(n-j)}_{\substack{\text{probability of n-j}\\ \text{birth objects}}} \underbrace{\sum_{l=j}^{\infty} p_{k-1}(l) \binom{l}{j} p_{s,k}^{j}(1-p_{s,k})^{l-j}}_{\text{probability of j surviving objects}} \tag{3.3.25}$$

where

- $p_{\Gamma,k}(\cdot)$ is the cardinality distribution of births at time k
- $\binom{l}{j}$ is the binomial coefficient with $\binom{l}{j} = \frac{l!}{j!(l-j)!}$

Update step

In the update step the predicted cardinality distribution and PHD are updated using the measurements. In the prediction step the estimation of the PHD was done with the same equations as the PHD filter. This is no longer the case in the correction step, since the cardinality and PHD function are coupled. When taking the dependency of the PHD on the cardinality into account, the update PHD equation becomes [78]:

$$\underbrace{v_k(x)}_{\text{updated PHD}} = \underbrace{(1-p_{D,k}) \frac{\langle \Upsilon_k^1[w_{k|k-1}, Z_k], p_{k|k-1}\rangle}{\langle \Upsilon_k^0[w_{k|k-1}, Z_k], p_{k|k-1}\rangle} v_{k|k-1}(x)}_{\text{PHD missed detections}} + \underbrace{\sum_{z\in Z_k} \sum_{j=1}^{J_{k|k-1}} w_k^{(j)}(z)\mathcal{N}(x; m_k^{(j)}(z), P_k^{(j)})}_{\text{PHD update prediction}} \tag{3.3.26}$$

So the only difference compared to the update equation of the PHD filter in equation 3.3.3 is the term $\frac{\langle \Upsilon_k^1[w_{k|k-1}, Z_k], p_{k|k-1}\rangle}{\langle \Upsilon_k^0[w_{k|k-1}, Z_k], p_{k|k-1}\rangle}$, which is also present in the calculation of the weights of the GM components:

$$w_k^{(j)}(z) = p_{D,k} w_{k|k-1}^{(j)} q_k^{(j)}(z) \underbrace{\frac{\langle \Upsilon_k^1[w_{k|k-1}, Z_k\{z\}], p_{k|k-1}\rangle}{\langle \Upsilon_k^0[w_{k|k-1}, Z_k], p_{k|k-1}\rangle}}_{\substack{\text{Likelihood of measurements}\\ \text{given the cardinality distribution}}} \tag{3.3.27}$$

The part $\Upsilon_k^0[w_{k|k-1}, Z_k](n)$ is the likelihood of the measurements Z_k, given that there are n targets. The inner product of all these likelihoods of n with the cardinality distribution results thus in a single value, representing the likelihood of the measurements Z_k, given the (predicted) cardinality distribution. The denominator of the term is normalization term. So the entire term is a factor between 0 and 1 representing likelihood of the measurements given the cardinality distribution.

This term is calculated with the following equations:

- $\langle \cdot, \cdot \rangle$ is the inner product. The inner product of two real-valued functions α and β is defined as

$$\langle \alpha, \beta \rangle = \int \alpha(x)\beta(x)dx$$

- $\Upsilon_k^u[w, Z](n) = \sum_{j=0}^{min(|Z|,n)} (|Z| - j)! p_{K,k}(|Z| - j) \frac{n!}{(n-j+u)!} \times \frac{\langle 1 - p_{D,k}, v \rangle^{n-(j+u)}}{\langle 1, w \rangle^{j+u}} e_j(\Xi_k(w, Z))$ the likelihood of measurements Z_k, given that there are n targets
- $e_j(Z)$ elementary symmetric function of order j
- $\Xi_{(w,Z)}(x) = \left\{ \frac{\langle 1, \kappa_k \rangle}{\kappa_k(z)} P_{D,k} w^T q_k(z) : z \in Z \right\}$
- $w_{k|k-1} = [w_{k|k-1}^{(1)} \dots w_{k|k-1}^{(J_{k|k-1})}]^T$
- $p_{K,k}$ Cardinality distribution of clutter.
- $p_D(x)$ the probability of the object x being detected by the sensor
- $\kappa_k(z)$ the intensity of the clutter in m^{-2}

The other terms in the equation are unchanged from the PHD filter and are respectively:

- $q_k^{(j)}(z) = \mathcal{N}(z; H_k m_{k|k-1}^{(i)}, R_k + H_k P_{k|k-1}^{(i)} H_k^T)$
- $m_{k|k}^{(j)}(z) = m_{k|k-1}^{(j)} + K_k^{(j)}(z - H_k m_{k|k-1}^{(j)})$
- $P_{k|k}^{(j)} = [I - K_k^{(j)} H_k] P_{k|k-1}^{(j)}$
- $K_k^{(j)} = P_{k|k-1}^{(j)} H_k^T (H_k P_{k|k-1}^{(j)} H_k^T + R_k)^{-1}$

The likelihood of the measurements with respect to the cardinality distribution is also included in the update equation of the cardinality distribution. The update of the predicted cardinality distribution $p_{k|k-1}(n)$ with this likelihood result in the update equation of the cardinality distribution [78]:

$$p_k(n) = \frac{\Upsilon_k^0[w_{k|k-1}, Z_k](n)}{\langle \Upsilon_k^0[w_{k|k-1}, Z_k], p_{k|k-1} \rangle} p_{k|k-1}(n) \tag{3.3.28}$$

Implementations of the CPHD

As already explained in section 3.3.2 is the Matlab code of the CPHD implemented by Vo et al. [73]. In this thesis the GM-CPHD implementation of this code is adapted to the radar and camera setup and the stereo camera setup. Besides a GUI is developed to visualize the PHD in time and to browse through the tracking results of the filter in time. Besides this code is used as a basis for the implementations done in this thesis for MM-CPHD and CMM-PHD explained in more depth in respectively section 3.4.1 and 3.4.2

The CPHD filter is also applied in automotive application by Lamard et al. for tracking vehicles and pedestrians using a camera and radar attached to a vehicle for both synchronous sensors [32] and asynchronous sensor [33].

3.4 Random Finite Set approaches and Classification

The PHD and CPHD of section 3.3.2 and 3.3.5 only allow the use of a single motion model. A single motion model is sufficient for nonmanoeuvring objects, but the motion of manoeuvring objects may have to be described by a combination of motion models covering the motion characteristics of the different manoeuvrers. Also in multi-object tracking, the different types of objects may not be described by a single, universal motion model. For example in automotive applications, each type of road users has its own motion characteristics. A pedestrian is, due to its agility, able to change direction instantly, while the motion direction of vehicles and cyclist are dependent on their orientation [49].

To address this issue, Mahler proposed a Jump Markov models based variant of the PHD filter called the Multiple Model PHD filter (MM-PHD) [45]. Vasha al. implemented a closed form Gaussian Mixture solution for the MM-PHD [60] and Georgescu et al. implemented a MMCPHD variant based on the bin-model approach [24]. The bin-model approach is closely related to the more familiar bin-occupancy CPHD filter [15]. A Multiple Model implementation of the GM-CPHD of Vo et al. [77] has, to our knowledge, not been devised so far. Therefore this MM-CPHD approach is implemented in this thesis and discussed in more details in section 4.1.

3.4.1 Multiple Model Probability Hypothesis Density Filter

The correct model associated with the object is selected based on its class or in case of manoeuvring objects on its mode. So, for example, if the object is classified as a pedestrian, the corresponding motion of a pedestrian is used to predict the motion. The information about the current class or mode of an object is in the MM-PHD stored in an augmented discrete state variable o called the $mode$ or $jump$ variable. This results in the augmented state for an object.

$$\mathring{x} = (x, o) \tag{3.4.1}$$

In the GM implementation, each candidate object is described by a GM component. The $mode$ of an object is therefore in the GM implementation added to the state of a GM component. For the GM implementation, where each candidate object is described by a GM component, this results in the augmented state of GM component i:

$$\mathring{x}^{(j)} = (o^{(j)}, w^{(j)}, \mathcal{N}(x; m^{(j)}, P^{(j)}) \tag{3.4.2}$$

with j the considered GM component.

Prediction step

In the prediction state, a prediction is made for the motion of each GM component. As a consequence of using multiple motion models, the transition model $f_{K|k-1}(x)$ has become dependent on the mode $f_{K|k-1}(x, o)$. In addition, the MM-PHD takes also into account the possibility that an object switches in mode between the previous time step and the upcoming time step. When this option was not taken into account and an object would switch in mode, the incorrect motion model would be selected, resulting in an incorrect state prediction compared to the measurement. Therefore per GM component a prediction is made for each motion model $F(o)$ and noise $Q(o)$ and its weight is factorized by the likelihood $t_{o_{k|k-1}}$ of jumping from one mode o_{k-1} to another o_k.

$$f(\mathring{x}_k|\mathring{x}_{k-1}) = t_{o_{k|k-1}} \cdot \mathcal{N}(x; F(o_{k-1})x^{(j)}_{k-1}, Q^{(j)}(o_{k-1})) \tag{3.4.3}$$

This model transition probability $t_{o_{k|k-1}}$ is predefined in the Markov transition matrix and assumed to be constant and equal for all GM components [60]. For example, for a scenario with the pedestrian, cyclist and car as classes the Markov transition matrix T is as follows:

$$T = \begin{bmatrix} t_{ped|ped} & t_{cyc|ped} & t_{car|ped} \\ t_{ped|cyc} & t_{cyc|cyc} & t_{car|cyc} \\ t_{ped|car} & t_{cyc|car} & t_{car|car} \end{bmatrix} \tag{3.4.4}$$

In this scenario it is very likely that an object classified as pedestrian in the current time step will be classified as a pedestrian in the upcoming time step. So the factor $t_{ped|ped}$, as well as the factors $t_{cyc|cyc}$ and $t_{car|car}$ will be high. But the likelihood that a pedestrian will become a car is not very likely, so the factor $t_{ped|car}$ will be low. However sometimes cyclists and pedestrians are mixed up by a classifier, so this accuracy in the classifier could be expressed in the $t_{cyc|ped}$ transition probability. The sum of each row in the Markov transition matrix is always equal to one. When including the transition probability and duplicates into the PHD prediction equation 3.3.3, the MM-PHD prediction equation becomes.

$$\underbrace{v_{k|k-1}(\mathring{x})}_{\text{predicted PHD}} = \underbrace{\sum_{o=1}^{O}}_{\substack{\text{class}\\\text{model}\\\text{duplicates}}} \left(\underbrace{\sum_{j=1}^{J_{\gamma,k}(o)} t^{(j)}_{o_{k|k-1}} w^{(j)}_{\gamma,k} \mathcal{N}(x; m^{(j)}_{\gamma,k}, P^{(j)}_{\gamma,k}}_{\text{PHD of birth objects}} + \underbrace{p_{S,k} \sum_{j=1}^{J_{k-1}(o)} t^{(j)}_{o_{k|k-1}} w^{(j)}_{k-1} \mathcal{N}(x; m^{(j)}_{S,k|k-1}, P^{(j)}_{S,k|k-1})}_{\text{PHD of survived objects}} \right) \tag{3.4.5}$$

where

- $m^{(j)}_{S,k|k-1} = F(o_{k|k-1}) m^{(j)}_{k-1}$
- $P^{(j)}_{S,k|k-1} = Q(o_{k|k-1}) + F(o_{k|k-1}) P^{(j)}_{k-1} F^T(o_{k|k-1})$

The duplication of each GM for every mode does have a consequence for the total number of Gaussians. Where in the PHD prediction the number of GM components is equal to the sum of the birth GM components and survival GM components $J_\gamma + J$ is in the MM-PHD this number multiplied by the amount of different modes $N_o(J_\gamma + J)$.

Update step

The update step is unchanged compared to the single motion model PHD, except the fact that the weight update is a summation over the model modes and the total PHD is the sum of the individual mode PHDs.

$$v_k(x) = \underbrace{\sum_{o=1}^{O}}_{\substack{\text{class}\\\text{duplicates}}} \left(\underbrace{(1 - p_{d,k}) v_{k|k-1}(\mathring{x})}_{\text{PHD of missed detection}} + \underbrace{\sum_{j=1}^{J_{k,k-1}(o)} w^{(j)}_k(z) \mathcal{N}(x; m^{(j)}_{k|k}(z), P^{(j)}_{k|k})}_{\text{update of predicted PHD}} \right) \tag{3.4.6}$$

with

$$w^{(j)}_k(z) = \frac{p_{d,k} w^{(j)}_{k|k-1} q^{(j)}_k(z)}{\kappa_k(z) + p_{d,k} \sum_{o=1}^{O} \sum_{j=1}^{J_{k|k-1}(o)} w^{(j)}_{k|k-1} q^{(j)}_k(z)} \tag{3.4.7}$$

After the update step a merging and pruning step can be applied according to the same principle as the phd filter in section 3.3.2. The duplicates of the unlikely modes are then pruned by the pruning threshold and similar GM components are merged.

MM-PHD implementations

The available Matlab code for the CPHD developed by Vo et al. [73] is extended to the MM-CPHD and used in the evaluation to compare the proposed approach with the MM-CPHD. Besides for automotive application Meissner et al. implemented the MM-PHD for tracking road users at intersections [50][49][52]. Their classifying MMPHD is discussed in more detail in section 3.4.2. Granström et al. implemented the MM-PHD for extended objects [28]. Extended objects are objects that might be described by more than one measurement. By combining the multiple model approach with the extended target tracking filter, Granström et al. has created the possibility to track bicycles whose appearances could abruptly change in time. Georgescu et al. implemented a MMCPHD variant based on the bin-model approach [24] and applied it to simulations on multi-static sonar datasets.

3.4.2 The Classifying Multiple Model Probability Hypothesis Density Filter

Meissner et al. [50] proposed the Classifying Multiple Model Probability Hypothesis Density filter (CMM-PHD), which uses classification probabilities to improve the performance of the MM-PHD. The CMM-PHD is part of a project with the aim of increasing traffic safety at urban intersections by improving the perception of these intersections. In this project a network of lasers and cameras installed on an urban intersection are used in the CMM-PHD in order to estimate the states, classifications and dimensions of pedestrians, bikes, cars and trucks. This estimated dynamical model of the intersection's scene is then send to all participating vehicles approaching the intersection, resulting in an improved perception of the intersection [50].

Structure of the CMM-PHD

Where an object in the original Multiple Model PHD filter is only represented by a Gaussian Mixture component, is the Gaussian Mixture component in the CMM-PHD extended with an additional classification probability m_c and object's dimension d.

$$GM^{(i)} = (w^{(i)}, \mathcal{N}(x, x^{(i)}, P^{(i)}), m_c^{(i)}, \mathcal{N}(d, d^{(i)}, P_{dim}^{(i)})) \tag{3.4.8}$$

with

- $w^{(i)}$ is the weight of the Gaussian Mixture Component
- $\mathcal{N}(x, x^{(i)}, P^{(i)})$ is the state of the object, i.e. its location, velocity
- $m_c^{(i)}$ is a vector containing the probabilities for each class
- $\mathcal{N}(d, d^{(i)}, P_{dim}^{(i)})$ are the dimensions of the object, i.e. the length, width, height.

However, the classification probabilities and dimensions of the Gaussian Mixture are independently estimated with respect to the state of the object. An illustration of a full cycle of the CMM-PHD is shown in figure 3.17.

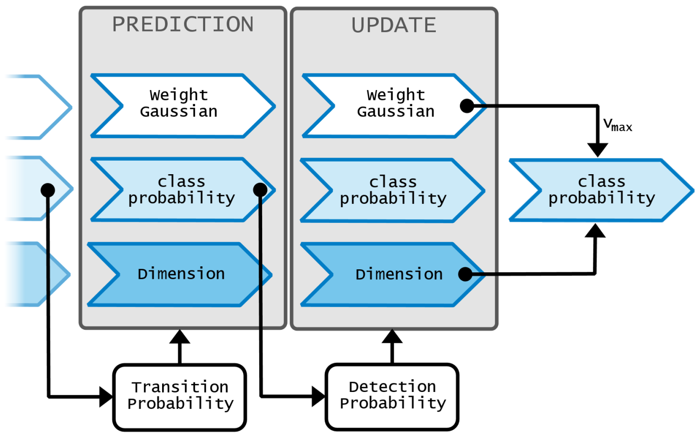


Figure 3.17: Illustration of a full cycle of the CMM-PHD

State The top layer represents the prediction and update of the state and weight of the Gaussian mixture. This is done with the original MM-PHD described in section 2.2.2. The original MM-PHD filter assumes that the transition probability and detection probability are constant in time and for every Gaussian mixture. This is no longer the case when using class probabilities, since these parameters are class dependent [50]. An implementation of these class dependent transition probability and detection probability is explained in more depth in the next paragraph.

Class probabilities The middle layer depicts the estimation process of the classification probabilities. These probabilities are estimated using measurement features from different type of sensor, together with features of the track. The different type of sensors have different detection abilities, for example the low resolution camera is only able to detect vehicles, while the lasers is able to distinguish all the four classes. The construction of the probabilities from the different and sometimes contradiction measurements is explained in section 3.4.2.

Dimensions In the bottom layer the estimation process of the length, width and height of the object is visualized. The dimensions of an object may differ per sensor and are highly dependent on the viewpoint of the sensor. For example when an object is occluded by one sensor, it is still visible on the sensor mounted on the opposite side. These variations are addressed by estimating the dimensions with Kalman filter independent of the state estimation.

State estimation using the Multiple Model PHD

The state and weight estimation of the Gaussian Mixture components are done with the original Multiple Model PHD. The prediction and update equations of the original filter themselves are untouched and the same as in section 2.2.2. However in the original filter is assumed that the class of the Gaussian Mixture component is known and not a probability. This class is then used to determine the probability of that class being the same or another class in the upcoming time step. This is based on predefined constant values in the transition matrix. However, in the CMM-PHD filter, the class uncertain and is defined as a probability. Besides it is different in time and different for each Gaussian Mixture component. Therefore in the CMM-PHD a class dependent Markov transition matrix is introduced.

Markov Transition matrix The Markov transition matrix T makes it possible for an object to jump from one class to another in the next time step. The chance of jumping from one class o to another $o+$ is predefined in the transition matrix by a probability $t_{o+,o}$ for each jump, for example $t_{cyc,car}$ represents the chance of a car to become a cyclist in the next step.

$$T = \begin{bmatrix} t_{ped,ped} & t_{ped,cyc} & t_{ped,car} & t_{ped,tru} \\ t_{cyc,ped} & t_{cyc,cyc} & t_{cyc,car} & t_{cyc,tru} \\ t_{car,ped} & t_{car,cyc} & t_{car,car} & t_{car,tru} \\ t_{tru,ped} & t_{tru,cyc} & t_{tru,car} & t_{tru,tru} \end{bmatrix} \tag{3.4.9}$$

In the prediction step a copy of each Gaussian Mixture component per class is created. Then a prediction is made for each copy, based on the motion model of the corresponding class, and subsequently factorized by its jump probability $t_{o+,o}$. This jump probability is larger for realistic scenarios with respect to unrealistic scenarios. So for the scenario that a pedestrian will be again a pedestrian $t_{ped,ped}$, the peak of the Gaussian Mixture component will be larger than the one with a more unrealistic scenario.

$$f(x_{+,o+}|x, o) = t_{o+,o} \cdot f(x_+|x, o) \tag{3.4.10}$$

In the CMM-PHD the class of an object is not known, but is expressed as a probability for each class. So instead of an object being a car, it is for example 80% pedestrian, 10% bike, 6% car and 4% truck. One way would be to take the highest probability as the class and then use the predefined transition matrix. However, in that case an object with a probability of 51% pedestrian would have the same possibility to jump to another class than an object with a probability of 99% pedestrian. While the one with 51% pedestrian could have a probability of 49% for another class and therefore a way larger chance of jumping to that other class than the 99% pedestrian scenario. So the class probabilities already contain information about the chance of jumping to another class and are therefore directly used in the transformation matrix. Since each Gaussian Mixture component i has a corresponding class-vector $m_c^{(i)}$, the Markov transformation matrix is unique for each GM component and is constructed from the class probability vector.

$$T^{(i)} = \begin{bmatrix} p_{ped} & p_{cyc} & p_{car} & p_{tru} \\ p_{ped} & p_{cyc} & p_{car} & p_{tru} \\ p_{ped} & p_{cyc} & p_{car} & p_{tru} \\ p_{ped} & p_{cyc} & p_{car} & p_{tru} \end{bmatrix} \tag{3.4.11}$$

So the higher the probability of being a certain class, the larger the peak of the corresponding predicted Gaussian mixture component. If the location of a measurement is then close to this prediction, the likelihood of this prediction is high, resulting in an even larger peak and vice versa.

Detection probability In the original MM-PHD, the probability of an object being detected by a sensor is assumed to be constant. The static environment of the intersection gives however a possibility to define a position and sensor dependent detection probability $P_D^{(i)}(x)$. Based on the FOV and range of the sensor, a detection map has been created which indicates at which location it is likely to detect an object and in which location it is not. Besides not all sensors are able to detect all classes, for example the low resolution camera only detects vehicles. So the detection probability of a low resolution camera to detect a pedestrian is well-nigh zero. This results in a detection probability $P_D^{(i)}(RA^S)$ based on the sensors recognition ability RA^S and based on the detection probability of the location of the object $P_D^{(i)}(x, s)$.

$$P_D^{(i)} = P_D^{(i)}(RA^s)P_D^{(i)}(x, s) \tag{3.4.12}$$

Class probabilities

The class probabilities are constructed from measurement features obtained by the sensors and track features from the filter estimates. The measurement features are gathered by three types of sensors, namely lasers, low resolution cameras and high resolution cameras.

Measurement features Each type of sensor has different recognition abilities. All four classes are recognizable by the lasers. The low resolution camera is only able to detect vehicles, so the detection could be a car or a truck, but it is unknown which one. The high resolution camera is only able to detect pedestrians.

- **Laser** The 14 lasers have fixed positions at the intersection, making it possible to distinguish background points from points originated from objects. The point clouds obtained by clustering are subsequently examined for features of each class. The height, location and standard deviation of the cluster are used as features. For example a large height distinguishes a truck from a pedestrian and it is more likely for a pedestrian to occur at a side walk than a car. The summation of the four likelihoods of the features results then in probabilities of each class.

- **Low resolution camera** Three low resolution cameras are used to detect approaching vehicles. The high mounted position of these cameras reduces the risk of occlusion and increases the field of view. The detection of the vehicles are based on an approach of Viola et al. which makes use of Haar-like features together with a cascaded processing scheme [72]. Verification and the assignment of the corresponding probability is done with a feed forward Neural Network [20].

- **High resolution camera** Two high resolution cameras are used to realize a wide angle stereo vision. The detection of pedestrians is based on classification using HOG descriptors together with linear and Gaussian kernel support vector machines (SVM) [80].

Track features The track velocity and dimensions are used to improve the class estimates from the sensors. For example, it is hard to distinguish pedestrians and bikes only based on measurement features, but they differ significantly in their possible maximum velocity [50]. At low maximum velocity, the object could be all four classes, so the probability of being a pedestrian or bike or car or truck is high. If the velocity becomes higher than 2.5 m/s it is probably no longer a pedestrian and the probability of being a bike or car or truck $P(BCT)$ starts increasing while $P(PBCT)$ decreases. When a maximum velocity of 5 m/s is reached, $P(BCT)$ starts decreasing and $P(CT)$ increases and so on. The same principle is applied to the filtered length of an object and independently to the filtered width of an object.

Fusion of classification data The final probabilities are constructed by the fusion of the probabilities obtained by the different sensors, together with the probabilities obtained by the track features. The probability information of different sources is however not of the same form. Besides the different information could also be contradictory. Therefore the Dempster-Shafer theory of evidence is used, which naturally handles the contradictory measurements of multiple sensor types [50]. However the Dempster-Shafer theory of evidence suffers from Zadeh's paradox. This paradox happens when two contradicting sources have a high probability and zero probability for the same classes but adversely. If both sources have a very low probability for the third class, this class becomes 1 due to the only overlap on this unrealistic class. This paradox is solved by preventing a zero assignment to a class by discounting [52]. Dempster-Shafer theory of evidence and Zadeh's paradox is explained in more depth in section 4.1.4.

3.4.3 The Classification Aided Cardinalized Probability Hypothesis Density Filter

When the output of a classifier consist of only one class, the original MM-PHD filter of section 2.2.2 can be used to improve the tracking accuracy. The class is then used to make an improved prediction by selecting the class-specific motion model. Ramona Georgescu et al. [23] proposed the Classifiication Aided CPHD filter where the class information, in addition to the prediction, is also used in the update step.

In the original CPHD the likelihood of a measurement being generated by the object is determined by kinematic information only. I.e. if a measurement is located close to an object, the likelihood will be larger than for measurements further away. Georgescu et al. also defined a likelihood based on the class information. So if the object's class is the same as the class corresponding to a measurement, the likelihood is relatively large compared to a situation where both classes are different.

Bin model implementation

So far only the Multiple Model PHD is considered in this thesis. However the PHD filter suffers from a "target death" problem, caused by the Poisson assumption for the cardinality distribution [78][24][14]. The CPHD solves this issue by propagating the distribution of the cardinality as well.

Georgescu et al. implemented a MMCPHD variant based on the bin-model approach [24], which is closely related to the more familiar bin-occupancy CPHD filter [15]. In the bin-model approach, the PHD surface is divided into infinitely small bins. Instead of representing candidate objects by Gaussian Mixture components with a mean, variance and weight, the bin model approach estimates for each bin the probability of containing an object. The bins are sufficiently small that each bin is potentially occupied by at most one target [15] and variance is expressed by decreasing probabilities in surrounding bins. The principles for the prediction and update calculations are the same for both the Gaussian mixture as the bin-model implementations, but they differ in particular in the fact that the estimations are calculated for each bin in the bin-model approach compared to each GM component in the GM implementation.

The CACPHD is based on this bin-model approach of the MMCPHD. In the CACPHD the update step is extended with a likelihood for a class as well. Where in the MMCPHD the likelihood of a bin containing an object given a certain measurement is determined by the location of the measurement only, is in the CACPHD the class of the measurement also taken into account.

Construction of Likelihood

In the original MM-CPHD, the likelihood of a measurement belonging to an objects is determined by its kinematic state only. So if a measurement is closely located to an object, it is more likely that the measurement is generated by that object compared to a measurement further away. However if only the kinematic state is taken into consideration, the likelihood of the measurement generated by a car would be the same as that of a pedestrian. This could result in the fact that a pedestrian could be updated with a measurement of a car, i.e. when a pedestrian and a car approach each other closely. This is prevented in the CACPHD by constructing the likelihood on both the kinematic state and the classification.

Likelihood in the original MMCPHD The likelihood of a measurement z being generated by bin j is in the original MMCPHD defined by

$$L_z = \sum_n q_x(z|x_i) \cdot \frac{p(U_i|c)}{\sum_j p(U_j|c)} \tag{3.4.13}$$

where

- $q_x(z|x_j)$ is the likelihood of measurement z given the object's location x_j

- $\frac{p(U_i|c)}{\sum_j p(U_j|c)}$ is the probability of bin i containing an object with respect to all bin's j
- n is the number of targets

So the likelihood is only defined by the kinematic state of the measurement compared to the object represented by the factor $f(z|x_j)$.

Likelihood in the CACPHD In the CACPHD the likelihood is not only defined by the likelihood of the kinematic state of the measurement with respect to the bin, but also by the likelihood of both classes. This likelihood is added by a the $c_{q\bar{q}}$, resulting in the new likelihood definition

$$L_z = \sum_j q_x(z|x_j) \cdot q_{c\bar{c}} \cdot \frac{p(U_j|c)}{\sum_l p(U_l|c)} \tag{3.4.14}$$

The factor $q_{c\bar{c}}$ models the accuracy of the classifier and represents the likelihood of the true class being c, when the classifier outputs $\bar{c}$. This factor is predefined in a confusion matrix, which for example for a situation with 3 classes of pedestrian, cyclist and car defined is as follows:

$$q_{c\bar{c}} = \begin{bmatrix} c_{ped,p\bar{e}d} & c_{ped,c\bar{y}c} & c_{ped,c\bar{a}r} \\ c_{cyc,p\bar{e}d} & c_{cyc,c\bar{y}c} & c_{cyc,c\bar{a}r} \\ c_{car,p\bar{e}d} & c_{car,c\bar{y}c} & c_{car,c\bar{a}r} \\ c_{tru,p\bar{e}d} & c_{tru,c\bar{y}c} & c_{tru,c\bar{a}r} \end{bmatrix} \tag{3.4.15}$$

If the class of the bin is the same as the classification of the measurement, this factor will be high with respect to a situation where both classes are different. However if it is known that the classifier regularly classifies a cyclist incorrectly as a pedestrian, the factor $q_{cyc,p\bar{e}d}$ will be defined high as well.

3.4.4 Multi-Target PHD Tracking and Classification Using Imprecise Likelihoods

Very recently Fortin et al. proposed a method for multi-object tracking and classification based on kinematic data only [19]. The used tracking framework is the MM-PHD introduced by Mahler [45] and the class-probabilities are included in the same way as Meissner et al. did in the CMM-PHD filter. So instead of using a fixed transformation matrix in the prediction step, the transformation matrix is different for each Gaussian and fed with the class-probabilities.

The difference between the implementation of Meissner et al. and Fortin et al. is in the way of determining the classification probabilities. Meissner et al. determines the classification probabilities at the end of the filtering step by the fusion of measurement features from multiple sensors, together with kinematic features such as the dimension and velocity of the object. However when using only one sensor, the classification data could be ambiguous or incomplete. Therefore Fortin et al. proposed a method for determining the classification probabilities by kinematic data only, resulting from the filtering step. This new approach, the Credal Classification Multiple Model PHD filter (CC-MMGMPHD), is evaluated in an application for tracking and classification of aircrafts using only one sensor.

Classification using kinematic features

Instead of using measurement features to classify an object, as is done with the HOG or CNN classifiers from section 2.4, the CC-MMGMPHD looks at the kinematic behaviour of the object. A correspondence between each class and specific behaviour is established, for example velocity classes or acceleration limits and used to identify the class of a Gaussian by its behaviour.

The construction of the classification probabilities starts by identifying the behaviour of the object. All the possible behaviours are defined in a set $\Omega^B = \{b_1, ..., b_{n_b}\}$ and during the classification process per GM component is determined how much it exhibits each of the behaviours in the set. This results in a mass function $m^{j,\beta}$ describing the likelihood of each behaviour for GM component j. Subsequently this mass function is transformed to a mass function $m^{j,c}$ describing the likelihood of each class. This is done with a matrix $\bar{M}$ defining the links between behaviours and the classes.

$$m^{j,c} = \bar{M} \cdot m^{j,\beta} \tag{3.4.16}$$

However, the class mass function does not have to consist of class-probabilities. It could also consist the mass of being two classes, but unknown which of the two. For example, it could contain the mass of being a vehicle, but unknown how much a car and how much a truck. Therefore a Pignistic transformation is used to convert the mass function into probabilities

$$P^j(c) \approx BetP^j(C) = \sum_{c \in C} \frac{m^{j,C}(C)}{|C|(1 - m^{j,C}(\emptyset))} \tag{3.4.17}$$

Evaluation

Evaluation is done for the application of tracking and classifying aircrafts using only one sensor. For the three classes liner c_1, bomber c_2 and fighter aircraft c_3 the distinctive manoeuvrability is used as a typical behavioural parameter; if the object has a constant velocity, the object could be all three classes $b_1 = \{c_1, c_2, c_3\}$. When it is moving with an acceleration of maximum $a \leq 2g$, the class could be a bomber or a fighter $b_2 = \{c_2, c_3\}$ and if the acceleration is higher than $a > 2g$ the object must be a fighter $b_3 = \{c_3\}$. This results in the matrix $\bar{M}$ describing the links between behaviours and classes:

$$m^{j,c} = \bar{M} \cdot m^{j,\beta}$$

$$\begin{bmatrix} \emptyset \\ c_1 \\ c_2 \\ c_1, c_2 \\ c_3 \\ c_1, c_3 \\ c_2, c_3 \\ c_1, c_2, c_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} \emptyset \\ b_1 \\ b_2 \\ b_1, b_2 \\ b_3 \\ b_1, b_3 \\ b_2, b_3 \\ b_1, b_2, b_3 \end{bmatrix} \tag{3.4.18}$$

This class mass function is subsequently converted into probabilities by the Pignistic transformation of equation 3.4.4.

The CC-MMGMPHD is tested on an aircraft tracking example and compared with two different Bayesian classifiers. The evaluation shows that tracking accuracy and classification has improved compared to the two other approaches with Bayesian classifiers [19].

3.5 Discussion

The CMM-PHD filter extends the original MM-CPHD filter with class probabilities and dimensions, in order to improve the prediction with class specific motion models. Multiple sensors with different characteristics are used and the obtained data is fused by accurate modelling of the sensors' recognition probabilities. The CMM-PHD is evaluated for track and classification accuracy for road users on a public traffic intersection, which results in an excellent performance of the filter [50]. Interesting would be to see if this also counts for the application of an autonomous vehicle with radar and camera. The static environment of the intersection together with the fusion of multiple of sensors are giving high reliability in classification. The FOV of the sensors on an autonomous driving vehicle is always relative to the vehicle itself and is therefore vulnerable to occlusion. The occlusion together with using only the camera data for classification results in a lower reliability in classification. The question is therefore whether the filter will also result in an improvement in situations where the classification is uncertain or even incorrect.

Besides the CMM-PHD is based on Mahler's PHD filter. The PHD filter however suffers from a large variation in the number of objects, caused by the Poisson assumption for the cardinality distribution [78][24][14]. The CPHD is an extension of the PHD and solves the "target death" problem by propagating the distribution of the cardinality as well. This CPHD is used in the CACPHD filter, which integrates the class information in the update step of the filter as well. The inaccuracy of the classifier is included in a confusion matrix, which is used in the update step to include the class-likelihood as well. By using this class-likelihood together with the kinematic state likelihood in the measurement to track association, a more accurate tracking result in multistatic sonar scenarios is achieved [24]. Georgescu et al. disabled switching from one class to another in the prediction step by defining the Markov transition matrix as the identity matrix and uses discrete classes instead of class-probabilities. The use of discrete classes instead of probabilities may lead to selecting the incorrect class due to rounding, resulting in a prediction with an incorrect motion model and decrease of the GM component weight due to the low class-likelihood. Therefore it would be interesting to see the results of the approach in situations with incorrect classification or uncertain classification.

In table 3.1 an overview is given of the two approaches. As shown in the overview is the uncertainty in classification in the approach of Meissner et al. expressed by classification probabilities and are the probabilities applied in the prediction step of the MM-PHD filter. The original MM-PHD state estimation equations are untouched and the classification probabilities are not used in the update step. Georgescu et al. does use the classification in the update step, but does that discrete classes in the bin model approach of the CPHD. Besides switching from class is disabled, so incorrect classified tracks are not able to change in classification and correct themselves. A combination of both approaches, where the uncertainty in classification is expressed as classification probabilities and used in both the prediction step and update step might be a robust way of dealing with uncertainties in classification. Thus, the .. filter is proposed where the classification uncertainties are propagated through all the steps of the filter. A Gaussian Mixture implementation of the MMCPHD of Vo et al. [77] has been derived as a cure for the "target death" problem from the PHD filter.

	CMM-PHD Meissner et al.	CACPHD Georgescu et al.
prediction	• prediction based on class-specific motion models using the MM-PHD framework • class-probabilities included in a GM component unique Markov transition matrix, with as a consequence that the weights of the class-duplicates are dependent on the class probability	• prediction based on class-specific motion models using the MM-PHD framework with a fixed Markov transition matrix • Switching from class disabled by defining the Markov transition matrix as the identity matrix
update	• original MM-PHD update equations used • association likelihood based on location only	• association likelihood based on location and classification similarity
class estimation	• Class probabilities used • Class probabilities estimated by the fusion of class-information from multiple sensors, together with class information obtained from track features. • The fusion of the different and sometimes contradictory class information using Dempster-Shafer theory.	• Discrete class received from classifier.

Table 3.1: Overview of the recursion steps in the CMM-PHD and CACPHD filter

4 Proposed Approach

In chapter 3 the Multiple Model PHD is discussed, which is able to use class-specific motion models in the prediction step of the PHD filter. The correct motion model per object is selected based on the rounded class given by the classifier. At the same time, the filter allows that objects switch in class between time steps, by making use of predefined switching probabilities in the Markov transition matrix.

The classification given by the classifier, on the other hand, often contains uncertainties. For example a bicycle is regularly mistaken by a pedestrian and vice versa as shown in section 2.4.4. In that case the incorrect motion model is selected, which results in an inaccurate prediction. Besides the incorrect classification is untouched in the filter and passed on to the next step of the system architecture of the autonomous driving vehicle. To prevent the selection of an incorrect classification model in classification uncertain conditions, the prediction could also be based on the class-probabilities instead of a single class. The original MM-PHD filter has no possibility to use class-probabilities and therefore Meissner et al. [50] has included the class-probabilities in the Markov Transition matrix in the CMM-PHD filter, discussed in section 3.4.2. As a result, the prediction is based on the classification uncertainty.

In this chapter, a more robust multi-sensor multi-object tracking approach for dealing with classification uncertainty is proposed. In this approach, called the Robust Classification Aided CPHD (RCA-CPHD), the classification uncertainties are not only used in the prediction, but in all steps of the filter. By propagating and using the classification uncertainty in all the steps, both the tracking accuracy and the classification accuracy are improved during classification uncertain conditions. Section 4.1 starts with the recursion of the RCA-CPHD. The way in which the class-probabilities in the filter are implemented in the prediction, update, gating, merging/pruning/capping step and how it contributes to the improvement is explained. Besides a Gaussian Mixture implementation of the MM-CPHD is devised and implemented in the steps to prevent the "target death" problem of the (MM-)PHD filter. Subsequently in section 4.2 the used motion models for considered classes are discussed and finally in section 4.3 the used measurement models for the applied sensors are explained.

4.1 Filter Recursion

The RCA-CPHD is based on the Multiple Model implementation of the GM-CPHD. A Multiple Model implementation of the closed form GM-CPHD, proposed by Vo et al. [77], has to our knowledge not been devised so far. Georgescu et al. proposed a GM-MMCPHD based on the bin-model approach discussed in section 3.4.3, which is closely related to the more familiar bin-occupancy CPHD filter [15]. However for the original Gaussian Mixture implementation only a Multiple Model variant for the PHD has been devised by Mahler [45].

4.1.1 Gaussian Mixture representation

Candidate objects in the GM implementation of the PHD and CPHD filter, discussed in respectively 3.3.2 and 3.3.5, are represented by GM components. The PHD $v_(x)$ in these filters is therefore a set of GM components:

$$v(x) = \sum_{j=1}^{J} w^{(j)} \mathcal{N}(x; \bar{m}^{(j)}, P^{(j)}) \tag{4.1.1}$$

With GM component j heaving a weight $w^{(j)}$, mean $m^{(j)}$ and variance $P^{(j)}$. In the MM-PHD, discussed in 3.4.1, the state of a GM component was extended by a mode variable $o^{(j)}$. This mode variable was substituted in the CMM-PHD of section 3.4.2 by class probabilities $m_c^{(j)}$. In the RCA-CPHD the probabilities are propagated in a similar way by augmenting the state of a GM component with a class probability vector $c^{(j)}$. Besides a label $l^{(j)}$ is added to each GM component containing its origin. This label enables the possibility to extract tracks from the state estimates and is used in the steps of the filter to improve the state estimation. The augmented state of a GM component is therefore described by:

$$MC^{(j)} = (l^{(j)}, c^{(j)}, w^{(j)}, \mathcal{N}(x; m^{(j)}, P^{(i)}) \tag{4.1.2}$$

with

- $w^{(j)}$ is the weight of the GM component
- $\mathcal{N}(x, x^{(j)}, P^{(j)})$ is the state and covariance of the GM component
- $c^{(j)}$ is a vector containing the probabilities for each class
- $l^{(j)})$) is a label containing the origin of the GM component

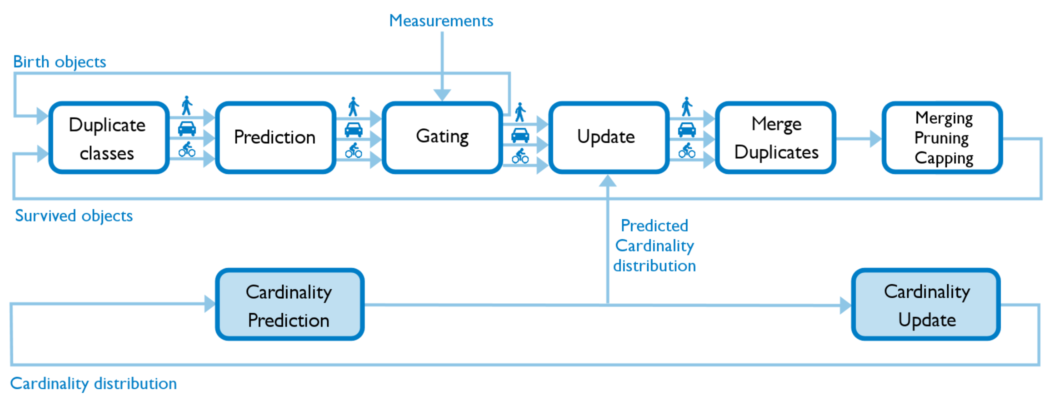


Figure 4.1: Overview of the recursion of the RCA-CPHD filter

4.1.2 Prediction

In the prediction step of the filter, a prediction is made what the state of each GM component will be at the time of arrival of the next measurements. The CPHD also propagates the distribution of the cardinality. So in addition to the prediction of the PHD, the RCA-CPHD also makes a prediction for the cardinality distribution.

PHD prediction

Every class of road users has its own motion characteristics. So when the exact class of an object is known, the corresponding motion model is selected for a more accurate prediction. However when there are uncertainties about the object's class, the incorrect motion model could be selected, resulting in a less accurate prediction. For example, when the appearance of a cyclist is very similar to that of pedestrian, it could be classified as being 50% pedestrian, 49% cyclist and 1% car. In that case the cyclist is in the original approach classified as being a pedestrian and the incorrect motion model corresponding to a pedestrian is selected.

In the RCA-CPHD approach, this is prevented by including the class uncertainties in the prediction step. For each GM component a duplicate per class o is created and the weights are factorized by the corresponding class probabilities $c(o)$. So for the GM component with a classification of 50% pedestrian, 49% cyclist and 1% car, three duplicate GM components are created with peaks of respectively 50%, 49% and 1% of the original weight. Subsequently for each duplicate a prediction is made according to the corresponding motion model. This principle is applied to both the surviving GM components as well as the birth GM components, resulting in the following prediction equation of the PHD:

$$\underbrace{v_{k|k-1}(x)}_{\text{predicted PHD}} = \underbrace{\sum_{o=1}^{O}}_{\substack{\text{class}\\\text{duplicates}}} \left(\underbrace{\sum_{j=1}^{J_{\gamma,k}(o)} c(o)^{(j)} w_{\gamma,k}^{(j)} \mathcal{N}(x; m_{\gamma,k}^{(j)}, P_{\gamma,k}^{(j)})}_{\text{PHD of birth objects}} + \underbrace{p_{S,k} \sum_{j=1}^{J_{k-1}(o)} c(o)^{(j)} w_{k-1}^{(j)} \mathcal{N}(x; m_{S,k|k-1}^{(j)}, P_{S,k|k-1}^{(j)})}_{\text{PHD of survived objects}} \right) \tag{4.1.3}$$

where

- $m_{S,k|k-1}^{(j)} = F(o) m_{k-1}^{(j)}$
- $P_{S,k|k-1}^{(j)} = Q(o) + F(o) P_{k-1}^{(j)} F^T(o)$

The prediction equation is very similar to the equation used in the CMM-PHD of Meissner et al. [50] discussed in section 3.4.2. Meissner et al. includes the classification probabilities in the Markov transition matrix of the MM-PHD framework. The Markov transition probability is then used in the prediction equation as a factor of switching from one class to another. Even though this results in the same predicted PHD, the approach of including the classification probabilities direct into the prediction equations is more intuitive according to the writer.

All the duplicates get the same label as their original GM component to retain the information which duplicates belong together. This gives the possibility to merge the duplicates after the update back to a single GM component again:

$$l_{+}^{(j)}(o) = l^{(j)} \tag{4.1.4}$$

The class duplicates also get the same class probability vector of their origin GM component. This gives the possibility to construct a likelihood between the prediction and measurements based on classification during the gating and update step.

$$c_{+}^{(j)}(o) = c^{(j)} \tag{4.1.5}$$

Cardinality Distribution Prediction

The duplicates have no influence on the prediction of the cardinality distribution. Therefore the prediction equation of the cardinality distribution is the same as the single model CPHD discussed in section 3.3.5 and is estimated by combining the distribution of birth objects and surviving objects [78]:

$$\underbrace{p_{k|k-1}(n)}_{\text{predicted cardinality}} = \underbrace{\sum_{j=0}^{n}}_{\substack{\text{sum of all}\\ \text{combinations}\\ \text{that make n}}} \underbrace{p_{\Gamma,k}(n-j)}_{\substack{\text{probability of n-j}\\ \text{birth objects}}} \underbrace{\sum_{l=j}^{\infty} p_{k-1}(l) \binom{l}{j} p_{s,k}^{j} (1-p_{s,k})^{l-j}}_{\text{probability of j surviving objects}} \tag{4.1.6}$$

where

- $p_{\Gamma,k}(\cdot)$ is the cardinality distribution of births at time k
- $\binom{l}{j}$ is the binomial coefficient with $\binom{l}{j} = \frac{l!}{j!(l-j)!}$
- p_{k-1} is the distribution of the previous time step t_{k-1}

4.1.3 Gating

The predicted PHD and cardinality distribution is updated after receiving new measurements from the sensor. This is done by looking at the likelihood of each measurement being generated by each GM component. The measurement dataset, however, also contains clutter and new born objects. To restrict the number of measurement to GM component comparisons, a measurement validation procedure called *gating* is used. In this step the measurements that are clearly not generated by one of the GM components are filtered out and labelled as potential new born objects.

Gating based on kinematic states

In the gating procedure the kinematic states of the GM components are compared with the measurements. The measurements that are more different in kinematic state to all GM components than a certain threshold are in the original gating procedure subsequently filtered out. The difference in states is determined by means of the normalized Euclidean distance. This Euclidean distance allows to compare all variables of the states and their corresponding covariances with those of the measurements. Given the state $x^{(j)}$ with corresponding covariance matrix $S^{(j)}$ of a GM component and a measurement z, the normalized Euclidean distance is as follows:

$$D(x^{(j)}, z) = \sqrt{\sum_{i=1}^{|z|} \frac{(x_i^{(j)} - z_i)^2}{(S_i^{(j)})^2}} \tag{4.1.7}$$

So when the gating threshold is τ_D, the measurements within the gate of GM component j are obtained using:

$$Z(\tau_D) = \left\{ z : D(x^{(j)}, z) \le \tau_D \right\} \tag{4.1.8}$$

Gating based on kinematic states and classification

In the RCA-CPHD the gating procedure is not only based on the Euclidean distance of the kinematic states, but also on the similarity in classification. This is based on the idea that if a measurement is within the Euclidean distance of GM component, but is totally different in classification, then it is probably not generated by that GM component. For example if a new appearing car arrives within the Euclidean distance of an already existing bicycle, then in the original procedure the measurement of the car is used within the update step. In the RCA-CPHD this measurement is ignored in the update step and is labelled as new appearing object. This has the advantage that the track of the car is faster initiated. Besides its measurement is not disturbing the track of the already existing bicycle, what might prevent a reduction in tracking accuracy in the bicycle track and the occurrence of classification errors. The gating procedure of the RCA-CPHD with both the kinematic state gating and the class similarity gating is as follows:

$$Z(\tau_D \cap \tau_c) = \left\{ z : D(x^{(j)}, z) \leq \tau_D \quad \cap \quad C(c^{(j)}, z_c) \geq \tau_c \right\} \tag{4.1.9}$$

Where $C(c^{(j)}, z_c)$ is the similarity in classification, z_c the classification vector corresponding to the measurement and τ_c the corresponding threshold. The similarity in classification is determined by the dot product of both classification vectors:

$$C(c^{(j)}, z_c) = \frac{c^{(j)} \cdot z_c}{|c^{(j)}||z_c|} \tag{4.1.10}$$

So if the classification probabilities of the GM component $c^{(j)}$ and the classification probabilities of the measurement z_c are similar, the angle between the vectors is zero and the dot product $C(c^{(j)}, z_c)$ is one. However if they are totally different, there the angle between both classification vectors is large and the resulting $C(c^{(j)}, z_c)$ will be zero. A visualization of this principle is shown in figure 5.15.

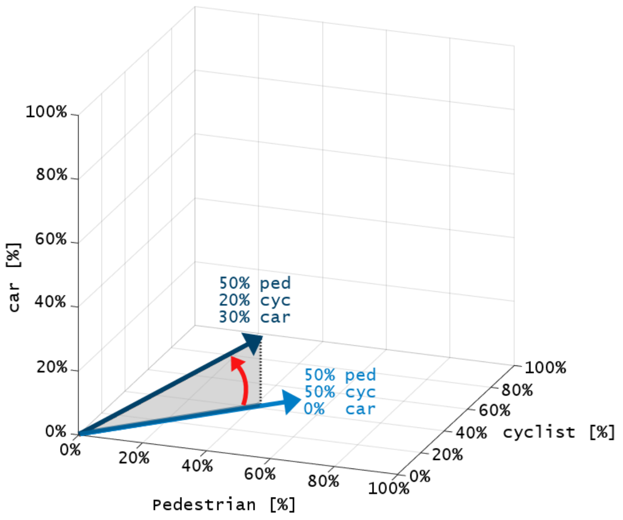


Figure 4.2: The likelihood in classification is based on the dot product of the GM component's class vector and the measurement's class vector. The size of the angle between both vectors is thus a measure for the class-likelihood. In the example the class likelihood is 0.8 for a measurement classification of 50% pedestrian, 50% cyclist and 0% car and GM component classification of 50% pedestrian, 20% cyclist and 30% car.

Birth Model

Measurements that are clearly not generated by an existing GM component, due to their deviation in classification and kinematic state, are filtered out in the gating procedure. If a measurement is not generated by an existing object, it means it is either a new appearing object or a false alarm. Therefore these measurements are initialized as birth GM components with a new unique track label and a relative low weight. If it actually turns out to be a new appearing object, then it will generate a new measurement in the next time step with a similar state and classification as the prediction. In that case the weight will increase in the update step and a new track is created. If it turns out to be a false alarm, no measurement will match in the update step and the pruning step at the end of the recursion will eliminate the GM component.

4.1.4 Update

After receiving the filtered measurements Z_k and corresponding classification probabilities $z_{c,k}$, the predicted PHD $v_{k|k-1}$ and predicted cardinality distribution $p_{k|k-1}$ are updated.

PHD update

In the original CPHD and MM-PHD, discussed in respectively section 3.3.5 and 3.4.1, the update is based on the likelihood in kinematic state of the measurement and GM component only. In the RCA-CPHD the update is based on the likelihood in kinematic state and the likelihood in classification together. After all, it is less likely that a measurement with a high probability of being a certain class is generated by an object with a high probability of being another class. Since in the original filter only the likelihood in kinematic state is taken into account, it is for example possible that the weight of a pedestrian GM component is increased by a car measurement if they are similar in states. However for the same situation in the RCA-CPHD, the likelihood in state could be high, but the likelihood in classification is at the same time very low. So the total likelihood, based on the product of both likelihoods, would result in a low likelihood and therefore a decrease in the weight of the pedestrian GM component corresponding to the car measurement. This provides especially an advantage in situations where two objects with different classification approaching each other closely, occlusion and in case of an incorrect classification of a measurement.

The update equation of the PHD is a multiple model extension of the single model CPHD update equation 3.3.5 and the classification likelihood is incorporated in the total measurement likelihood. In the multiple model equation, the total number of GM components is the sum of the duplicates of all the classes. Including this sum of classes in the single CPHD update equation results in the update equation of the RCA-CPHD:

$$\underbrace{v_k(x)}_{\text{updated PHD}} = \underbrace{\sum_{o=1}^{O}}_{\substack{\text{class}\\\text{duplicates}}} \left(\underbrace{(1-p_{D,k})\frac{\langle \Upsilon_k^1[w_{k|k-1}, Z_k], p_{k|k-1}\rangle}{\langle \Upsilon_k^0[w_{k|k-1}, Z_k], p_{k|k-1}\rangle} v_{k|k-1}(x)}_{\text{PHD missed detections}} + \underbrace{\sum_{z\in Z_k} \sum_{j=1}^{J_{k|k-1}(o)} w_k^{(j)}(z)\mathcal{N}(x; m_k^{(j)}(z), P_k^{(j)})}_{\text{PHD update prediction}} \right) \tag{4.1.11}$$

With the same components as in the single model equation 3.3.5, except the GM component weight $w_k^{(j)}$ and weight vector $w_{k|k-1}$ containing all the GM component weights. As a result of the class-duplicates, the weight vector $w_{k|k-1}$ has become set of all duplicates:

- $w_{k|k-1} = [w_{k|k-1}^{(1,o_1)} ... w_{k|k-1}^{(J_{k|k-1},O)}]$

The likelihood in kinematic states $q_s(z)$ is found in the original filter in the update equation of the weight and in the elementary symmetric function used in the equation for the likelihood in cardinality distribution. The likelihood in classification $q_c(z_c)$ is incorporated by replacing this likelihood in kinematic state by the product of both likelihood $q_s(z)q_c(z_c)$. So by applying the product of both likelihoods at these positions, the update equation for the weight is obtained:

$$w_k^{(j)}(z) = p_{D,k}\, \underbrace{q_{s,k}^{(j)}(z)}_{\substack{\text{likelihood}\\\text{states}}}\, \underbrace{q_{c,k}^{(j)}(z_c)}_{\substack{\text{likelihood}\\\text{classification}}}\, \underbrace{\frac{\langle \Upsilon_k^1[w_{k|k-1}, Z_k\{z\}], p_{k|k-1}\rangle}{\langle \Upsilon_k^0[w_{k|k-1}, Z_k], p_{k|k-1}\rangle}}_{\substack{\text{Likelihood of measurements}\\\text{given the cardinality distribution}}} w_{k|k-1}^{(j)} \tag{4.1.12}$$

And the following equation for the likelihood in cardinality

$$\frac{\langle \Upsilon_k^1[w_{k|k-1}, Z_k], p_{k|k-1}\rangle}{\langle \Upsilon_k^0[w_{k|k-1}, Z_k], p_{k|k-1}\rangle} \tag{4.1.13}$$

with

- $\Upsilon_k^u[w, Z](n) = \sum_{j=0}^{min(|Z|,n)} (|Z|-j)! p_{K,k}(|Z|-j) \frac{n!}{(n-j+u)!} \times \frac{\langle 1-p_{D,k}, v\rangle^{n-(j+u)}}{\langle 1, w\rangle^{j+u}} e_j(\Xi_k(w, Z))$
- $\Xi_k(w_{k|k-1}, Z) = \{\frac{\langle 1,\kappa_k\rangle}{\kappa_k(z)} p_{D,k} w_{k|k-1}^T q_{s,k}(z) q_{c,k}(z_c) \quad : \quad z \in Z\}$

Label update

In the update step a GM component is created per predicted GM component and measurement. So if there are N birth and surviving GM components before the prediction step and O classes, the number of GM components after the prediction step is $N \cdot O$ with each GM component of N having an unique track number. For example, if there is one GM component at the start of the prediction step, the labels after the prediction step will be:

	GM_{ped}	GM_{cyc}	GM_{car}
Track label	1	1	1
Class label	0	1	2

When receiving Z measurements from the sensor, each measurement is compared with each predicted GM component and the predicted GM components are propagated with a low weight to deal with missed detections. So the number of GM components after the update step is increased to $N \cdot O \cdot (Z+1)$. Since the class-duplicates are merged after the measurement update, also a label containing the coupled measurement is created. For the given example and two measurements arriving from the sensor, the labels will become:

	GM_{ped}^v	GM_{cyc}^v	GM_{car}^v	GM_{ped}^{z1}	GM_{cyc}^{z1}	GM_{car}^{z1}	GM_{ped}^{z2}	GM_{cyc}^{z2}	GM_{car}^{z2}
Track label	1	1	1	1	1	1	1	1	1
Class label	0	1	2	0	1	2	0	1	2
Measurement label	0	0	0	1	1	1	2	2	2

After the measurement update the class duplicates are merged according to the procedure explained in the next paragraph. After this procedure, the labels for the given example will be reduced to:

	GM^v	GM^{z1}	GM^{z2}
Track label	1	1	1
Measurement label	0	1	2

In the merging and pruning step GM, components with a weight lower than the pruning threshold will be removed and similar GM components are merged in order to truncate the increasing number of GM components. Subsequently from the remaining GM components, the largest weighted GM component of each track label is selected as the contribution of the corresponding track. The merging/pruning step and track extraction step will be explained in more depth in section 4.1.5 and 4.1.6.

Classification update

The class vector of the GM component is updated with classification probabilities corresponding to the measurement. However the classification of the GM component and the measurement could be contradictory. Therefore Dempster-Shafer theory of evidence is used, which is able to handle contradictory measurements in a natural way [50]. Dempster's rule of combination is as follows:

$$c_{update} = \frac{\sum\limits_{c_{gm} \cap c_z = A} c_{gm} c_z}{1 - \sum\limits_{c_{gm} \cap c_z = \emptyset} c_{gm} c_z} \forall A \in 2^{\Omega} \tag{4.1.14}$$

However Dempster-Shafer theory of evidence suffers from Zadeh's paradox. This paradox happens when two contradicting sources have a high probability and zero probability for the same classes but adversely. If both sources have a very low probability for the third class, this class becomes 1 due to the only overlap on this unrealistic class. For example when considering the class vector of the GM component to be 90% pedestrian, 10% cyclist and 0% car and the class vector of the measurement 0% pedestrian, 10% cyclist and 90% car. The multiplication of both vectors is as follows

	GM →	ped	cyc	car
	z↓	0.9	0.1	0
ped	0	0	0	0
cyc	0.1	0.09	0.01	0
car	0.9	0.81	0.09	0

According to Dempster's rule of combination, the updated class vector becomes subsequently:

$$\begin{aligned} c_{ped} &= \frac{0}{1 - 0.09 - 0.81 - 0.09} = 0\% \\ c_{cyc} &= \frac{0.01}{1 - 0.09 - 0.81 - 0.09} = 100\% \\ c_{car} &= \frac{0}{1 - 0.09 - 0.81 - 0.09} = 0\% \end{aligned} \tag{4.1.15}$$

So due to the contradictory zeros and the only overlap in the cyclist class, the probability of being a cyclist becomes 100%. Which is an unrealistic outcome, since both class vectors only have 10% probability for being a cyclist. This Zadeh's paradox is solved by preventing a zero assignment to a class by discounting each class vector [52]. The discounting factor introduces the probability that the classification is incorrect and unknown, with means that there is no longer a zero assignment. The discounting factor can therefore be interpreted as a confidence in the classification. For example when taking a discounting factor of 0.9 for both classifications, the multiplications of both vectors is as follows

	GM →	ped	cyc	car	ped/cyc/car
	z↓	0.81	0.09	0	0.10
ped	0	0	0	0	0
cyc	0.09	0.073	0.008	0	0.005
car	0.81	0.656	0.073	0	0.041
ped/cyc/car	0.10	0.081	0.009	0	0.010

The updated class probabilities using Dempster's rule of combination becomes in this case:

$$\begin{aligned} c_{ped} &= \frac{0.081}{1-0.073-0.656-0.073} = 41\% \\ c_{cyc} &= \frac{0.01}{1-0.073-0.656-0.073} = 13\% \\ c_{car} &= \frac{0}{1-0.073-0.656-0.073} = 41\% \\ c_{ped/cyc/car} &= \frac{0.081}{1-0.073-0.656-0.073} = 5\% \end{aligned} \tag{4.1.16}$$

Normalizing for the classes gives subsequently the updated class vector, which is in this example 43% pedestrian, 14% cyclist and 43% car.

Merging class duplicates

In the prediction step, the GM components were duplicated per class, in order to keep the uncertainty of the classification in the filter. Even though a GM component is split up in duplicates, they still represent only one object at maximum together. So after the update step, the duplicates are merged together using the stored information in their labels. Another implementation, as applied in the CMM-PHD of Meissner et al., could be to leave this step to the general merging and pruning step at the end of the recursion. However, in case two or more duplicates do not meet the merging and pruning threshold, they both continue to exist in the next time step and might result in ghost objects. In addition, the merging results in less GM components, which can provide an advantage in computational load in the subsequent steps.

The weights of the duplicates were defined by factorizing the original weight by the probability of each class. So the merged weight is simply the sum of all the individual weights:

$$w_k^{(i)} = \sum_o w_{k,o}^{(i)} \tag{4.1.17}$$

The states and covariances are calculated in proportion to the weight. So the higher the weight of a certain class, the more the state of that class duplicate contributes to the state of the merged GM component:

$$\begin{aligned} m_k^{(i)} &= \frac{\sum\limits_o w_{k,o}^{(i)} m_{k,o}^{(i)}}{\sum\limits_o w_{k,o}^{(i)}} \\ P_k^{(i)} &= \frac{\sum\limits_o w_{k,o}^{(i)} \left(P_{k,0}^{(i)} + (m_k^{(i)} - m_{k,o}^{(i)})(m_k^{(i)} - m_{k,o}^{(i)})^T\right)}{\sum\limits_o w_{k,o}^{(i)}} \end{aligned} \tag{4.1.18}$$

Cardinality distribution update

The cardinality update equation is the same as the single model CPHD. However the likelihood of cardinality includes in the RCA-CPHD the likelihood in classification in the same way as the PHD update:

$$p_k(n) = \underbrace{\frac{\Psi_k^0[w_{k|k-1}, Z_k](n)}{\langle \Psi_k^0[w_{k|k-1}, Z_k], p_{k|k-1}\rangle}}_{\text{Likelihood of cardinality}} p_{k|k-1}(n) \tag{4.1.19}$$

with

- $\Upsilon_k^u[w,Z](n) = \sum_{j=0}^{min(|Z|,n)} (|Z|-j)! p_{K,k}(|Z|-j) \frac{n!}{(n-j+u)!} \times \frac{\langle 1-p_{D,k}, v\rangle^{n-(j+u)}}{\langle 1, w\rangle^{j+u}} e_j(\Xi_k(w,Z))$

- $\Xi_k(w_{k|k-1}, Z) = \{ \frac{\langle 1,\kappa_k \rangle}{\kappa_k(z)} p_{D,k} w_{k|k-1}^T q_{s,k}(z) q_{c,k}(z_c) \quad : \quad z \in Z\}$

4.1.5 Merging/Pruning/Capping

In order to truncate the increasing number of gaussians, similar GM components are merged and less important GM components are pruned. The used merging and pruning procedures are derived by Vo et al. [58] and Clark et al. [7].

Merging similar gaussians

Just as in the original MM-PHD is the similarity of two GM components measured using the Mahalanobis distance. However when similarity is only determined by the Mahalanobis distance between two GM components, there might be a chance that a pedestrian and a cyclist GM component are merged. So in the RCA-CPHD the similarity between GM components are not only determined by the Mahalanobis distance, but also in class similarity. The similarity in class is, just like in the gating step, determined by the dot product of the classification vectors. If the dot product is higher than the threshold τ_c and if the Mahalanobis distance is smaller than the threshold τ_{merge}, then the GM are assumed to be similar and merged:

$$(m^{(1)} - m^{(2)})^T \cdot (P^{(1)})^{-1} \cdot (m^{(1)} - m^{(2)}) \leq \tau_{merge} \quad \cap \quad C(c^{(1)}, c^{(2)}) \geq \tau_c \tag{4.1.20}$$

When a set of $Q = \{q = 1, ..., Q_{max}\}$ gaussians are within the Mahalanobis threshold from each other, they are merged according to:

$$\begin{aligned} w_k &= \sum_{q}^{Q_{max}} w_k^{(q)} \\ m_k &= \frac{\sum_{q}^{Q_{max}} w_k^{(q)} m_k^{(q)}}{\sum_{q}^{Q_{max}} w_k^{(q)}} \\ P_k &= \frac{\sum_{q}^{Q_{max}} w_k^{(q)} \left(P_k^{(q)} + (m_k - m_k^{(q)})(m_k - m_k^{(q)})^T \right)}{\sum_{q}^{Q_{max}} w_k^{(q)}} \\ c_k &= \frac{\sum_{q}^{Q_{max}} w_k^{(q)} c_k^{(q)}}{\sum_{q}^{Q_{max}} w_k^{(q)}} \end{aligned} \tag{4.1.21}$$

Pruning

To prevent a large number of GM components, less important GM components are pruned. If the weight of a GM component is smaller than the pruning threshold τ_{prune}, than the GM component is supposed to be unimportant and pruned:

$$w^{(i)} < \tau_{prune} \tag{4.1.22}$$

However after pruning the less important GM components, the sum of the weights of the remaining GM components is less than before pruning. Since the sum of the objects represent the number of objects J, the remaining GM components are normalized:

$$J_{new} = J - J_{prune}$$
$$w^{(i)} = \frac{\sum_{j=1}^{J} w^{(j)}}{\sum_{j=1}^{J_{new}} w^{(j)}} \cdot w^{(i)} \tag{4.1.23}$$

4.1.6 Track extraction

In the original PHD the object states are estimated by taking the N GM components with the largest weights, with N the estimated cardinality as discussed in section 3.3.2. Clark et al. [7] implemented an approach to improve the continuity of the individual tracks by using track labels. All the largest weighted GM components per defined track label with a weight larger than a threshold of 0.5 are taken as the estimated objects.

The same approach is applied to the RCA-CPHD. However the RCA-CPHD is based on the CPHD and has therefore a more accurate cardinality estimate than the PHD. So instead of using a threshold, the RCA-CPHD is using the cardinality estimate of the CPHD. First all the highest weighted GM components of each track label are selected and subsequently the N highest weighted tracks are selected as the state estimates, with N the estimated cardinality.

4.1.7 Implementation of the RCA-CPHD

The RCA-CPHD has been implemented adopting Matlab code of the CPHD filter from Vo et al. [73]. Besides both the CMM-PHD and the MM-CPHD are also implemented adopting the code from Vo et al. [73] of the CPHD filter and used to compare the performance of the RCA-CPHD with the performance of CMM-PHD and MM-CPHD. The results of this evaluation are discussed in chapter 5.

4.2 Motion Models

As seen in the previous section, the RCA-CPHD uses a prediction based on the different motion characteristics per class. In this thesis the main three classes of road users are considered, namely pedestrian, cyclist and car. The RCA-CPHD is however able to deal with more and different classes if required by the application. A pedestrian is very agile and is able to move sidewards, making a pedestrian able to instantly change direction. Consequently, the direction of motion of a pedestrian is independent on the orientation [50][49][51]. Therefore the Linear Constant Velocity (LCV) motion model is used for pedestrians.

Cyclists and cars have however constraints in their motion ability and are not able to move laterally. Their motion is dependent on the orientation and is therefore commonly modelled using a Constant Turn Rate and Velocity (CTRV) model [53][50][35].

4.2.1 Constant Velocity model

The constant velocity model (CV) models the motion as a linear straight motion with fairly constant velocity and the acceleration almost zero. However, in practice pedestrians can actually have accelerations. Therefore a small acceleration is included as a white noise process w_k. This results in the following motion model for pedestrians:

$$\begin{bmatrix} x_{k|k-1} \\ \dot{x}_{k|k-1} \\ y_{k|k-1} \\ \dot{y}_{k|k-1} \\ z_{k|k-1} \\ \dot{z}_{k|k-1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ \dot{x}_{k-1} \\ y_{k-1} \\ \dot{y}_{k-1} \\ z_{k-1} \\ \dot{z}_{k-1} \end{bmatrix} + w_k \tag{4.2.1}$$

4.2.2 Constant Turn Rate and Velocity model

The motion of vehicles and cyclist in the $x-y$ plane are dependent on the orientation. This is implemented in the Constant Turn Rate and Velocity model, which presumes that an object moves with fairly constant velocity $\dot{x}$ and fairly constant angular turn rate w. For the z-axis a constant velocity model is used. This results in the following motion model for cyclists and cars:

$$\begin{bmatrix} x_{k|k-1} \\ \dot{x}_{k|k-1} \\ y_{k|k-1} \\ \dot{y}_{k|k-1} \\ z_{k|k-1} \\ \dot{z}_{k|k-1} \end{bmatrix} = \begin{bmatrix} 1 & \frac{\sin\omega\Delta t}{\omega} & 0 & -\frac{1-\cos\omega\Delta t}{\omega} & 0 & 0 \\ 0 & \cos\omega\Delta t & 0 & -\sin\omega\Delta t & 0 & 0 \\ 0 & \frac{1-\cos\omega\Delta t}{\omega} & 1 & \frac{\sin\omega\Delta t}{\omega} & 0 & 0 \\ 0 & \sin\omega\Delta t & 0 & \cos\omega\Delta t & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ \dot{x}_{k-1} \\ y_{k-1} \\ \dot{y}_{k-1} \\ z_{k-1} \\ \dot{z}_{k-1} \end{bmatrix} + w_k \tag{4.2.2}$$

4.3 Measurement Models

The states of the objects in the environment are estimated using the observations received by the sensors. Each different type of sensor has its own output of the perceived environment, which is not always the same as the quantities used in the state vector. For instance the radar measures the distance R, the velocity $\dot{R}$ the polar angle θ and azimuth angle ψ of the object with respect to the radar. However the camera, after processing the image by a classifier, outputs the position-coordinates u and v of the object in the captured image frame and a corresponding classification c.

$$\mathbf{z}_{radar} = \begin{bmatrix} r \\ \theta \\ \psi \\ \dot{r} \end{bmatrix} \qquad \mathbf{z}_{camera} = \begin{bmatrix} u \\ v \\ c \end{bmatrix}$$

In order to compare the states with the observations from the sensor, an *observation model* or *measurement model* for each type of sensor is required. This measurement model describes how the quantities of the observation **z** are related to the quantities used in the object states x.

$$\mathbf{z} = \mathbf{h}(x)$$

Besides the observations may include noise or an uncertainty, which is different for each type of sensor either. This is taken into account in the measurement model by the covariance matrix R.

4.3.1 Radar

The states of the objects in the environment are described according to the Cartesian coordinate system of the car. This coordinates system, the car reference frame, is for the DAVI-vehicle located at the center of the rear axis as shown in the schematic overview in figure 4.3. The radar coordinate system is however a spherical coordinate system and the radar sensor is in general not located at the location of the car reference frame. So in order to compare the states with the observations, the states have to transformed from the cartesian car reference frame to the spherical radar coordinate system. This is done by first rotating and translating the states to the radar's coordinate system in Cartesian coordinates and subsequently these coordinates are transformed into spherical coordinates.

$$\begin{bmatrix} x_{car} \\ y_{car} \\ z_{car} \end{bmatrix} \xrightarrow[\text{(Cartesian)}]{\text{Car to Radar}} \begin{bmatrix} x_{radar} \\ y_{radar} \\ z_{radar} \end{bmatrix} \xrightarrow{\text{Cartesian to spherical}} \begin{bmatrix} r \\ \theta \\ \psi \end{bmatrix}$$

4.3.2 Aligning coordinate system of Car to Radar

The first step is aligning the car reference frame to the coordinate system of the radar. The radar is in general not located at the position of the car reference frame and often rotated. So in order to align the car reference frame with the coordinate system of the radar, a rotation and translation is applied. When the radar sensor is mounted onto the vehicle with the angles α,β and γ from respectively the x-axis, y-axis and z-axis with respect to the car reference frame, the rotation matrix becomes

$$\mathbf{R} = \mathbf{R}_x \cdot \mathbf{R}_y \cdot \mathbf{R}_z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

And when te radar is located at position $\mathbf{T}_{radar} = [\Delta x \; \Delta y \; \Delta z]'$ with respect to the car reference frame, the total transformation matrix become

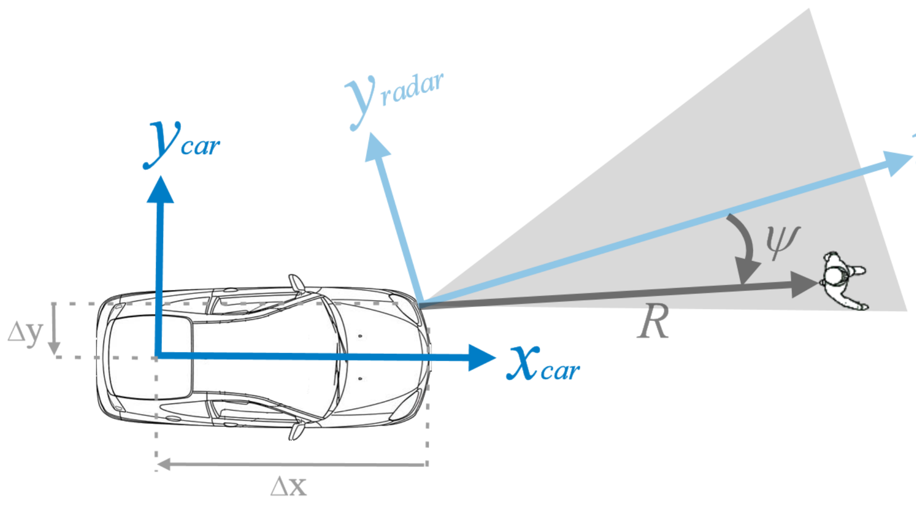


Figure 4.3: In order to use the observations for state estimation, the states have to be rotated and translated to the radar coordinates system and subsequently transformed to spherical coordinates.

$$\begin{bmatrix} x_{car} \\ y_{car} \\ z_{car} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & -\mathbf{T}_{radar} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{radar} \\ y_{radar} \\ z_{radar} \\ 1 \end{bmatrix}$$

The states are however given with the velocities as well. Even though they are not used in the actual comparison with the observation, they have to be included into the measurement model as well. Rearranging the transformation matrix results in the measurement model of the radar.

$$\begin{bmatrix} z_{x,radar} \\ z_{y,radar} \\ z_{z,radar} \end{bmatrix} = \begin{bmatrix} \mathbf{R} & -\mathbf{T}_{radar} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ z \\ \dot{z} \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{R}_{11} & 0 & \mathbf{R}_{12} & 0 & \mathbf{R}_{13} & 0 & \Delta x \\ \mathbf{R}_{21} & 0 & \mathbf{R}_{22} & 0 & \mathbf{R}_{23} & 0 & \Delta y \\ \mathbf{R}_{31} & 0 & \mathbf{R}_{32} & 0 & \mathbf{R}_{33} & 0 & \Delta z \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ z \\ \dot{z} \\ 1 \end{bmatrix}$$

Cartesian coordinates to Spherical coordinates

Since the radar measures the objects in the environment in spherical coordinates, the next step would be to transform the states from Cartesian to spherical coordinates. This step however would make the measurement model non-linear. An extended Kalman filter could be used to linearize the non-linear measurement model for each time step.

An easier way is however to transform the measurements first from spherical to Cartesian and subsequently use these measurements to compare with the transformed states in the Cartesian radar coordinate system, resulting from in the linear measurement model of 4.3.2. When the sensor measures the the range r, polar angle θ and azimuth angle ψ, the measurements in Cartesian coordinates become

$$\begin{aligned} z_{x,radar} &= r \cdot \cos\theta \cdot \cos\psi \\ z_{y,radar} &= r \cdot \cos\theta \cdot \sin\psi \\ z_{z,radar} &= r \cdot \sin\theta \end{aligned}$$

Measurement noise

The process noise takes into account the inaccuracy of the radar sensor. For the used radar sensor in the DAVI vehicle, the ARS309-2, the process noise is dependent on the measured range r of the object and therefore different for every object in the environment.

$$\begin{aligned} \sigma_r &= 0.015 \cdot r \text{ [m]} \\ \sigma_\psi &= 0.1 \text{ [deg]} \end{aligned} \tag{4.3.1}$$

Since the measurements are compared with the states in Cartesian coordinates to prevent a non-linear measurement model, the process noise should be transformed to the Cartesian coordinate system as well. This is done using the following elements of the covariation matrix R [2].

$$\begin{aligned} R_{11} &= var(x) = r^2\sigma_\psi^2 \sin\psi^2 + \sigma_r^2 \cos\psi^2 \\ R_{22} &= var(y) = r^2\sigma_\psi^2 \cos\psi^2 + \sigma_r^2 \sin\psi^2 \\ R_{12} &= cov(x,y) = (\sigma_r^2 - r_m^2\sigma_\psi^2) \cos\psi \sin\psi \end{aligned} \tag{4.3.2}$$

These elements are however only valid when the following condition is met [2]

$$\frac{r\sigma_\psi^2}{\sigma_r} < 0.4 \tag{4.3.3}$$

4.3.3 Camera

The camera captures an image frame of the environment at each time step. This image frame is subsequently processed by a classifier to recognize the objects in the environment. This classifier outputs the estimated positions u and v of the objects, respectively the horizontal center and vertical bottom, and its classification C.

Measurement model

The states are described by the position and velocity of an object with respect to the car reference frame. The camera however uses the 2D coordinates u and v which are relative to the camera coordinate system. So in order to compare the states and measurements, the positions of the states are first transformed to the camera reference frame and subsequently a projection is made from the 3D coordinates into the 2D image plane.

$$\begin{bmatrix} x_{car} \\ y_{car} \\ z_{car} \end{bmatrix} \xrightarrow[\text{(extrinsic)}]{\text{Car to Camera}} \begin{bmatrix} x_{camera} \\ y_{camera} \\ z_{camera} \end{bmatrix} \xrightarrow[\text{(intrinsic)}]{\text{Camera to Image plane}} \begin{bmatrix} u \\ v \end{bmatrix} \tag{4.3.4}$$

Car reference frame to Camera coordinate system

First the position of the states are transformed to the camera coordinate system. This is done the same way as done with the radar, using a translation and rotation matrix. This results in the transformation matrix, also called the *extrinsic matrix*, where $\mathbf{T}_{camera}$ is the position of the camera with respect to the car reference frame

$$\begin{bmatrix} x_{camera} \\ y_{camera} \\ z_{camera} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & -\mathbf{T}_{camera} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{car} \\ y_{car} \\ z_{car} \\ 1 \end{bmatrix}$$

with

$$\mathbf{R} = \mathbf{R}_x \cdot \mathbf{R}_y \cdot \mathbf{R}_z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}_{camera} = \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix}$$

Camera coordinate system to Image plane

The next step is to transform the 3D camera coordinates into the 2D image plane coordinates. This is done using the focal length of the camera:

$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} z_{x,camera} \\ z_{y,camera} \\ z_{z,camera} \\ 1 \end{bmatrix} \tag{4.3.5}$$

The resulting coordinates are still in 3D. Dividing u' and v' by w' would result in a 2D projection in homogeneous coordinates. This step would make the measurement nonlinear as well. However note that if $w' \neq 0$, then

$$(f\frac{u'}{w'}, f\frac{v'}{w'}, 1) \equiv (fu', fv', w')$$

This equivalence makes it possible to write the 2D project in the form of equation 4.3.5 [34]. Since the camera coordinates are in meters and the image coordinates are in pixels, a scaling has to be applied as well. These scaling factors m_x and m_y are dependent on the resolution of the image. Besides the intersection of the optical axis with the image plane, also called the *principal point*, corresponds in general not exactly to the center of the image. This is corrected with a translation x_0 and y_0. At last a difference in alignment of the camera and image plane could be compensated with a factor s compensating the skew. Putting all these factors in one matrix results in the *camera calibration matrix* or the *intrinsic matrix*.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} m_x f & s & m_x x_0 & 0 \\ 0 & m_x f & m_x y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_{camera} \\ y_{camera} \\ z_{camera} \\ 1 \end{bmatrix}$$

Combining both the extrinsic and intrinsic transformation results in the measurement model of the radar

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} m_x f & s & m_x x_0 & 0 \\ 0 & m_x f & m_x y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{P}_{cam} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ z \\ \dot{z} \\ 1 \end{bmatrix}$$

5 Evaluation

In this chapter, the performance of the proposed RCA-CPHD, defined by the tracking accuracy and the classification accuracy, is evaluated. The evaluation is done in a two step approach; verification and validation. First a verification is done by comparing the performance of the RCA-CPHD with the MM-CPHD and the CMM-CPHD on simulated data from one radar and one camera. A comparison with the MM-PHD and CMM-CPHD would not be a fair comparison, given that the CPHD has an improved cardinality with respect to the PHD filter [78]. Therefore an implementation has been made for the MM-CPHD and CMM-CPHD by adopting the the MM-PHD and CMM-PHD and implemented in Matlab by adopting the code of the CPHD from Vo et al. [73]. Two different scenarios are simulated and the performance is determined by averaging over 100 Monte Carlo runs.

Subsequently the performance is validated by comparing the performance of the RCA-CPHD with the MM-CPHD and the CMM-CPHD on six real world data experiments recorded with a stereo camera. The first two experiments are traffic situations recorded with a stereo camera on the DAVI-vehicle. Subsequently four experiments are evaluated selected from the KITTI database, a database containing camera images, laser scans, high precision GPS measurements and IMU accelerations recorded while driving on German roads [22].

In addition, the performance for both the simulations and the real world data scenarios are determined for two different classifiers. The two classifiers are the LRR classifier developed by Ghiasi et al. [25] and the SSD developed by Wei Liu et al. [36] and are described in more detail in section 2.4. The two classifiers differ in the way of dealing with classification uncertainty and this fundamental difference makes it interesting for taking into account both classifiers in the evaluation. Both the code of the LRR from Ghiasi et al. [25] and the code of the SSD from Wei Liu et al. [36] are adjusted to output the class probabilities for the three considered pedestrian, cyclist and car classes.

In section 5.1 the performance measures of the tracking accuracy and classification accuracy are discussed. Subsequently in section 5.2 the verification using the simulated data is discussed. An overview is given of the experiment setup, the used parameters and performance analysis followed by the verification conclusion. Then in section 5.3 the validation based on the six different traffic situations is discussed followed by the conclusion based on the validation results.

5.1 Performance Measures

In order to compare and evaluate the tracking performance of the different tracking algorithms, performance measures are required. The performance is evaluated by two different measures, namely the tracking accuracy and the classification accuracy. The tracking accuracy describes the accuracy of the state estimations and is measured by the deviation of the estimated tracks relative to the ground truth. Given that it is not only important to estimate the states, but also the corresponding classification, the classification accuracy is used as a second measure to determine the performance. The tracking accuracy is estimated by the deviation in the estimated classification of the tracks relative to ground truth classification.

5.1.1 Tracking accuracy

The tracking accuracy might be estimated by the absolute error between the state estimates of the tracks and the corresponding ground truth. However in case a part of the track is missing, for example due to an incorrect cardinality estimate, there will be no error between the ground truth and the missing part. That means an incorrect estimation of the cardinality would result in a lower error and thus an increase in performance. This is prevented by incorporating both the cardinality error and the localisation error in the tracking accuracy metric. A common used metric in object tracking that incorporates both is the Optimal SubPattern Assignment (OSPA) metric proposed by Schuhmacher et al. [70].

When considering the state estimates for the tracks at a certain time step $X_{tr} = \{x_{tr}^{(1)}, ..., x_{tr}^{(m)}\}$ and the corresponding ground truth vectors of the tracks $X_{gt} = \{x_{gt}^{(1)}, ..., x_{gt}^{(n)}\}$, then the OSPA distance is as follows:

$$d_p^c(X_{tr}, X_{gt}) = \left(\frac{1}{n}\left(\min_{\pi\in\Pi_n}\sum_{i=1}^{n} d_c(x_{tr}^{(i)}, x_{gt}^{(\pi(i))})^p + c^p(n-m)\right)\right)^{\frac{1}{p}} \tag{5.1.1}$$

With p the order, c the cut-off parameter and $d^{(c)}(x_{tr}^{(1)}, x_{(gt)}^{1})$ the distance between the state vectors calculated according to:

$$d^{(c)}(x_{tr}^{(1)}, x_{gt}^{(1)}) = \min(c, d(x_{tr}^{(1)}, x_{gt}^{(1)})) \tag{5.1.2}$$

Where $d(x_{tr}, x_{gt})$ is an arbitrary distance metric, as the Root Mean Square Error (RMSE), the Euclidean or Mahalanobis distance. In the evaluation the Euclidean distance is used. Since only the estimated location of the objects are considered, the Euclidean distance represents the absolute distance between the state estimate and ground truth.

Basically the OSPA distance at a certain time step is calculated by the sum of Euclidean distances between the tracks state estimates at that time step and the corresponding ground truth. If the state estimate of a track is missing, then that mismatch will be penalized by the cut-off parameter c. To prevent that an error between a state estimate vector and ground truth vector can have a greater influence on the OSPA distance than a cardinality mismatch, the Euclidean distance is limited by the cut-off parameter c [52].The order p sets the sensitivity to outliers. In the evaluation the order is one, so outliers have the same sensitivity with respect to the rest.

When looking only at the OSPA distance, it is not clear if the main contribution to the OSPA distance is due to a localisation error or due to a cardinality mismatch. So in order to get an insight of the main contribution to the OSPA distance, next to the OSPA distance also the OSPA localisation error and OSPA cardinality error are separately calculated:

$$\begin{aligned} d^{(c)}_{p,loc} &= \left(\frac{1}{n} \cdot \min_{\pi \in \Pi_n} \sum_{i=1}^{n} d_c(x^{(i)}_{tr}, x^{(\pi(i))}_{gt})^p \right)^{\frac{1}{p}} \\ d^{(c)}_{p,card} &= \left(\frac{1}{n} \cdot c^p(n-m) \right)^{\frac{1}{p}} \end{aligned} \tag{5.1.3}$$

5.1.2 Classification accuracy

The classification accuracy is evaluated with the Brier score, also known as the Mean Square Error (MSE) [6]. This measure is commonly used for assessing the reliability of classifiers, because it does not only give an insight in when an object is misclassified, but also whether this happens with a high or low probability [17][82]. The MSE is calculated by the mean square deviation of the estimated class with respect to the true class. So the average classification MSE per track at a certain time step for a set of track class estimates $C_{tr} = \{c^{(1)}_{tr}, ..., c^{(m)}_{tr}\}$ and corresponding ground truth $C_{gt} = \{c^{(1)}_{gt}, ..., c^{(n)}_{gt}\}$ is as follows

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (c_{gt}(o) - c_{tr}(o))^2 \tag{5.1.4}$$

Where $c_{tr}(o)$ is the estimated probability corresponding to the true class. So when the classification estimate is incorrect for all the tracks, the MSE will be 100%, while if the classification estimates for all tracks are correct, the MSE will be 0%.

In case an object is missed by the filter, for example due to a cardinality mismatch, the MSE for that track will be 0. In order to see if a low MSE is not caused by a cardinality mismatch, the absolute average cardinality error per track is also calculated and used as the second metric for the classification accuracy.

5.2 Verification using Simulated Data

The performance of the RCA-CPHD is compared to the performance of the MM-CPHD and CMM-CPHD. A comparison with the original CPHD would not be a fair comparison, given that the multiple model variant has already been proven to outperform the CPHD in tracking accuracy [60]. The same principle counts for a comparison with the MM-PHD instead of the MM-CPHD, since the CPHD filter has an improved cardinality with respect to the PHD filter [78]. Therefore the only decent performance comparison is one with the MM-CPHD. In the CMM-PHD the class uncertainty is also implemented in the prediction equation. Therefore it is also interesting to see what the performance of the RCA-CPHD will be compared to this approach. The cardinality distribution is also propagated in the CMM-PHD to prevent an unfair comparison due to the PHD "target death" problem [60].

The verification of the RCA-CPHD is done with two different scenarios. The first scenario is a simulation of a real life scenario for verification of the performance in realistic scenarios. Subsequently a second scenario is simulated for a more challenging scenario, with a higher manoeuvrability of the objects and higher classification uncertainty, mainly due to occlusion.

Experiment setup

The simulation data is created with PreScan simulation software and the generated data is derived from a virtual radar and camera operating at 5 hz. The virtual radar has been given the same specifications as the original Continental ARS 309-2 radar mounted on the Davi Vehicle [9]. The verification of the performance is done by averaging the simulation results of 100 Monte Carlo runs. Before each Monte Carlo run, a random noise is added to the generated sensor data to simulate the inaccuracy of the sensors. The random noise is analogous to the specifications of the sensors mounted on the DAVI-vehicle, resulting in a normal distributed random noise of $\sigma_x = \sigma_y = 0.8$ m for the radar data and $\sigma_u = \sigma_v = 5$ pixels for the classified camera data.

Asynchronous update The radar and camera sensor are operating at the same frequency, but are in reality almost never exactly synchronous. Also during recording is seen that sometimes a data package from a sensor is not arrived at all. As explained in section 3.1.4 the asynchronous data could be synchronized first and then used in the tracking filter or by asynchronously updating in the filter.

During the experiments, the filter recursion is done by asynchronously updating. As soon as a data package is received from the sensor, the filter recursion has been run according to the procedure discussed in section 4.1. The recursion is run with a time step equal to the difference between the arrival time of the previous package and the current package. The measurements received by the radar sensor are not containing classification probabilities. So in case of the radar measurements update, the classification probabilities of the GM components are not updated by the measurements and the current classifications are propagated. New appearing targets during the radar measurement update are initialized with equal class probabilities, so in this case 33% pedestrian, 33% cyclist and 33% car.

Ground truth In order to estimate the performance measures, the ground truth of the tracks is required. The ground truth is directly obtained from the PreScan simulation software.

Classifier The image frames from the camera are processed by both the LRR and SSD classifier. As explain in section 2.4, both classifier differ in the way of dealing with classification uncertainty, making it interesting for taking into account both classifiers. Therefore the 100 Monte Carlo simulations are run twice, once for the LRR and once for the SSD classifier. As explained in section 2.4 the output of the classifiers are adjusted to output the probability of all the observed classes. Besides the LRR classifier outputs the class-probabilities per pixel instead of bounding box. Therefore the bounding boxes are annotated and the corresponding class-probabilities are determined by the median of all the pixel class-probabilities in the bounding box.

Filter parameter setup

The RCA-CPHD, MM-CPHD and CMM-CPHD are for both simulations run with similar filter parameters. An overview of the parameters is given in table 5.1.

Detection probability The detection probability is dependent on the sensor modalities and the amount of occlusions in the recording. Since the SSD classifier is a detector, the detection results for persons, bikes and cars for a trained model of VOC2007,VOC2012 and COCO dataset [36] are used for determining detection probability P_D. The conversion of classification per pixel to bounding boxes for the LRR classifier is however done by annotation and the detection probability is therefore dependent on the annotation process. The average number of bounding boxes with respect to the actual number of objects in the ground truth is therefore used as the detection probability for the LRR classifier.

Clutter The conversion of classification per pixel to bounding boxes by annotation in the LRR classifier results in virtually no clutter. Therefore the amount of clutter is set to 0.1 clutter objects per scan. In the SSD classifier, however, clutter occurs more often. For example when a cyclist is seen as both a bike and a pedestrian. Therefore the clutter intensity for the SDD is higher with respect to the LRR and is set to 0.5 clutter objects per scan.

Measurement noise The measurement noise in the measurement models is considered to be the same as the applied noise on the generated virtual sensor data, namely $\sigma_x = \sigma_y = 0.8$ m for the radar data and $\sigma_u = \sigma_v = 5$ pixels for the camera data.

Parameters used in RCA-CPHD, MM-CPHD, CMM-CPHD		
Detection probability radar	$P_{D,radar}$	98%
Detection probability LRR classifier	$P_{D,LRR}$	89%
Detection probability SSD classifier	$P_{D,SSD}$	90%
Clutter intensity LRR	$\kappa_k(z)$	$0.1\,\frac{objects}{scan}$
Clutter intensity SSD	$\kappa_k(z)$	$0.5\,\frac{objects}{scan}$
Survival probability	P_s	98%
Merging threshold	τ_{merge}	1
Pruning threshold	τ_{prune}	$10^{-5}\ \frac{\text{number of objects}}{m^2}$
Capping threshold	τ_{cap}	400 GM components
Measurement noise radar	$\sigma_{x,radar}$	0.8 m
	$\sigma_{y,radar}$	0.8 m
Measurement noise camera	$\sigma_{u,camera}$	5 pixels
	$\sigma_{v,camera}$	5 pixels
Parameters used in RCA-CPHD only		
Class threshold	τ_c	25°
Class discount	$\alpha_{discount}$	0.95

Table 5.1: Summary of filter parameters used all simulations

5.2.1 Simulation 1

The first scenario is a simulated real life scenario. A representative urban traffic scenario has been selected from the KITTI database, a database containing camera data captured with an equipped vehicle driving on German roads [22].

The scenario has a duration of 10.5 seconds and has the weather conditions are sunny with no rain or dust. Four objects are considered in this simulation; 2 cyclists, 1 pedestrian and 1 car, with a maximum of 4 objects at the same time. The objects are moving within a range of 20 meter to 79 meter with respect to the ego-vehicle. During the scenario the ego-vehicle is parked in front of an intersection, with a footpath and bicycle path perpendicular to the front of the vehicle as shown in figure 5.1.

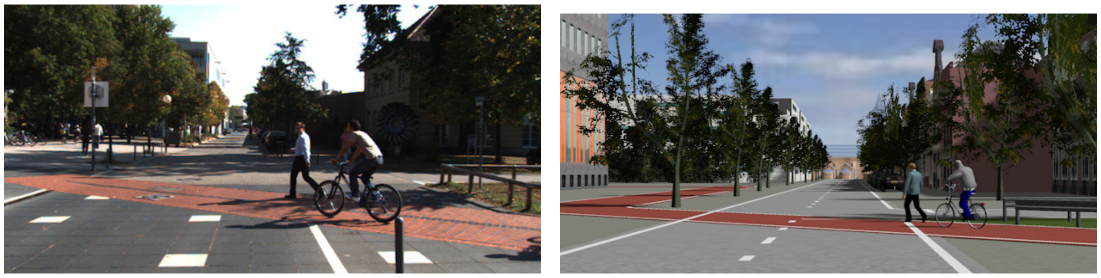


Figure 5.1: The first scenario is a representative urban traffic scenario selected from the KITTI database [22]

Course of scenario An overview of the course of the scenario is shown in figure 5.2. In the first 2.5 seconds a pedestrian is appearing in the FOV and walks in a straight line, accompanied by a car parked on the background. At about 2.5 seconds a cyclist arrives in the FOV and occludes first the parked car and crosses subsequently the pedestrian between 4.5 and 5 seconds as shown in figure 5.3a. Subsequently the pedestrian is occluding the car till 6 seconds as shown in figure 5.3b.

During the occlusions, the second cyclists appears at 5.5 seconds. Meanwhile the pedestrian manoeuvrers to the other side of the bike path. At 7.5 seconds the second cyclist occludes first the car and around 8.5 the cyclist crosses the pedestrian as shown in figure 5.3c and 5.3d. Subsequently both the pedestrian and cyclist are following the bike path without crossing and occluding till 10.5 seconds.

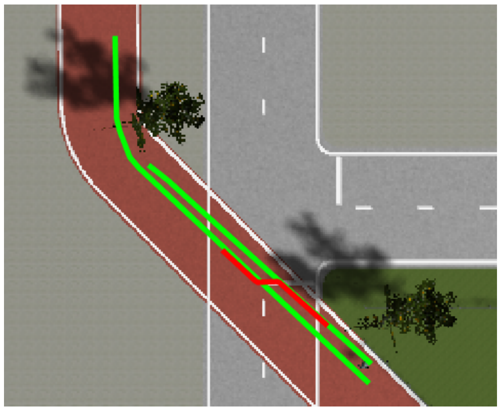
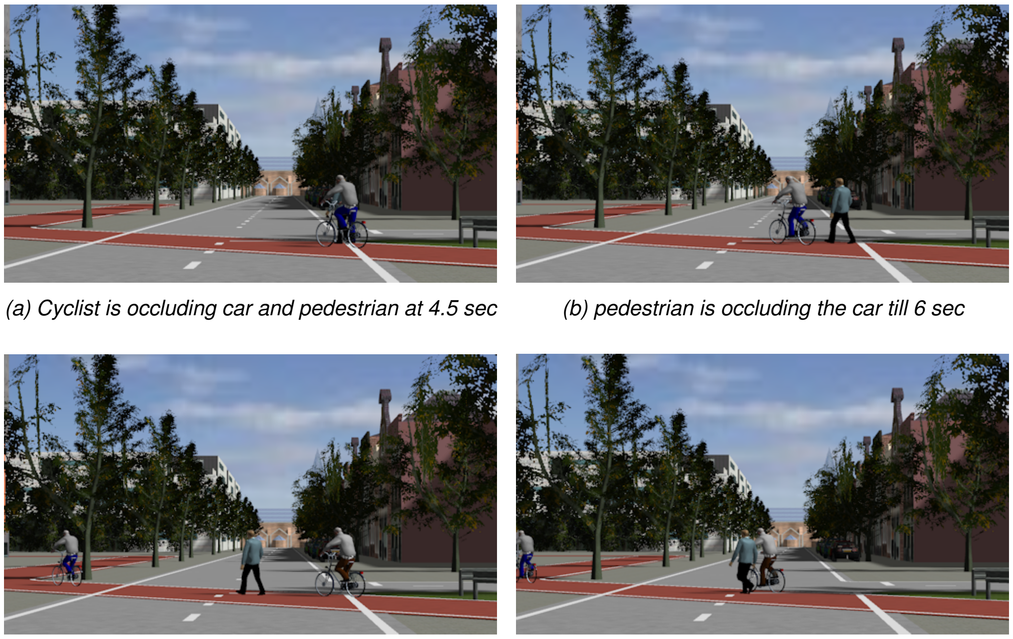


Figure 5.2: Overview of the course of simulation 1. During the scenario a pedestrian walks on the footpath and manoeuvrers halfway to the other side of the bicycle path. Meanwhile two cyclists are crossing the pedestrian as they cycle along the bicycle path. In the background is a vehicle parked, but is outside the range of the overview


(a) Cyclist is occluding car and pedestrian at 4.5 sec

(b) pedestrian is occluding the car till 6 sec

(c) cyclist is occluding car at 7.5 sec

(d) pedestrian is occluding cyclist at 8.5 sec

Figure 5.3: Objects are occluding and crossing each other multiple times during the simulation.

Analysis

The performance is evaluated by two different measures, namely the tracking accuracy and the classification accuracy explained in more depth in section 5.1. First the tracking accuracy results are analysed for both classifiers and subsequently the classification accuracy.

Tracking accuracy The tracking accuracy is measured by averaging the OSPA distance over 100 Monte Carlo runs for both the LRR classifier and SSD classifier. The result based on SSD classifier and radar is shown in figure 5.4 and for the LRR classifier and radar in figure 5.5.

In the first 2.5 seconds hardly any difference is observable between the filters. The pedestrian and car are both detected by all the filters and hardly any difference is observable between the filters. After about 2.5 seconds the first cyclist arrives in the FOV, resulting in an increase in cardinality error for all filters. The new appearing cyclist is on average faster observed by the RCA-CPHD, resulting in an average lower OSPA distance for the next second, mainly due to a lower cardinality error.

Between 4 and 6 seconds the classification uncertainty is high due to the successive occlusions. First the pedestrian is occluding the car at 4.5 seconds as shown in figure 5.3a and the pedestrian is subsequently occluding the car till 6 seconds as shown in figure 5.3b. During these occlusion the OSPA distance is significantly lower for the RCA-CPHD relative to the MM-CPHD and CMM-CPHD filters. The cyclist is within the distance range of the occluded pedestrian and is therefore in the MM-CPHD and CMM-CPHD used for updating the pedestrian's location. In the RCA-CPHD, however, the cyclist's class vector deviates more than the classification threshold and the cyclists measurement is not used for updating the pedestrian. Instead the RCA-CPHD uses the predicted location based on the motion model, which is during the occlusion more accurate.

The same principle occurs at 7.5 seconds where the second cyclist occludes the car and around 8.5 where the cyclist is occluded by the pedestrian as shown in figure 5.3c and 5.3d. The MM-CPHD and

CMM-CPHD are less able to deal with the successive occlusions, resulting in higher cardinality error during this period of time. The track of the pedestrian in also regularly mixed up by the cyclist's track during the crossing, with as a consequence that the track starts to wander and eventually dies out. This phenomenon is visible during 9.2 and 10.5 seconds with a higher OSPA distance with respect to the RCA-CPHD.

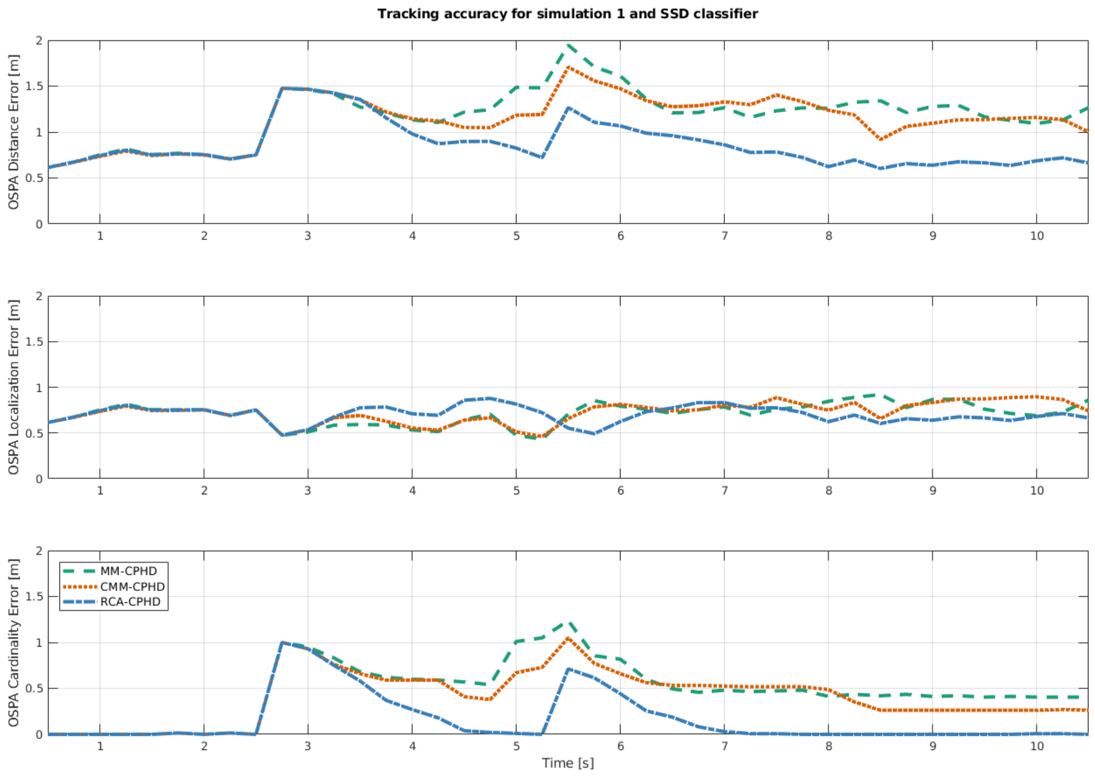


Figure 5.4: Tracking accuracy for 100 Monte Carlo runs for simulation 1 with measurements from a radar and SSD classifier

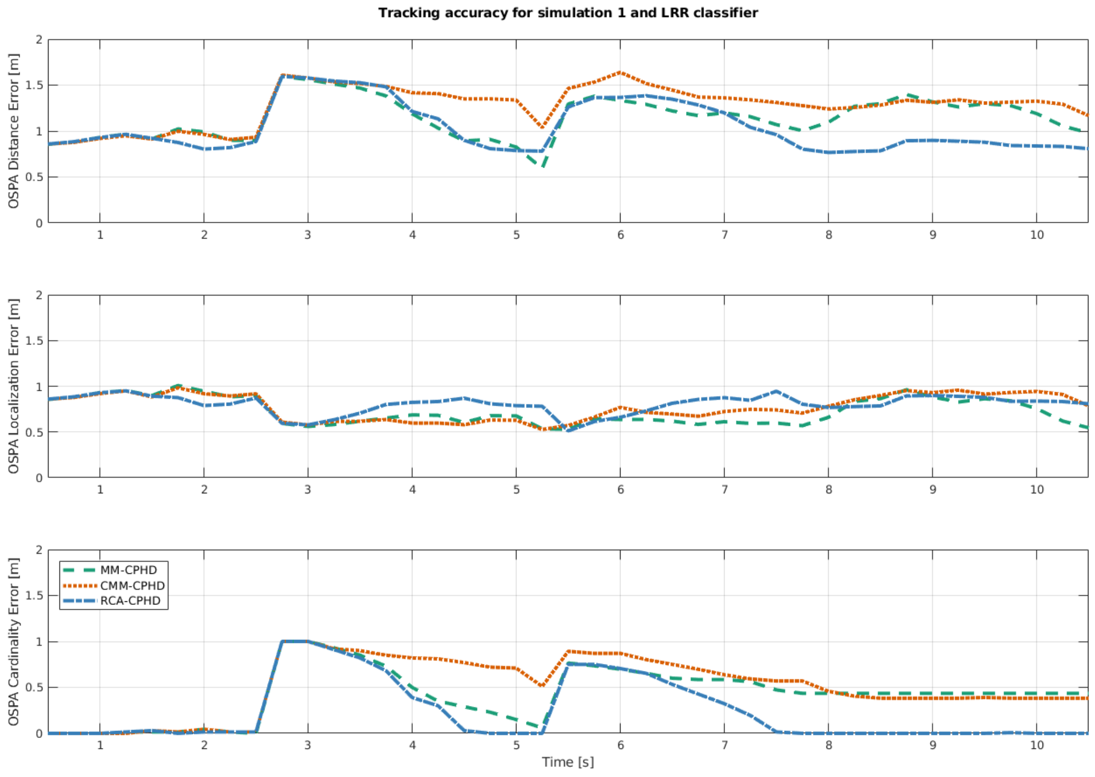


Figure 5.5: Tracking accuracy for 100 Monte Carlo runs for simulation 1 with measurements from a radar and LRR classifier

Classification accuracy The classification accuracy is measured by the averaging the classification MSE and absolute cardinality error over 100 Monte Carlo runs for both the LRR classifier and SSD classifier. The result for the SSD classifier and LRR classifier are respectively shown in figure 5.6 and 5.7. The top graph shows the MSE of the classification and the bottom graph the absolute cardinality error.

The occlusion moments are, just as in the tracking accuracy, clearly visible in the classification results. Although the first occlusion at 2.5 seconds only consequences a cardinality error, there is clearly a classification error at the other moments for the MM-CPHD and CMM-CPHD with peaks at the highest class uncertainty occlusions at 4.5 seconds and 8.5 seconds for the LRR classifier. Unlike the MM-CPHD and CMM-CPHD, the RCA-CPHD has continuously no classification error at all for the SSD and only two small errors with respect to the MM-CPHD and CMM-CPHD at the highest class uncertainty moments for the LRR.

Another peak in the classification error is seen between 9.2 and 10.5 seconds. During that period of time the pedestrian's track is regularly mixed up by the cyclist's track during the crossing. This is visible by a higher classification error for the MM-CPHD and CMM-CPHD, while the RCA-CPHD keeps a nearly zero classification error.

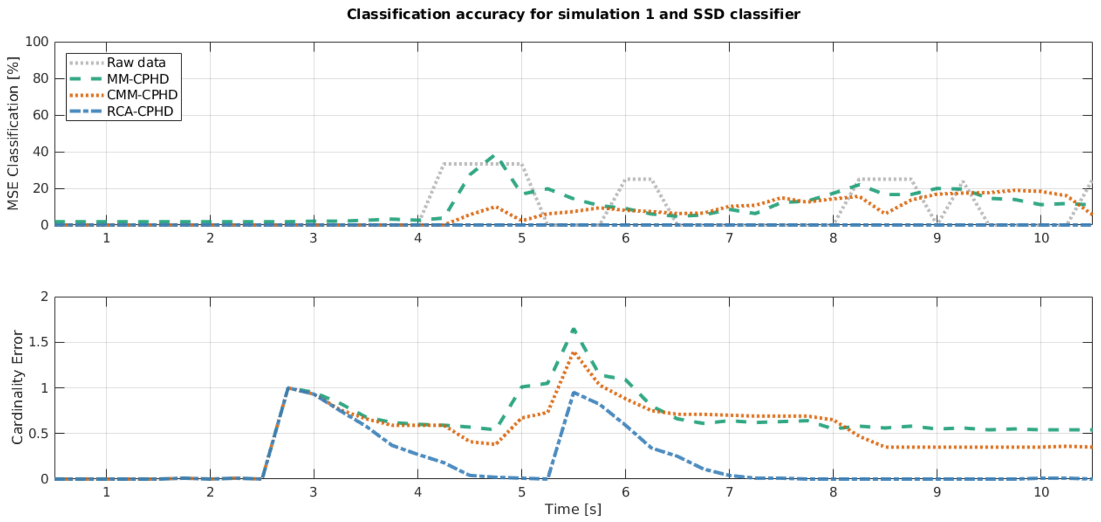


Figure 5.6: Classification accuracy for 100 Monte Carlo runs for simulation 1 with measurements from a radar and SSD classifier

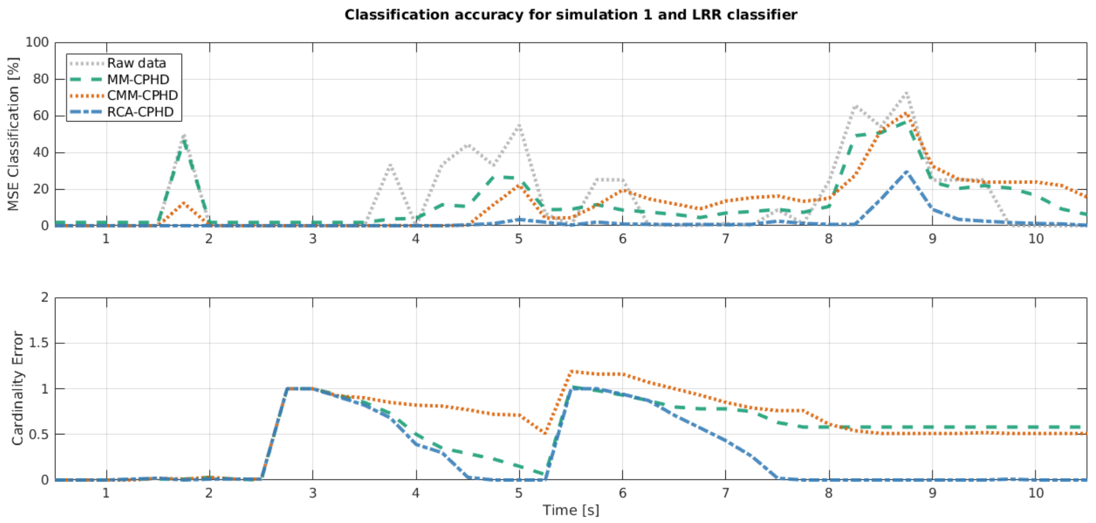


Figure 5.7: Classification accuracy for 100 Monte Carlo runs for simulation 1 with measurements from a radar and LRR classifier

The classification error gives an insight in the average classification accuracy of all the tracks together. The individual classification result per track is not visible in the error and are therefore visualized in figure 5.8 for the SSD classifier averaged over 100 Monte Carlo runs. The individual classification results for the LRR classifier are visualized in figure A.1 in Appendix A. Horizontally are the individual tracks visualized and in vertical direction the result per filter.

In the individual track classifications is clearly visible that the ground truth and the classifications of the RCA-CPHD are nearly the same. The MM-CPHD and CMM-CPHD do, however, have incorrect classifications in particular in the pedestrian's track 2 and the first cyclist's track 3. During the occlusion moments at 4.5 and 8.5 seconds both classifications are mixed up, resulting in the classification error.

The mixing up of the pedestrian's track with the cyclist's track is also clearly visible in the pedestrian's track 2 between 9.2 and 10.5 seconds.

Summary of the results

In this simulation is seen that the MM-CPHD and CMM-CPHD are less able to deal with the classification uncertainty caused by successive occlusions and mixing up of tracks during crossings. At these moments the tracking accuracy and classification accuracy of the MM-CPHD and CMM-CPHD decrease significantly with respect to the RCA-CPHD. This can also be seen when the OSPA distance and MSE are averaged over time as shown in table 5.2. Where the MM-CPHD and CMM-CPHD have almost the same average OSPA distance and average MSE, is the average OSPA distance and average MSE of the RCA-CPHD significantly lower.

	Raw Data	MM-CPHD	CMM-CPHD	**RCA-CPHD**
Total Average OSPA Distance LRR [m]	-	1.12	1.25	**1.01**
Total Average OSPA Distance SSD [m]	-	1.16	1.11	**0.84**
Total Average MSE LRR [%]	14.51	12.43	11.95	**1.95**
Total Average MSE SSD [%]	7.31	11.15	8.92	**0.00**

Table 5.2: The total average tracking and classification accuracy for simulation 1. The lowest error per performance measure is highlighted in bold

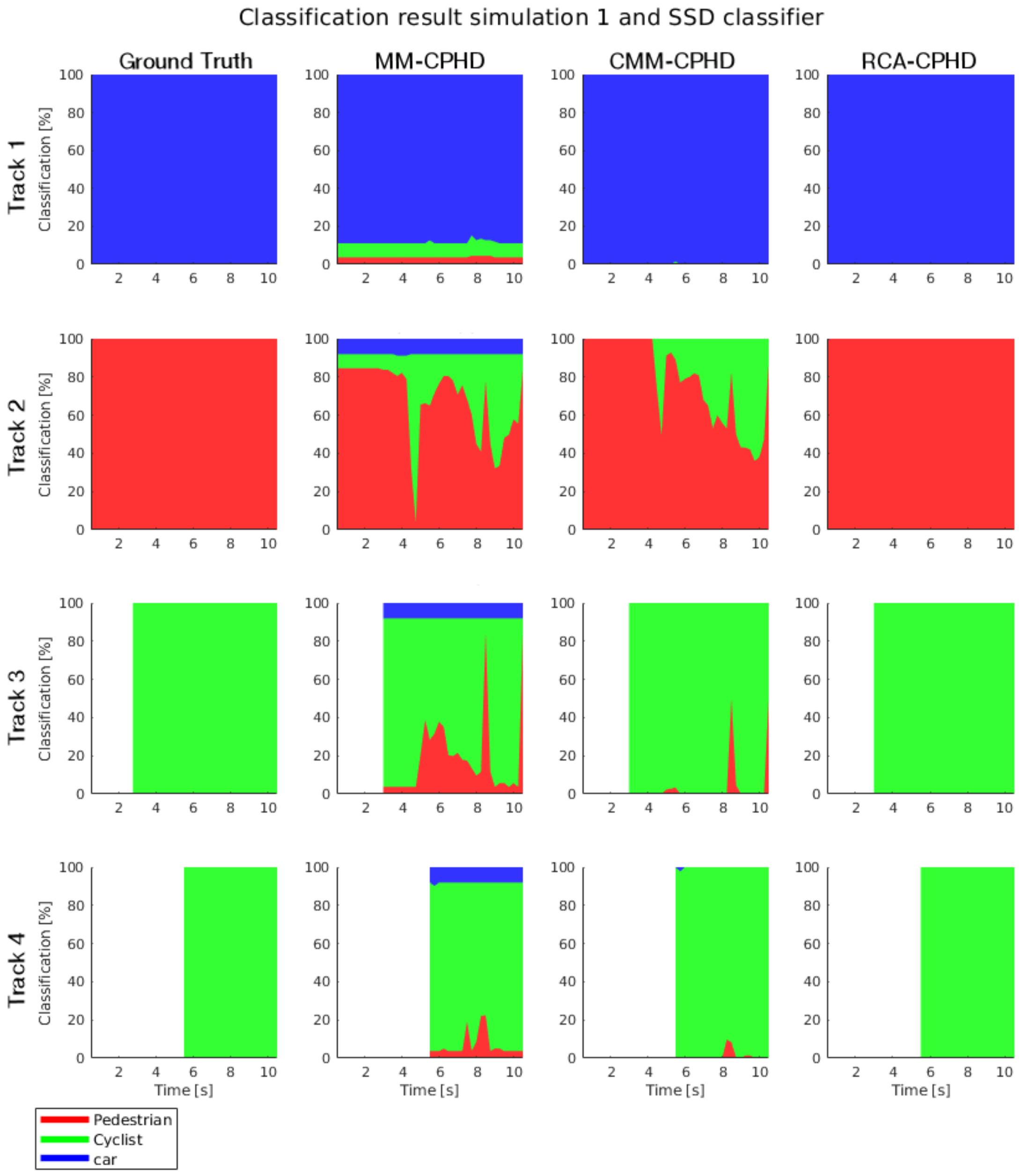


Figure 5.8: Visualization of the classification result for the individual tracks per filter.. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

5.2.2 Simulation 2

The second simulated scenario is a scenario with higher amount of occlusions and higher manoeuvrability of the objects. The tracking and classification is more challenging in this scenario than in section 5.2.1, making the robustness of the MM-CPHD, CMM-CPHD and RCA-CPHD visible.

The scenario has a duration of 16.5 seconds and has the weather conditions are a sunny day with no rain or dust. In total 3 objects, a pedestrian, a cyclist and a car, are appearing during the simulation in a range of 7 meters to 68 meters, with a maximum of 3 objects at the same time. During the scenario the ego-vehicle is parked in front of an intersection, while a pedestrian, cycling and car manoeuvrer over the intersection and cross each other several times. An overview of the course of this scenario is shown in figure 5.9.

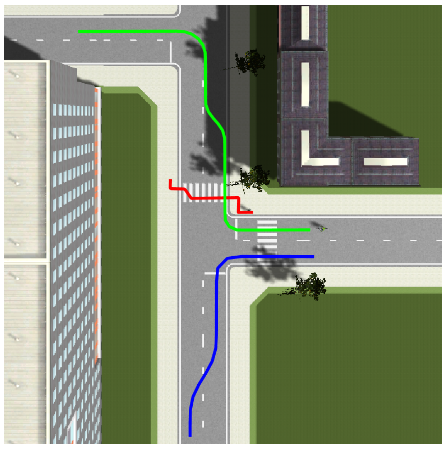
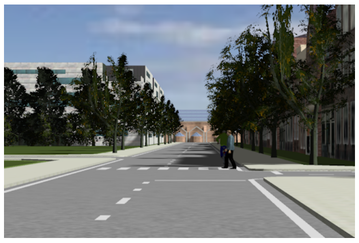


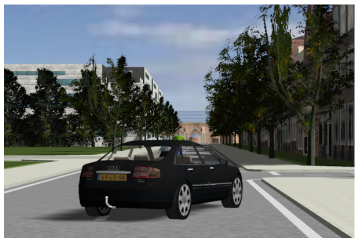
Figure 5.9: Overview of the course of the second simulated scenario. This scenario has a higher amount of occlusions and higher manoeuvrability of the objects

Course of scenario The simulation starts with a pedestrian walking next to a road and manoeuvrers towards a pedestrian crossover. After 1.5 seconds a cyclist enters the FOV, which subsequently occludes and crosses the cyclist from about 3 to 4 seconds as visualized in figure 5.10a. Subsequently from 5.5 to 8 seconds the pedestrian is occluding the cyclist as shown in figure 5.10a and 5.10b. Meanwhile at 8 seconds a car arrives in the FOV and occludes both the pedestrian and cyclist from 9 to 10 seconds. After 10 seconds, the pedestrian, cyclist and car continue their trajectory without occluding or crossing each other.

(a) Cyclist is occluding pedestrian at 3-4 sec



(b) Pedestrian is occluding cyclist at 5.5-8 sec



(c) Car is occluding both pedestrian and cyclist at 8-10 sec

Figure 5.10: Objects are crossing each other multiple times while manoeuvring.

Analysis

The performance is evaluated by two different measures, namely the tracking accuracy and the classification accuracy explained in more depth in section 5.1. First the tracking accuracy results are analysed for both classifiers and subsequently the classification accuracy.

Tracking accuracy The tracking accuracy results are shown in figure 5.11 and figure 5.12 for respectively the SSD and LRR classifier. The pedestrian is initialized for all three filters in the first second without a significant difference in error. The arrival of the cyclist at 1.5 seconds results in a cardinality error in all three filters. The RCA-CPHD initializes the cyclist on average faster than the MM-CPHD and CMM-CPHD, resulting in a lower OSPA cardinality and OSPA distance error from about 2 seconds

The successive occlusions and crossings between 3 and 10 seconds result in a classification uncertainty. During this classification uncertainty, the MM-CPHD and CMM-CPHD have an higher OSPA distance with respect to the RCA-CPHD. After 10 seconds, the pedestrian, cyclist and car continue their trajectory without occluding or crossing each other. During that time period, there is no classification uncertainty and the MM-CPHD, CMM-CPHD and RCA-CPHD have a nearly similar OSPA distance.

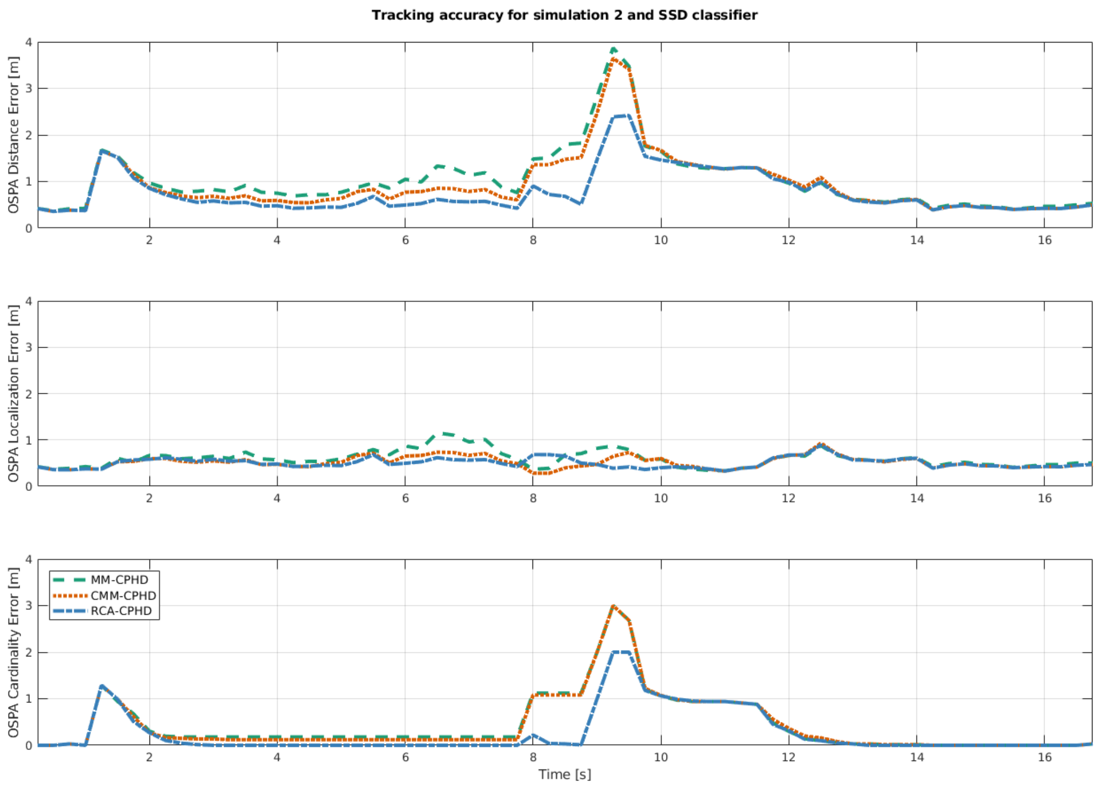


Figure 5.11: Tracking accuracy for 100 Monte Carlo runs for simulation 2 with measurements from a radar and SSD classifier

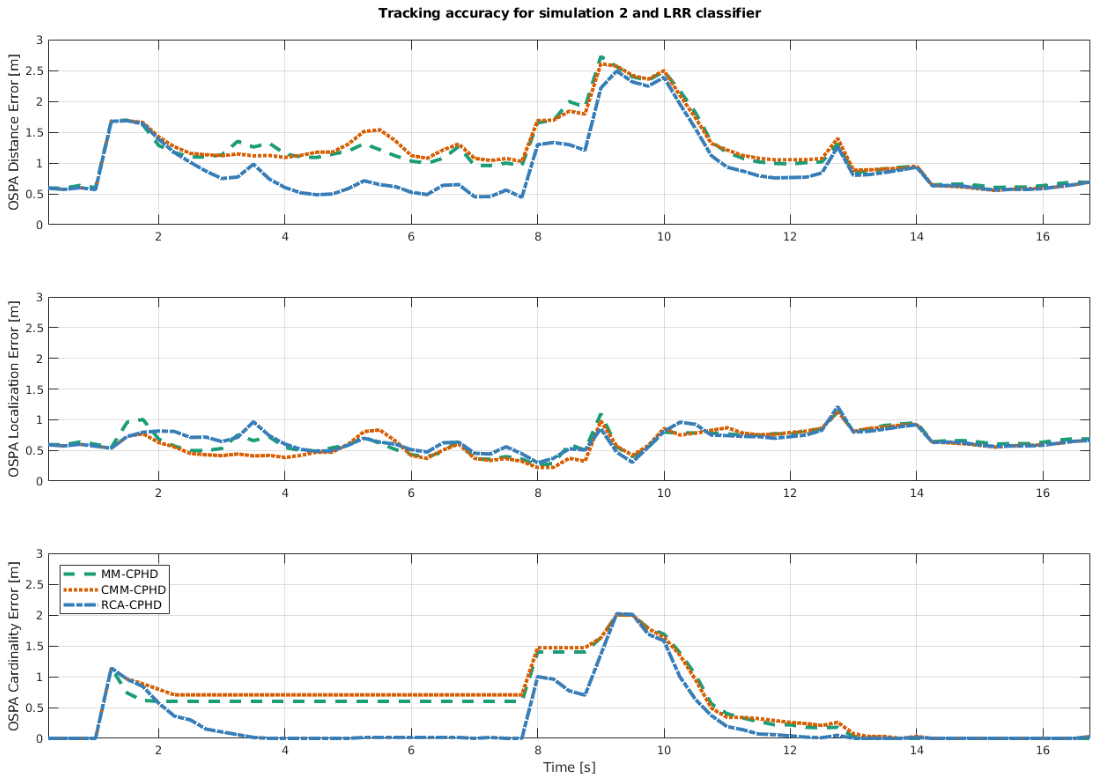


Figure 5.12: Tracking accuracy for 100 Monte Carlo runs for simulation 1 with measurements from a radar and LRR classifier

Classification accuracy The class uncertainty due to occlusion is even more visible in the classification accuracy results shown in figure 5.13 for the SSD classifier and 5.13 for the LRR classifier. At the three occlusion moments at about 3.5 seconds, 5.5 to 8 seconds and 9 seconds, the classification error shows a peak in the classification MSE for the MM-CPHD and CMM-CPHD. For the RCA-CPHD is, however, no classification error observable at all for the SSD and a small error for the LRR at the occlusion of both object by the car at 9 seconds.

Besides there is also a peak observable in the MSE with the SSD classifier at 2.5 seconds and a peak at 12.5 seconds. At these moments the SSD detected also a pedestrian in the cyclist. The RCA-CPHD recognizes the difference in classification and ignores this mis-detection during the update of the cyclist. The MM-CPHD and CMM-CPHD, however, use the mis-detection in their update resulting in a classification error.

When looking at the classification result of the individual tracks for the SSD classifier, the class uncertain moments are clearly visible in the tracks as well. The multiple occlusions between 3.3 and 10 seconds result in an incorrect classification for the MM-CPHD and CMM-CPHD, while the RCA-CPHD keeps the correct classification for the tracks.

The result of an occlusion for a longer period of time is observable between 9 and 10 seconds, for example in the pedestrian track 1. The RCA-CPHD retains the correct classification for the first few time steps of the occlusion, but after not receiving a measurement corresponding to that object, the track is terminated at 9 seconds. At 10 seconds a new measurement arrives and the track is continued again. The MM-CPHD and CMM-CPHD, however, are updating the track a little longer with the incorrect measurement visible by the incorrect classification of the pedestrian track 1 between 9-10 seconds. The incorrect measurement pollutes the track estimate, resulting in a higher OSPA distance and potential wandering of the track.

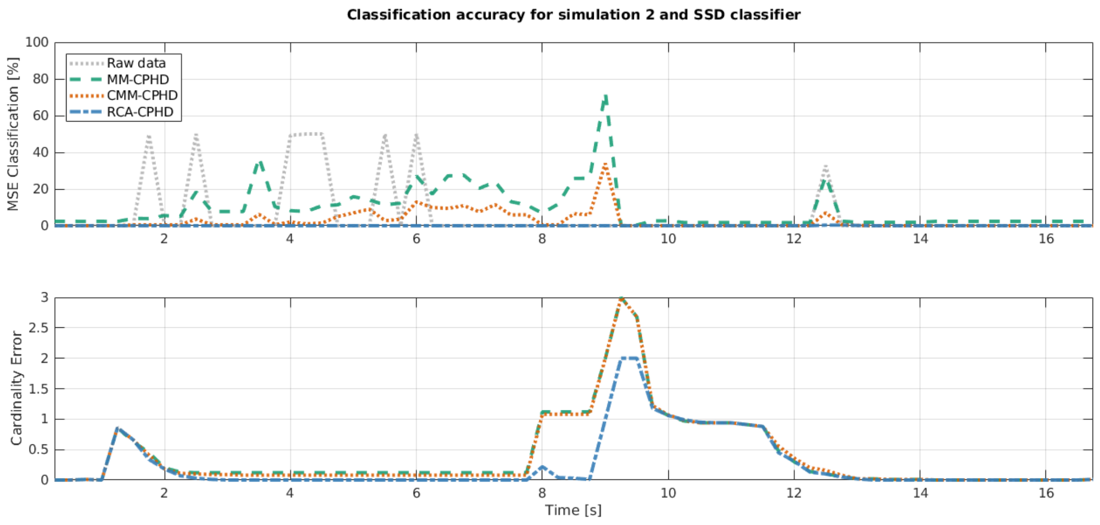


Figure 5.13: Classification accuracy for 100 Monte Carlo runs for a simulated scenario with high occlusion and manoeuvring objects using measurements from a radar and SSD classifier

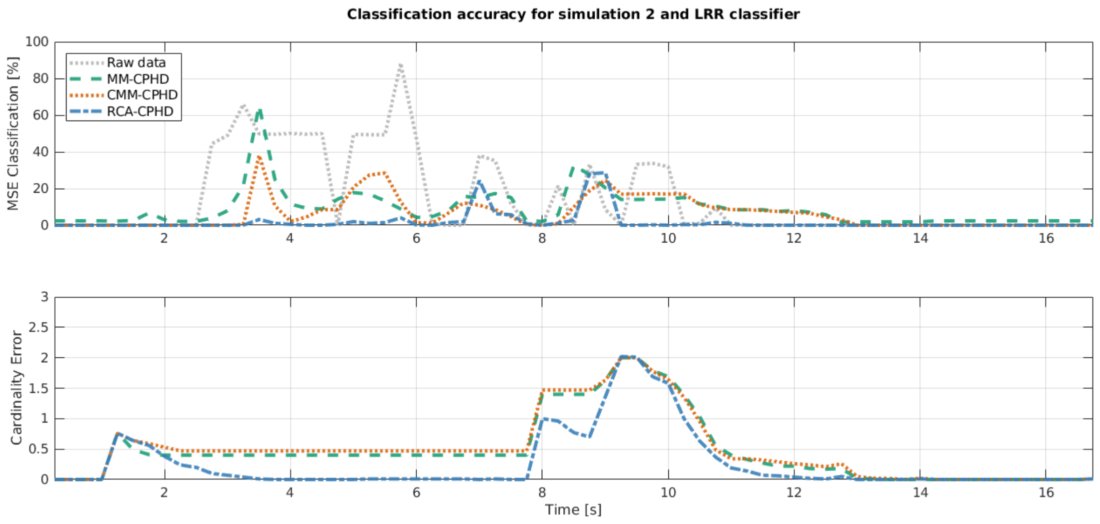


Figure 5.14: Classification accuracy for 100 Monte Carlo runs for a simulated scenario with high occlusion and manoeuvring objects using measurements from a radar and LRR classifier

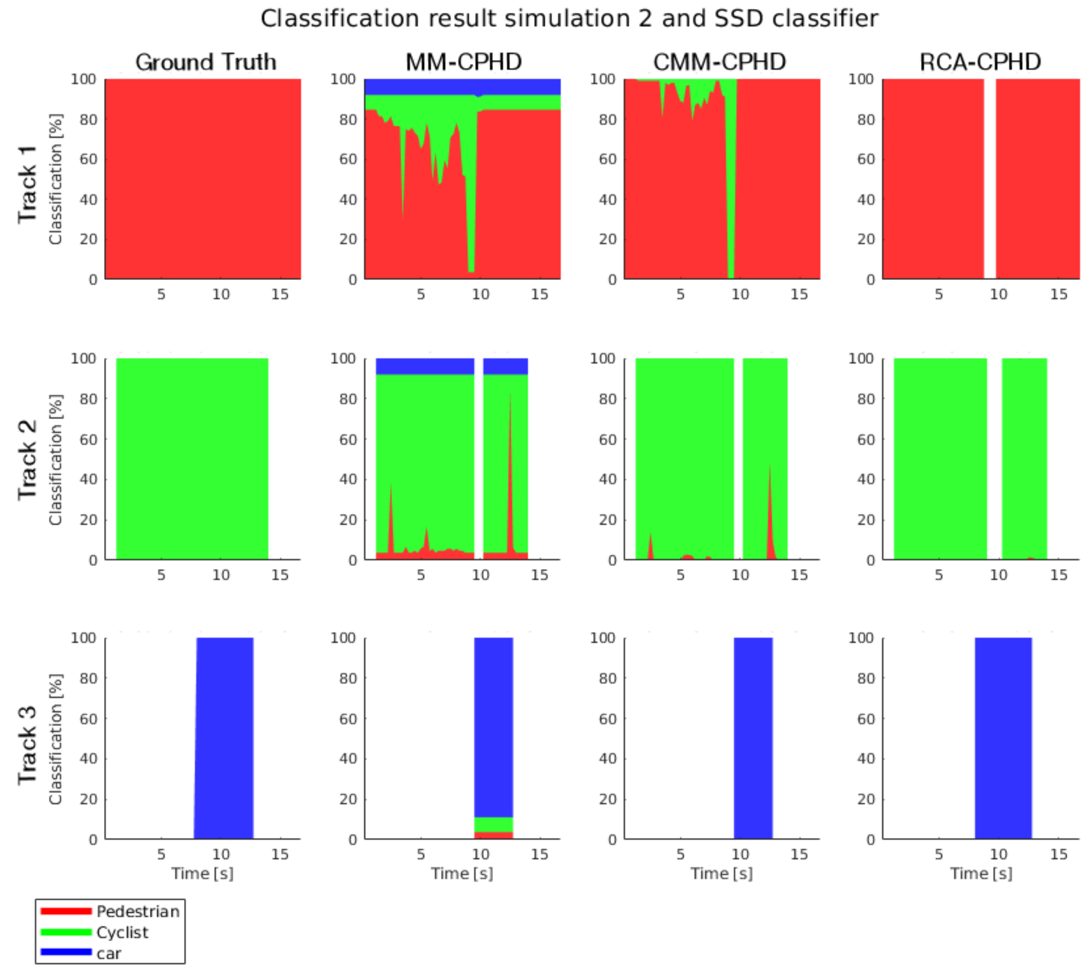


Figure 5.15: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

Summary of the results

The RCA-CPHD has a higher tracking- and classification accuracy during the high classification uncertainty moments with respect to the MM-CPHD and CMM-CPHD in the examined simulation. This is also seen in the average OSPA distances and average MSE over the entire simulation. The RCA-CPHD has a significantly lower OSPA distance and lower MSE for both classifiers with respect to the CMM-CPHD and MM-CPHD.

	Raw Data	MM-CPHD	CMM-CPHD	**RCA-CPHD**
Total Average OSPA Distance LRR [m]	-	1.19	1.21	**0.94**
Total Average OSPA Distance SSD [m]	-	1.02	0.93	**0.77**
Total Average MSE LRR [%]	14.20	9.07	6.28	**1.77**
Total Average MSE SSD [%]	5.70	9.08	2.62	**0.01**

Table 5.3: The total average tracking and classification accuracy for simulation 2. In bold the lowest error is highlighted per performance measure.

5.2.3 Verification Conclusion

The results of both simulations show that for every individual simulation, the RCA-CPHD has a significant higher tracking accuracy compared to the MM-CPHD and CMM-CPHD during classification uncertain moments, like occlusions, crossings and misclassifications:

	Simulation 1	Simulation 2
RCA-CPHD	**1.01 m**	**0.94 m**
CMM-CPHD	1.25 m	1.21 m
MM-CPHD	1.12 m	1.19 m

LRR classifier

	Simulation 1	Simulation 2
RCA-CPHD	**0.84 m**	**0.77 m**
CMM-CPHD	1.11 m	0.93 m
MM-CPHD	1.16 m	1.02 m

SSD classifier

Table 5.4: Summary of OSPA error for all simulations

In addition, the results also show that for every individual simulation, the RCA-CPHD has a significant higher classification accuracy during the classification uncertain moments:

	Simulation 1	Simulation 2
RCA-CPHD	**1.96 %**	**1.77 %**
CMM-CPHD	11.96 %	6.28 %
MM-CPHD	12.43 %	9.07 %

LRR classifier

	Simulation 1	Simulation 2
RCA-CPHD	**0.00 %**	**0.01 %**
CMM-CPHD	6.60 %	2.62 %
MM-CPHD	9.56 %	9.08 %

SSD classifier

Table 5.5: Summary of classification MSE for all simulations

To validate whether the simulations are also representative for real world scenarios, the tracking accuracy and classification accuracy are also evaluated for two different scenarios with real world data in section 5.3.

5.3 Validation using Real Data

In the validation step, the performance of the RCA-CPHD is, just like the verification, compared to the performance of the MM-CPHD and CMM-CPHD. The performance comparison in the verification step was done using simulated data. In the validation step is examined whether the results of the verification step are also representative for real world data.

Experiment setup

The validation is done on real world data from six different traffic situations. The data used in experiment 1 and 2 is recorded with a stereo camera mounted on the DAVI vehicle. Experiment 3,4,5 and 6 are selected traffic situations from the KITTI database, a database containing camera images, laser scans, high precision GPS measurements and IMU accelerations recorded while driving on German roads [22].

Ground truth In contrast to the simulations, there is no ground truth on the real data. A commonly used method for defining the ground truth for object detection and tracking is by human annotation of the camera images [26]. The KITTI database provides annotation labels for a number of recordings, however for this specific recordings there were no labels available. The annotation is therefore for all experiments done using the annotation tool VATIC developed by Vondrick et al. [79]. A disparity map is created using both stereo images and based on the intrinsic and extrinsic parameters of the stereo camera, the annotations are converted to ego vehicle coordinate system. Subsequently the ground truth is obtained by Kalman Smoothing to correct annotation irregularities and measurement noise.

Classifier The image frames from the left camera are processed by both the LRR and SSD classifier. As explain in section 2.4, both classifier differ in the way of dealing with classification uncertainty, making it interesting for taking into account both classifiers. Therefore the experiment is run twice, once for the LRR and once for the SSD classifier. As explained in section 2.4 the output of the classifiers are adjusted to output the probability of all the observed classes. Besides the LRR classifier outputs the class-probabilities per pixel instead of bounding box. Therefore the bounding boxes are annotated and the corresponding class-probabilities are determined by the median of all the pixel class-probabilities in the bounding box.

Filter parameter setup

The RCA-CPHD, MM-CPHD and CMM-CPHD are run with similar parameters and an overview of the used parameters given in table 5.6.

Detection probability The detection probability is dependent on the sensor modality and the amount of occlusions in the recording. Since the SSD classifier is a detector, the detection results for persons, bikes and cars for a trained model of VOC2007,VOC2012 and COCO dataset [36] are used for determining detection probability P_D. The conversion of classification per pixel to bounding boxes for the LRR classifier is however done by annotation and the detection probability is therefore dependent on the annotation process. The average number of bounding boxes with respect to the actual number of objects in the ground truth is therefore used as the detection probability for the LRR classifier.

Clutter The conversion of classification per pixel to bounding boxes by annotation in the LRR classifier results in virtually no clutter. Therefore the amount of clutter is set to 0.1 clutter objects per scan. In the SSD classifier, however, clutter occurs more often. For example when a cyclist is seen as both a bike and a pedestrian. Therefore the clutter intensity for the SDD is higher with respect to the LRR and is set to 0.5 clutter objects per scan.

Parameters used in RCA-CPHD, MM-CPHD, CMM-CPHD		
Detection probability LRR classifier	$P_{D,LRR}$	89%
Detection probability SSD classifier	$P_{D,SSD}$	90%
Clutter intensity LRR	$\kappa_k(z)$	$0.1\frac{objects}{scan}$
Clutter intensity SSD	$\kappa_k(z)$	$0.5\frac{objects}{scan}$
Survival probability	P_s	98%
Merging threshold	τ_{merge}	1
Pruning threshold	τ_{prune}	$10^{-5}\ \frac{\text{number of objects}}{m^2}$
Capping threshold	τ_{cap}	400 GM components
Measurement noise classifier	σ_u	5 px
	σ_v	5 px
	$\sigma_{d,camera}$	0.25 px
Parameters used in RCA-CPHD only		
Class threshold	τ_c	25°
Class discount	$\alpha_{discount}$	0.95

Table 5.6: Summary of filter parameters used in all real world data experiments

5.3.1 Experiment 1

The first traffic situation is a representative urban traffic situation recorded at an intersection near the TU Delft campus. An overview of the intersection, from the viewpoint of the DAVI vehicle, is shown in figure 5.16.

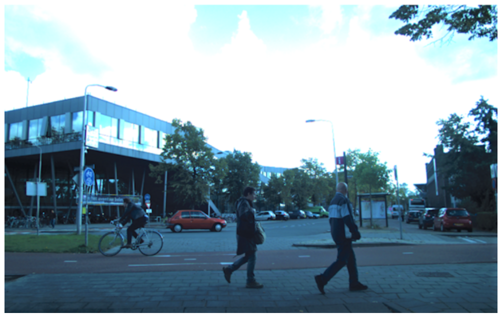


Figure 5.16: Experiment 1 is a representative urban traffic situation recorded near the TU Delft campus

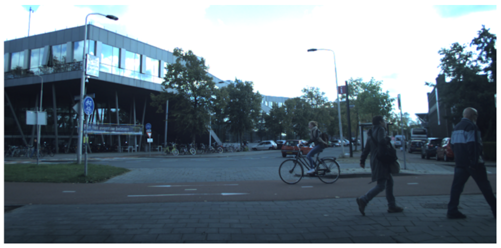
The experiment data is recorded with a stereo camera mounted on the DAVI vehicle. The recording has a total length of 13 seconds and is recorded with a frequency of 5 hz. The weather condition were slightly cloudy with no rain or dust and the total number of objects, consisting of pedestrians, cyclists and cars, is 13. The maximum number of objects at the same time is 6 and the objects are ranging from 5 to 31 meters.

Course of recording The recording starts with two pedestrians walking on a footpath, a cyclist on a bike path behind it, a car making a turn on the road on the background and a parked car on the right of the FOV as shown in figure 5.17a. At 0.8 seconds a cyclist appears in the FOV and is subsequently occluded by the two pedestrians at 1.0 seconds and 1.4 seconds as shown in figure 5.17b. After being occluded, the cyclist is visible again and then occluding a car at 1.6 seconds as shown in figure 5.17c. At the same time between 1.2 seconds and 2.8 seconds the two pedestrians are occluding a parked car after each other as also shown in figure 5.17c.

After 5.8 seconds, seven cyclists pass the FOV successively and disappear again between 8.4 and 12.7 seconds as shown in figure 5.17d. During the passing, the cyclists occlude consecutively the parked park.

(a) situation at start of recording

(b) pedestrian is occluding the cyclist at 1.1 sec


(c) Pedestrians are occluding the parked car in 1.3 - 2.8 sec

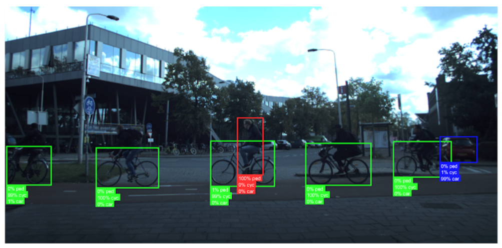

(d) Multiple cyclists occlude successively the parked car in 6 - 9 seconds

Figure 5.17: Overview course and events of experiment 1

Analysis

The performance is evaluated by two different measures, namely the tracking accuracy and the classification accuracy explained in more depth in section 5.1. First the tracking accuracy results are analysed for both classifiers and subsequently the classification accuracy.

Tracking accuracy The tracking accuracy results are shown in figure 5.18 and figure 5.19 for respectively the SSD and LRR classifier. Between 0.8 and 3 seconds there is a classification uncertainty due to multiple occlusions. During these occlusions the RCA-CPHD is more able to distinguish the different objects from each other. This results in a lower cardinality error and therefore in a lower OSPA distance in the first 3 seconds for both types of classifiers compared to the MM-CPHD and CMM-CPHD. Between 3 and 5.4 seconds all the moving objects are leaving the FOV without occlusions and the parked car is noticed by all three filters. During this period there is virtually no classification uncertainty, which results in an equivalent OSPA distance in this period of time for the LRR classifier. The occlusion, however, have caused a lower cardinality estimate for the MM-CPHD when using the SSD.

After 5.8 seconds, the seven cyclists pass the FOV successively and disappear again between 8.4 and 12.7 seconds as shown in figure 5.17d. During the passing, the cyclists occlude consecutively the parked park, resulting in a classification uncertainty. During the passing, the OSPA distance is lower for the RCA-CPHD with respect to the MM-CPHD and CMM-CPHD. The RCA-CPHD initiates the tracks faster and is more able to distinguish the different tracks. In addition, the SSD classifier causes multiple misclassifications of the cyclists, resulting in false alarms of pedestrians as shown in figure 5.17d. The misclassifications are in the MM-CPHD and CMM-CPHD used in the update step, while they are filtered out in the gating step of RCA-CPHD. The misclassifications disturb the association process and therefore cause a larger cardinality error.

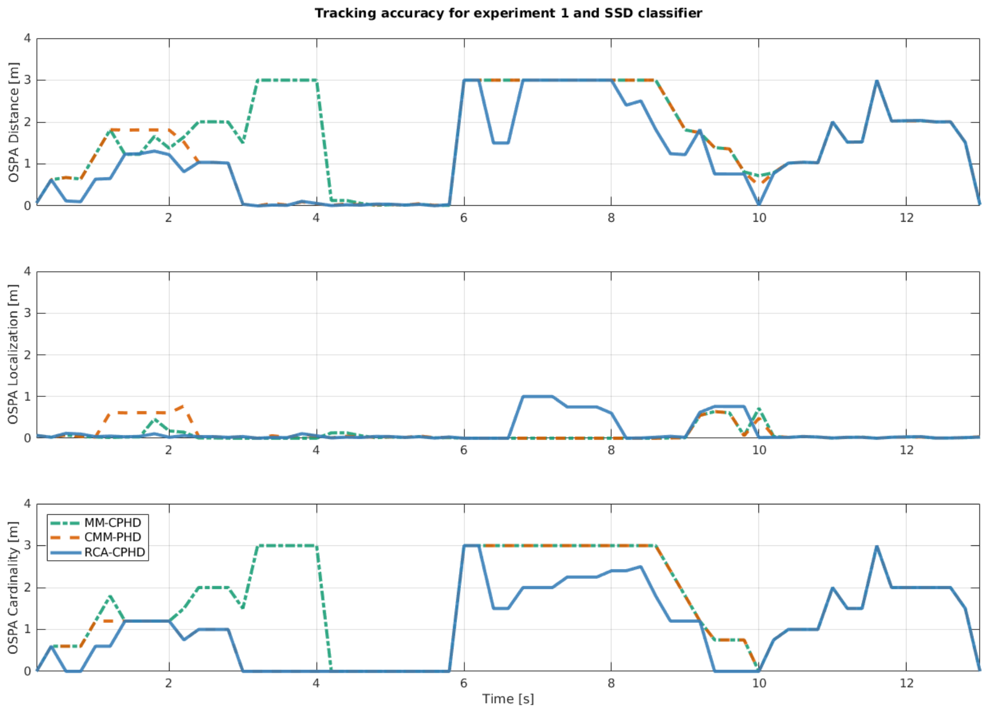


Figure 5.18: Tracking accuracy for experiment 1 using SSD classifier

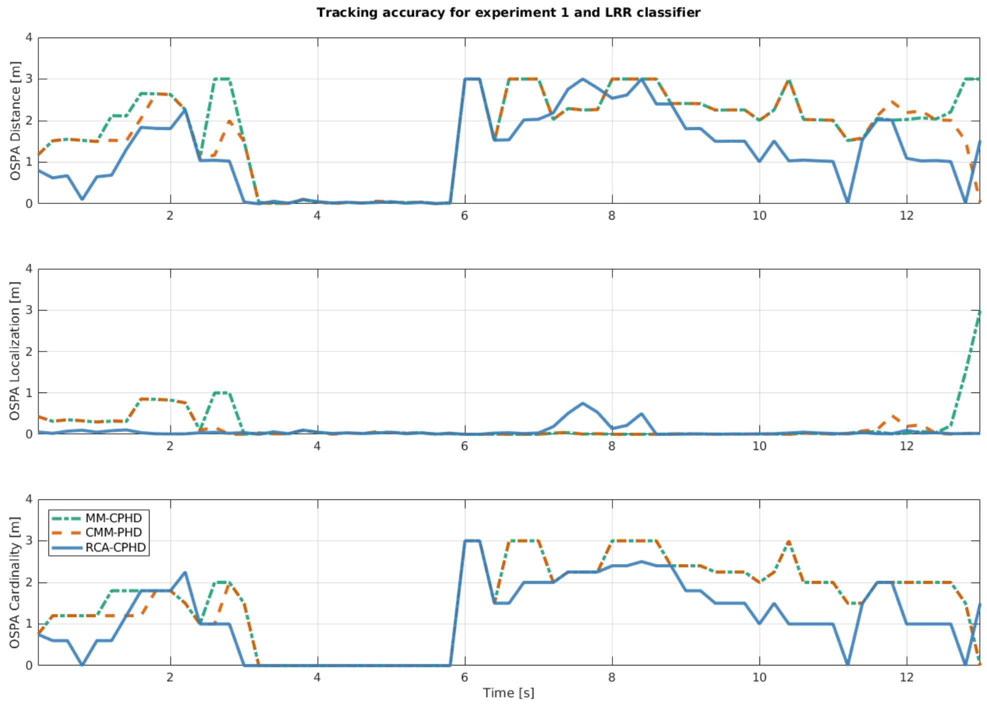


Figure 5.19: Tracking accuracy for experiment 1 using LRR classifier

Classification accuracy The classification accuracy results are shown in figure 5.20 and 5.21 for respectively the SSD and LRR classifier. The class uncertainty due to the pedestrian occluding the cyclist is clearly visible by the MSE peaks at 1.1 seconds for the LRR classifier. The occlusions of the parked car by the cyclist and pedestrians in the first second are not visible in the MSE, since the cyclist was not initiated by the MM-CPHD and CMM-CPHD in the first second. Since the tracks are not initiated, they result in a cardinality error as seen in the first seconds. The RCA-CPHD, however, did initiate all objects already in the first second, resulting in a lower cardinality error and OSPA distance. This difference is also visible in the individual track classifications visualized in figure 5.22 and A.3.

Between 6-9 the parked car is continuously occluded by passing cyclists. During this time period the track of the parked car is not noticed by all the filters and no classification error results from these occlusions. After 9 seconds, the parked car is visible for a short period of time and initiated again. The subsequent occlusions from cyclists passing the parked car result in an increase in the MSE error after 9 seconds. Besides the RCA-CPHD is able to identify more of the cyclists than the MM-CPHD and CMM-CPHD during this period of time, resulting in a lower cardinality error.

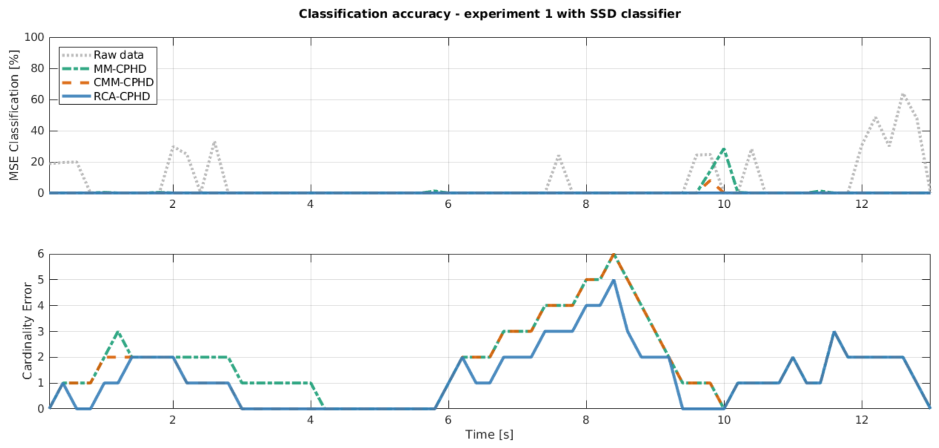


Figure 5.20: Classification accuracy for experiment 1 using SSD classifier

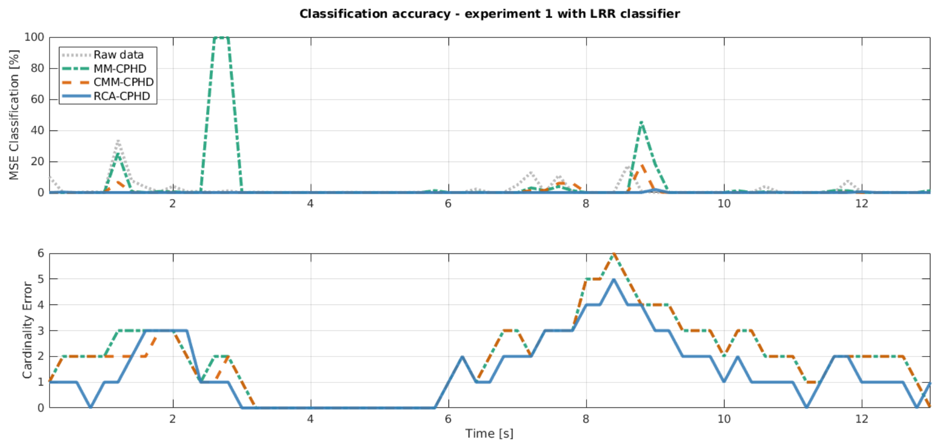


Figure 5.21: Classification accuracy for experiment 1 using LRR classifier

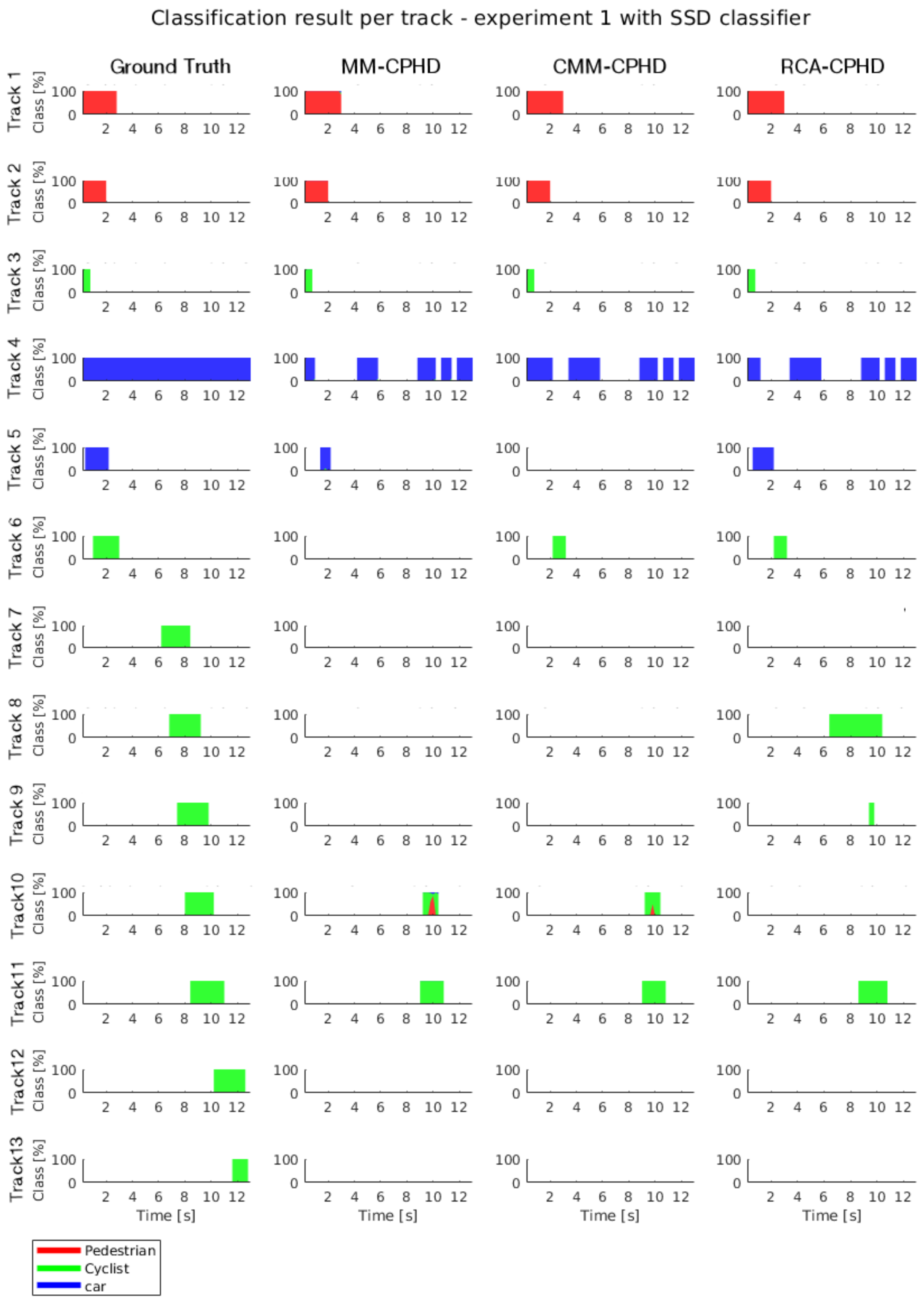


Figure 5.22: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

Summary of the results

The results of experiment 1 show that the RCA-CPHD has a higher tracking- and classification accuracy during classification uncertainty moments, e.g. occlusions and crossings, with respect to the MM-CPHD and CMM-CPHD. This is also seen in the average OSPA distances and average MSE over the entire recording. The RCA-CPHD has a significantly lower OSPA distance and lower MSE for both classifiers with respect to the CMM-CPHD and MM-CPHD.

	Raw data	MM-CPHD	CMM-CPHD	**RCA-CPHD**
Total Average OSPA Distance LRR [m]	-	1.80	1.67	**1.22**
Total Average OSPA Distance SSD [m]	-	1.71	1.43	**1.21**
Total Average MSE LRR [%]	1.94	4.72	0.62	**0.40**
Total Average MSE SSD [%]	7.21	0.72	0.12	**0.00**

Table 5.7: The total average tracking and classification accuracy for simulation 1. In bold the lowest error is highlighted per performance measure.

5.3.2 Experiment 2

The second traffic situation is recorded at another intersection near the TU Delft campus and the recording is characterized by a higher classification of uncertainty. An overview of the intersection, from the viewpoint of the DAVI vehicle, is shown in figure 5.23.

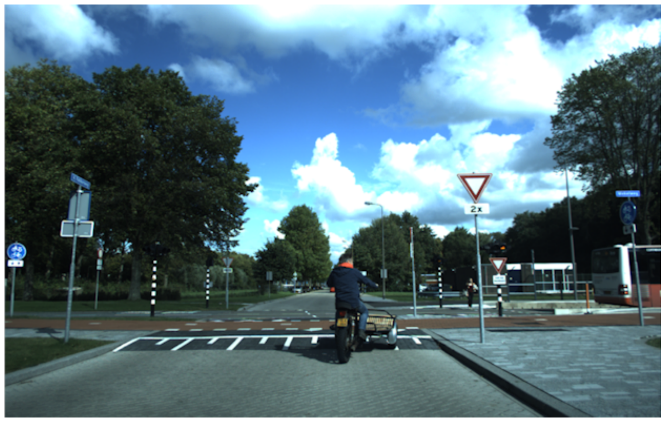


Figure 5.23: The second traffic situation is an urban traffic situation with high classification uncertainty recorded near the TU Delft campus

The experiment data is, just like experiment 1, recorded with a stereo camera mounted on the DAVI vehicle. The recording has a total length of 22 seconds and is recorded with a frequency of 5 hz. The weather condition were slightly cloudy with no rain or dust and the total number of objects, consisting of pedestrians, cyclists, motorized cargo bike and cars, is 6. The maximum number of objects at the same time is 3 and the objects are ranging from 4 to 65 meters.

Course of recording The recording starts with a pedestrian walking on a bus platform and after 1.2 seconds a car appears in the FOV as shown in figure 5.24a. Subsequently at 5.4 seconds a cyclist appears in the FOV, which occludes the car at 6.8 seconds as shown in figure 5.24b.

At 11.2 a motorized cargo bike and bus appear, which occlude both the car and and pedestrian as shown in figure 5.24c. The cargo bike is mostly recognized as a cyclist, but has a class uncertainty of being a car. At 14.8 seconds a cyclist appears on the bike path and occludes first the cargo bike at 16.2 seconds and subsequently the pedestrian at 17.4 seconds as shown in figure 5.24d.

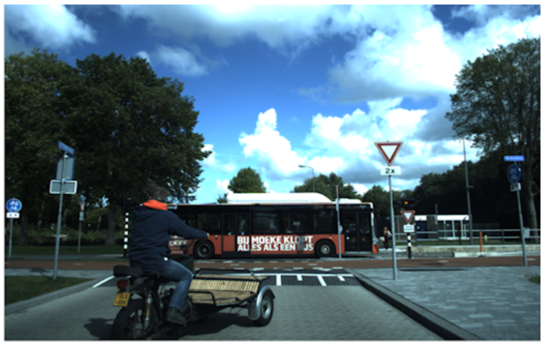
Analysis

The performance is evaluated by two different measures, namely the tracking accuracy and the classification accuracy explained in more depth in section 5.1. First the tracking accuracy results are analysed for both classifiers and subsequently the classification accuracy.

Tracking accuracy The tracking accuracy results are shown in figure 5.25 and figure 5.26 for respectively the SSD and LRR classifier. The pedestrian is difficult detect in the first 5.4 seconds, with as a consequence a high class uncertainty for the pedestrian for the LRR classifier and no detections for the SSD classifier. These missed detection in the SSD classifier results in an equal classification error and equal OSPA distance for all three filters. The high class uncertainty for the pedestrian with the LRR classifier, however, does result in a difference in cardinality between the RCA-CPHD with respect to the MM-CPHD and CMM-CPHD. The RCA-CPHD is able to initialize the car faster, resulting in a lower cardinality error between 2.5 and 65.4 seconds for both classifiers.

(a) car appearing after 1.2 seconds

(b) cyclist occludes the car at 6.8 seconds


(c) Cargo bike appears and bus appear and occlude both the car and pedestrian at 11.2 seconds

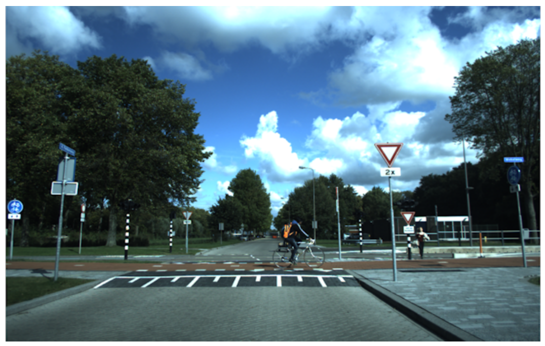

(d) Cyclist occludes first the cargo bike and subsequently the pedestrian

Figure 5.24: Overview of classification uncertain moments in urban traffic situation with high classification uncertainty

The appearance of the cyclist at 5.4 seconds increases the cardinality of all the filters. As a result the MM-CPHD and CMM-CPHD initialize the cars track as well and the difference in cardinality error is corrected at 6 seconds. Since the cyclist occludes the car for several time steps, the cars track dies out for one time step for the RCA-CPHD at 7 seconds, resulting in a cardinality error at that time step. The MM-CPHD and CMM-CPHD allow the possibility to update with the incorrect classified measurement, causing the track to continue exist, but with a classification error at the same time.

At 11.2 the motorized cargo bike and bus appear, which occlude both the car and and pedestrian as shown in figure 5.24c. The cargo bike is mostly recognized as a cyclist, but has a class uncertainty of being a car. At 14.8 seconds a cyclist appears on the bike path and occludes first the cargo bike at 16.2 seconds and subsequently the pedestrian at 17.4 seconds as shown in figure 5.24d. The cyclist appearing at 14.8 seconds is faster initialized by the RCA-CPHD, resulting in a lower OSPA cardinality error for the SSD classifier. The subsequent occlusion of this cyclist cause a cause a mix up between the tracks in the MM-CPHD and CMM-CPHD for the SSD classifier. This results in a larger OSPA distance with respect to the RCA-CPHD during that period of time. The classification uncertainty in the LRR during this time period is low, resulting in virtually no difference between the three filters for the LRR classifier.

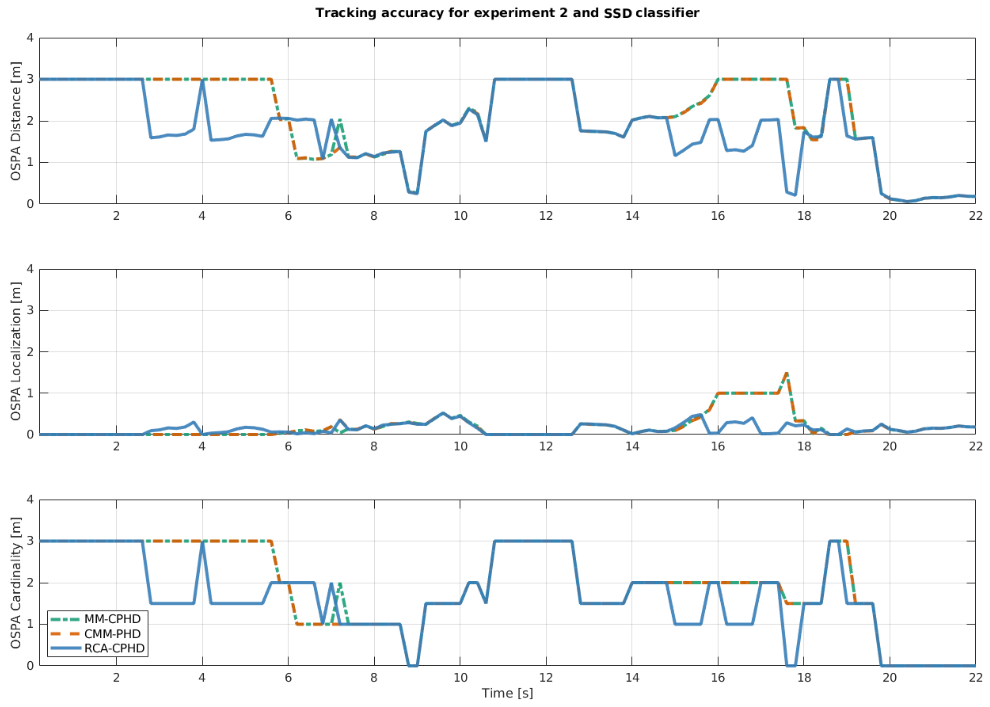


Figure 5.25: Tracking accuracy for experiment 2 using SSD classifier

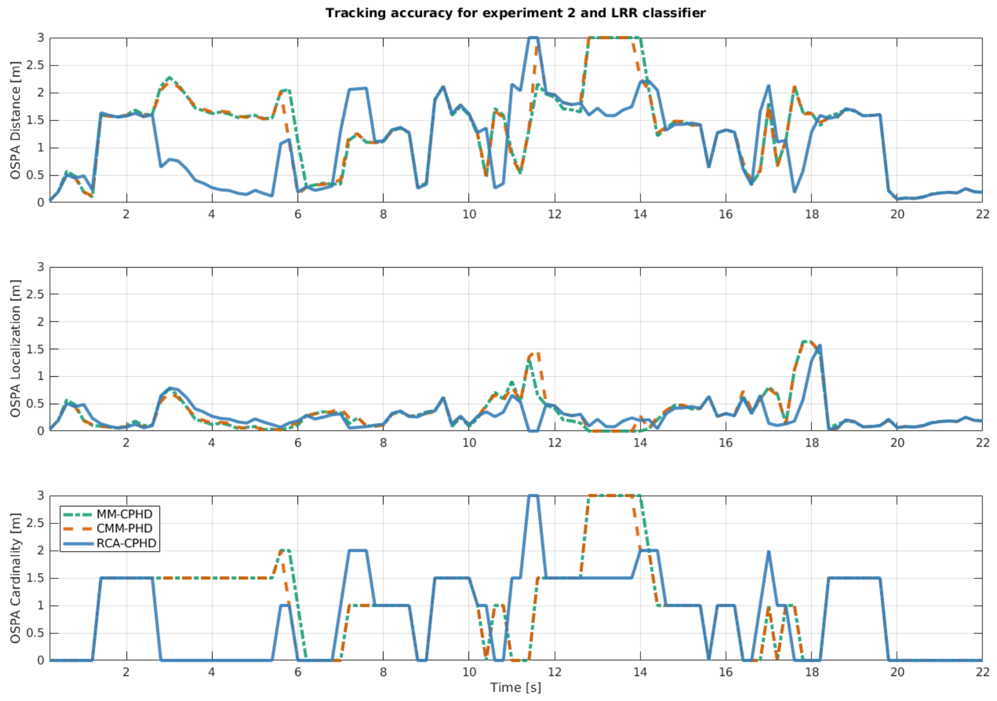


Figure 5.26: Tracking accuracy for experiment 2 using LRR classifier

Classification accuracy The classification accuracy for both the SSD and LRR classifier is shown in respectively figure 5.27 and 5.28. Since the pedestrian is difficult to detect in the first seconds, the class uncertainty is high in the first five seconds. This is visible at the peaks at 1, 2 and 4 seconds in the MSE of the LRR classifier. During that period of time, the pedestrian was missed by the SSD classifier, resulting in a cardinality error.

Between 5 and 8 seconds the SSD creates multiple false alarms by classifying the cyclist as both a cyclist and pedestrian. At 6 and 8 seconds the false alarms are influencing the track of the MM-CPHD and CMM-CPHD, resulting in a peak in the MSE at these time steps. This is also visible in the individual track classification of track 3 shown in figure 5.29. The individual track classification visualization for the LRR classifier is shown in figure A.8

Between 16 and 18 seconds the cyclist occludes the cargo bike and pedestrian, with as a consequence a mix up of the tracks in the MM-CPHD and CMM-CPHD and a high MSE in the SSD classifier. This is also clearly visible in the pedestrians track 2 shown in the classification visualizations of the tracks in figure 5.29.

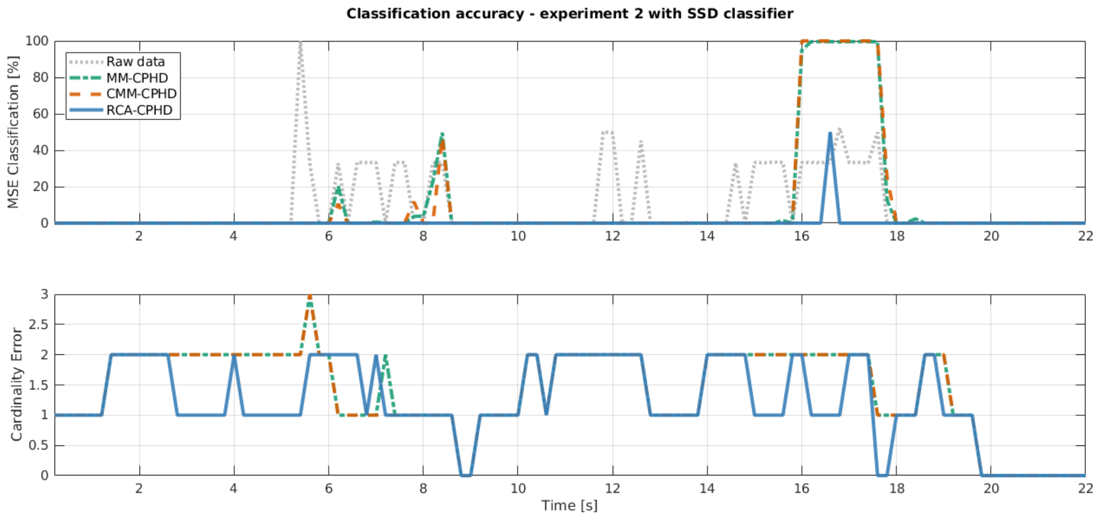


Figure 5.27: Classification accuracy for experiment 2 using SSD classifier

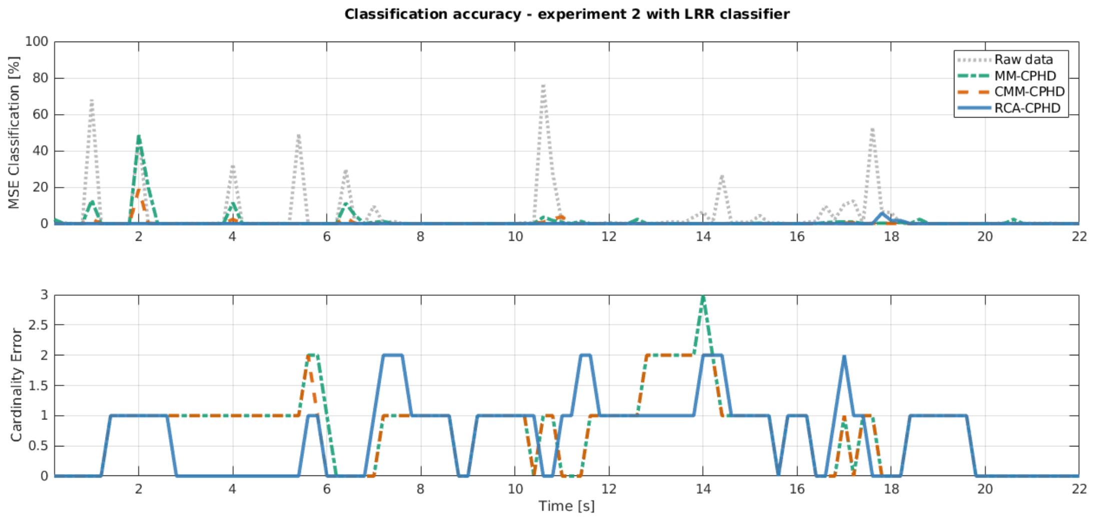


Figure 5.28: Classification accuracy for experiment 2 using LRR classifier

Summary of results

The results of experiment 2 show that the RCA-CPHD has a higher tracking- and classification accuracy during the high classification uncertainty moments with respect to the MM-CPHD and CMM-CPHD. This is also seen in the average OSPA distances and average MSE over the entire recording as shown in table 5.8. The RCA-CPHD has a significantly lower OSPA distance and lower MSE for both classifiers with respect to the CMM-CPHD and MM-CPHD.

	Raw data	MM-CPHD	CMM-CPHD	**RCA-CPHD**
Total Average OSPA Distance LRR [m]	-	1.30	1.29	**1.08**
Total Average OSPA Distance SSD [m]	-	2.11	2.10	**1.78**
Total Average MSE LRR [%]	4.54	1.21	0.33	**0.09**
Total Average MSE SSD [%]	9.50	9.17	9.05	**0.45**

Table 5.8: The total average tracking and classification accuracy for experiment 2. In bold the lowest error is highlighted per performance measure.

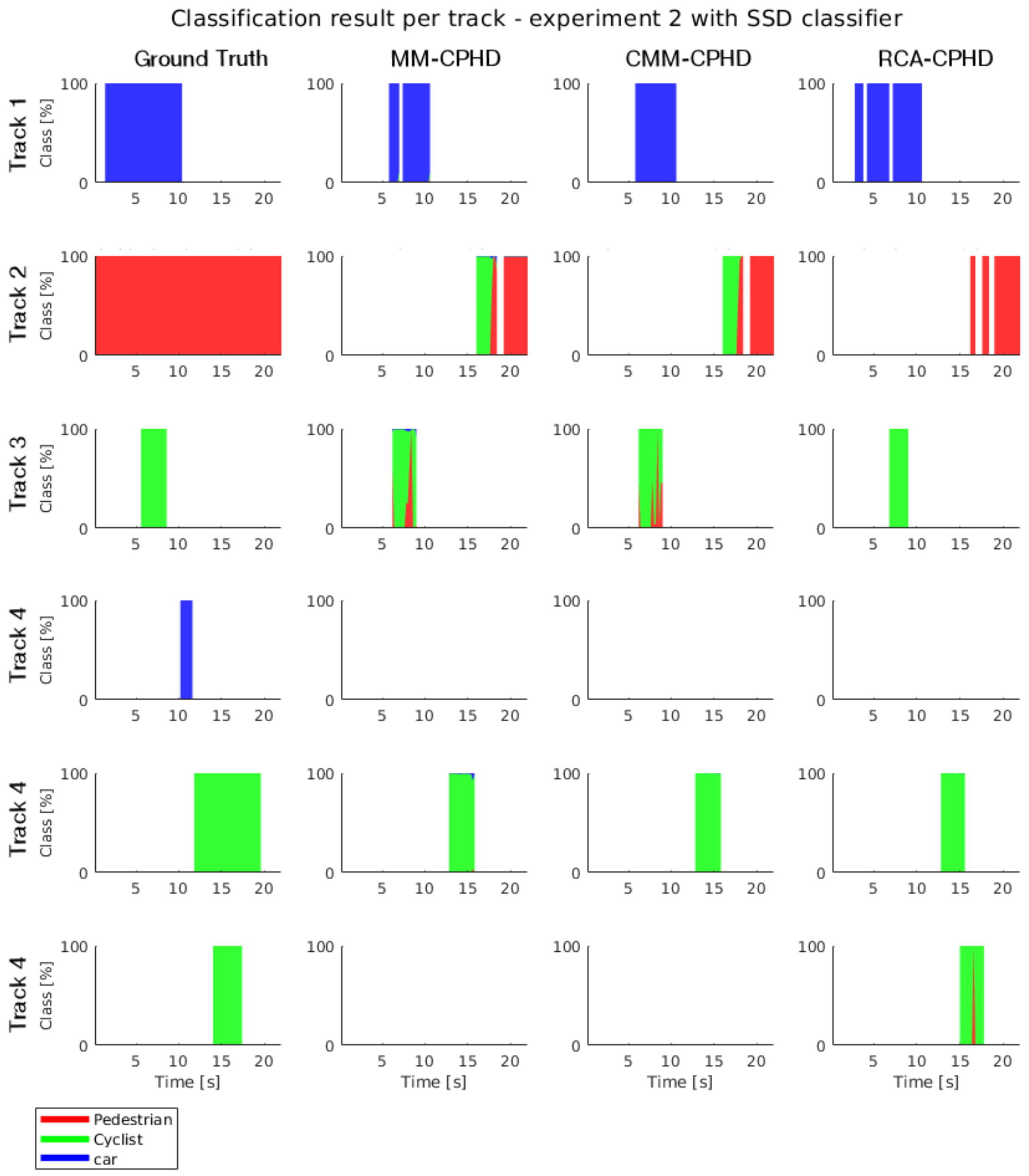


Figure 5.29: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

5.3.3 Experiment 3

The third experiment is a recording selected from the KITTI database, a database containing camera images, laser scans, high precision GPS measurements and IMU accelerations recorded while driving on German roads [22]. An overview of the intersection, from the viewpoint of the vehicle, is shown in figure 5.30.

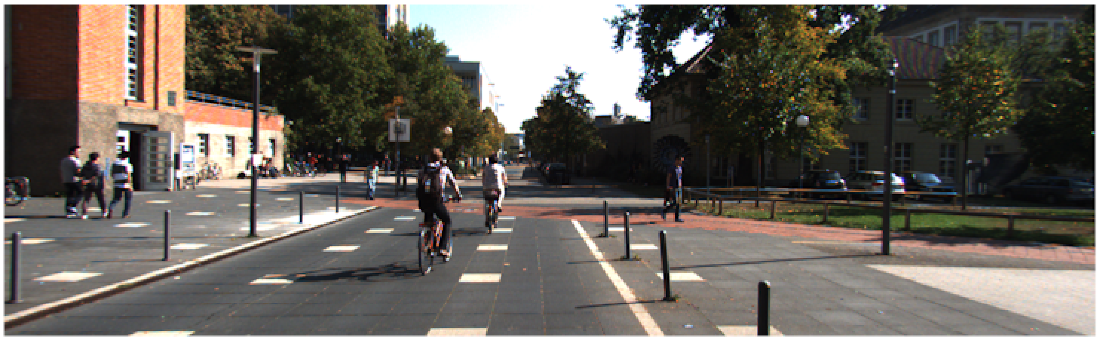


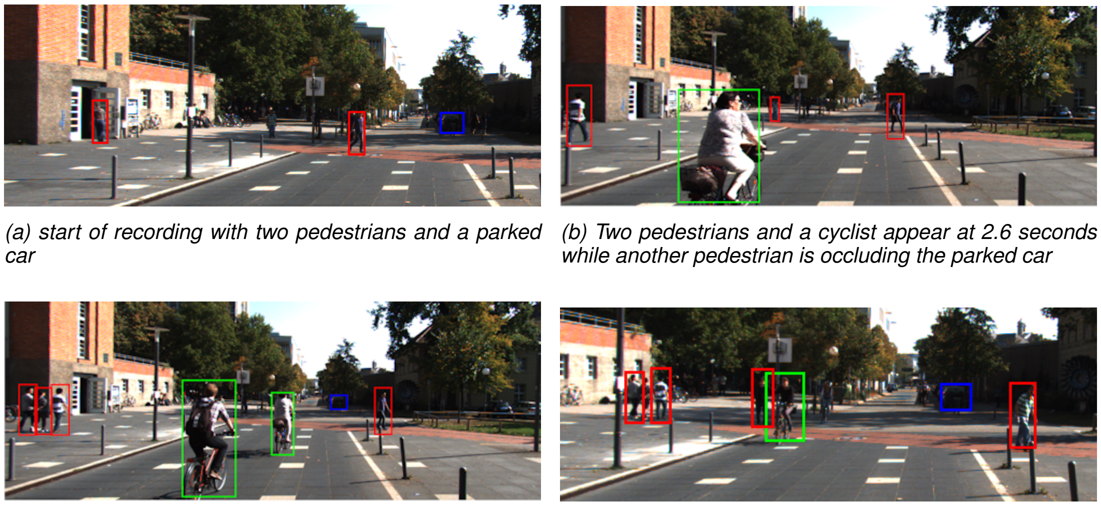
Figure 5.30: Experiment 3 is a traffic situation selected from the Kitti database. The ego vehicle is waiting in front of an intersection, while 7 pedestrians, 3 cyclist and 1 vehicle are observed.

In this experiment, the stereo colour camera recording is used, recorded at 5 hz with a resolution of 1224x370 pixels. The experiment recording has a total length of 15.2 seconds and the weather condition of the recording were slightly cloudy with no rain or dust. During the recording a total number of 11 objects are observed, consisting of 7 pedestrians, 3 cyclists and 1 car with a maximum of 9 objects at the same time.

Course of recording The recording starts with pedestrian 1 crossing the road ahead, another pedestrian 2 walking towards a building on a footpath on the left side of the ego-vehicle and a parked car on the background as shown in figure 5.31a. At 2.2 seconds pedestrian 1 starts occluding the parked car and at 2.4 seconds a new pedestrian 3 appears at the cyclist path. During the occlusion cyclist 1 and another pedestrian 4 are appearing at 2.6 seconds as shown in figure 5.31b. At 2.8 seconds the new appeared cyclist 1 occludes pedestrian 3 path till 3.4 seconds.

Subsequently at 3.6 seconds another two pedestrians, respectively pedestrian 5 and 6, and a cyclist 2 appear, resulting in a total number of 8 objects in the FOV. At 4 seconds, cyclist 2 occludes pedestrian 3 as shown in figure 5.31c and at 6.4 seconds pedestrian 7 appears on the cycle path. All objects then continue their way without occluding or crossing each other and at respectively 7.2, 8.4 and 9.4 seconds cyclist 1, cyclist 2 and pedestrian 1 leave the FOV.

At 11.2 seconds pedestrian 3 occludes the parked car. Subsequently at 12.6 seconds a new cyclist 3 appears in the field of view, which crosses pedestrian 7 at 13.6 seconds as shown in figure 5.31d. Cyclist 3 then occludes pedestrian 4,5 and 6 at 15 seconds.



(a) start of recording with two pedestrians and a parked car

(b) Two pedestrians and a cyclist appear at 2.6 seconds while another pedestrian is occluding the parked car

(c) Two pedestrians and cyclist appeared on the left and subsequently the cyclist occludes a pedestrian at 4 sec

(d) A new cyclist appears and crosses a pedestrian on the cycle path and 13.6 seconds

Figure 5.31: Overview course and events of experiment 3

Analysis

The performance is evaluated by two different measures, namely the tracking accuracy and the classification accuracy explained in more depth in section 5.1. First the tracking accuracy results are analysed for both classifiers and subsequently the classification accuracy.

Tracking accuracy The tracking accuracy results are shown in figure 5.32 and figure 5.33 for respectively the SSD and LRR classifier. During the initiation of the tracks in the first second, the MM-CPHD is mixing up the track of pedestrian 1 and the parked car for the SSD classifier. This results in a larger OSPA error in the first second for the MM-CPHD, first due to a cardinality and subsequently due to a larger localisation error caused by an incorrect measurement-to-track association. For the LRR classifier all tracks are correct initialized, resulting in virtually no difference between the different approaches in the first two seconds.

Between 2 and 2.5 seconds pedestrian 1 occludes the parked car and at the same time a cyclist and pedestrian are appearing in the FOV. The RCA-CPHD initiates the new tracks faster and is more able to distinguish the different tracks. This results in a lower OSPA for both the SSD and LRR classifier during this period of time. The appearance of two pedestrians and a cyclist at 3.6 seconds and the following occlusions between 4 and 4.4 seconds result in a lower OSPA for the RCA-CPHD compared to the MM-CPHD and CMM-CPHD for both classifiers.

Between 7.2 and 9.4 the two cyclists and a pedestrian are leaving the FOV. The RCA-CPHD is more able to distinguish the different tracks, with as a consequence a lower cardinality error for both classifiers. At 11.2 seconds, pedestrian 3 occludes the car, resulting in a lower OSPA distance for the RCA-CPHD with respect to the other filters for the SSD classifier. The crossing of pedestrian 7 and cyclist 3 at 13.6 results in a lower OSPA error for the LRR classifier.

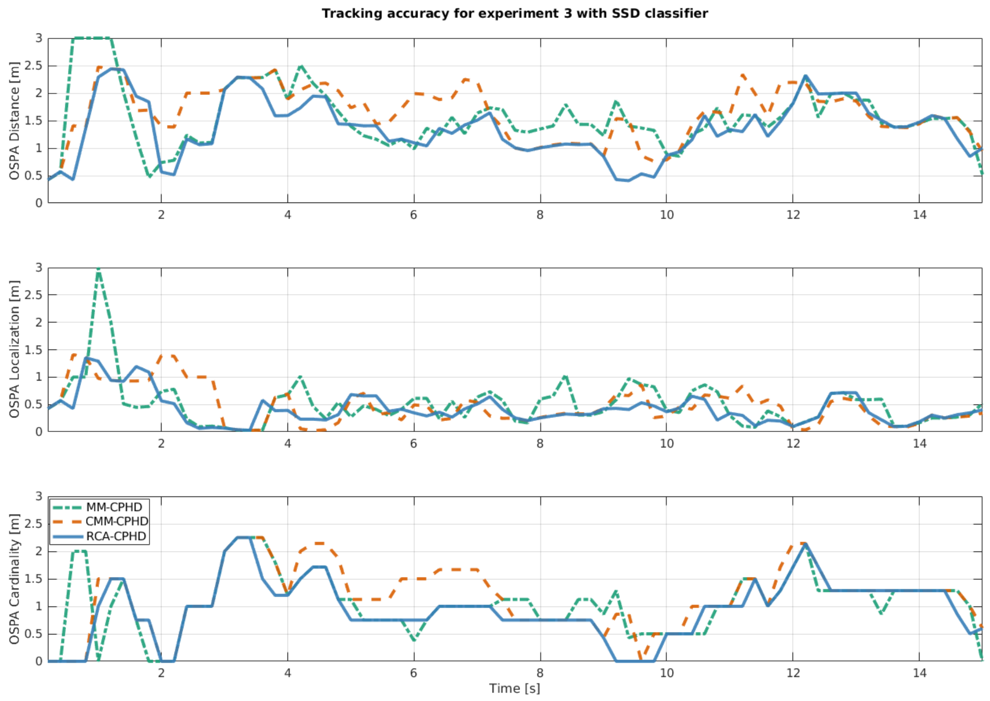


Figure 5.32: Tracking accuracy for experiment 3 using SSD classifier

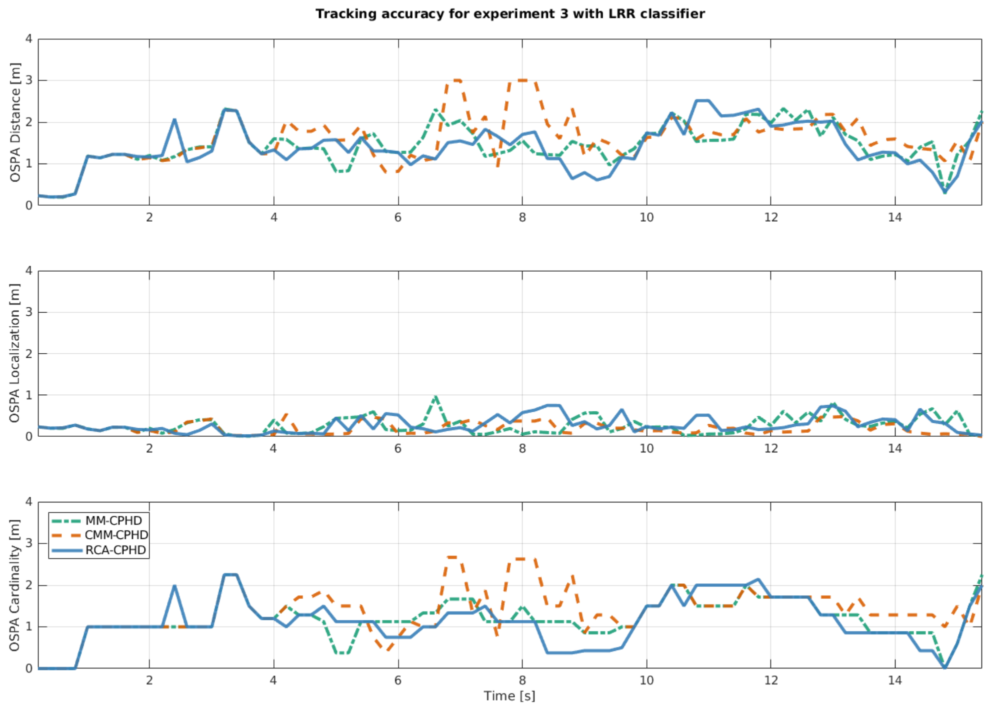


Figure 5.33: Tracking accuracy for experiment 4 using LRR classifier

Classification accuracy The classification accuracy for both the SSD and LRR classifier is shown in respectively figure 5.34 and 5.35. The MM-CPHD is mixing up the track of pedestrian 1 and the parked car for the SSD classifier during initialization of the tracks, resulting in a classification error in the first second. Between 2 and 3 seconds, pedestrian 1 is occluding the parked car. This creates a classification uncertainty, resulting in a classification error for the MM-CPHD and CMM-CPHD with the LRR classifier. This is in contrast to the RCA-CPHD, which shows virtually no error during the occlusion.

Between 3.6 and 4.6 the SSD creates multiple false alarms classifying the cyclists as both a cyclist and pedestrian. The false alarms are influencing the tracks of the MM-CPHD and CMM-CPHD, resulting in a peak in the MSE. The occlusion of cyclist 2 and pedestrian 1 at 4 seconds results in an increased MSE for the MM-CPHD with the LRR classifier.

Between 7.2 and 8 seconds, cyclist 2 and pedestrian 3 are approaching each other. The RCA-CPHD is able to distinguish the different tracks due to the difference in classification, while the MM-CPHD and CMM-CPHD are mixing up the two tracks. This result in a MSE in classification for the MM-CPHD and CMM-CPHD, while the RCA-CPHD has no error at all. The same principle then also happens at 13.6 seconds, when pedestrian 7 and cyclist 3 are crossing each other. This is visible at the peak in MSE for the MM-CPHD with the LRR classifier.

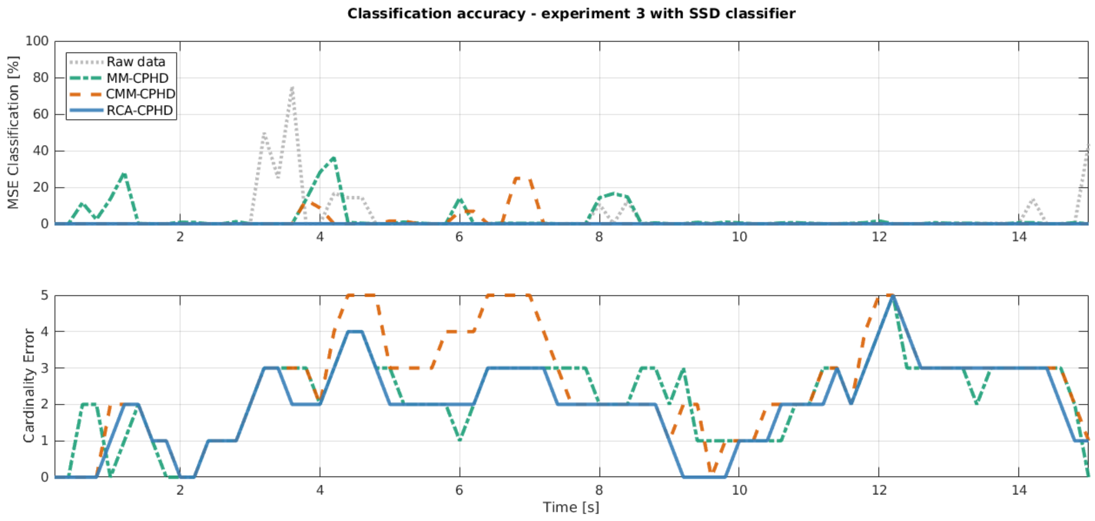


Figure 5.34: Classification accuracy for experiment 3 using SSD classifier

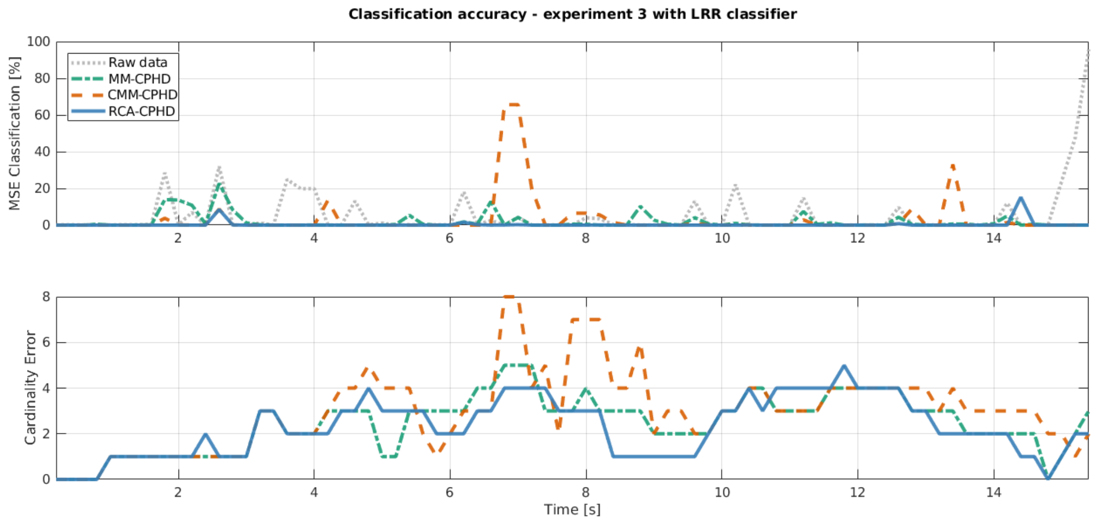


Figure 5.35: Classification accuracy for experiment 3 using LRR classifier

Summary of results

The results of experiment 3 show that the RCA-CPHD has a higher tracking- and classification accuracy in classification uncertain moments, e.g. occlusions, crossings and misclassifications, with respect to the MM-CPHD and CMM-CPHD. This is also seen in the average OSPA distances and average MSE over the entire recording as shown in table 5.9. The RCA-CPHD has a significantly lower OSPA distance and lower MSE for both classifiers with respect to the CMM-CPHD and MM-CPHD.

	Raw data	MM-CPHD	CMM-CPHD	**RCA-CPHD**
Total Average OSPA Distance LRR [m]	-	1.46	1.59	**1.39**
Total Average OSPA Distance SSD [m]	-	1.56	1.64	**1.36**
Total Average MSE LRR [%]	5.52	1.81	3.19	**0.35**
Total Average MSE SSD [%]	3.69	2.81	1.18	**0.00**

Table 5.9: The total average tracking and classification accuracy for experiment 3. In bold the lowest error is highlighted per performance measure.

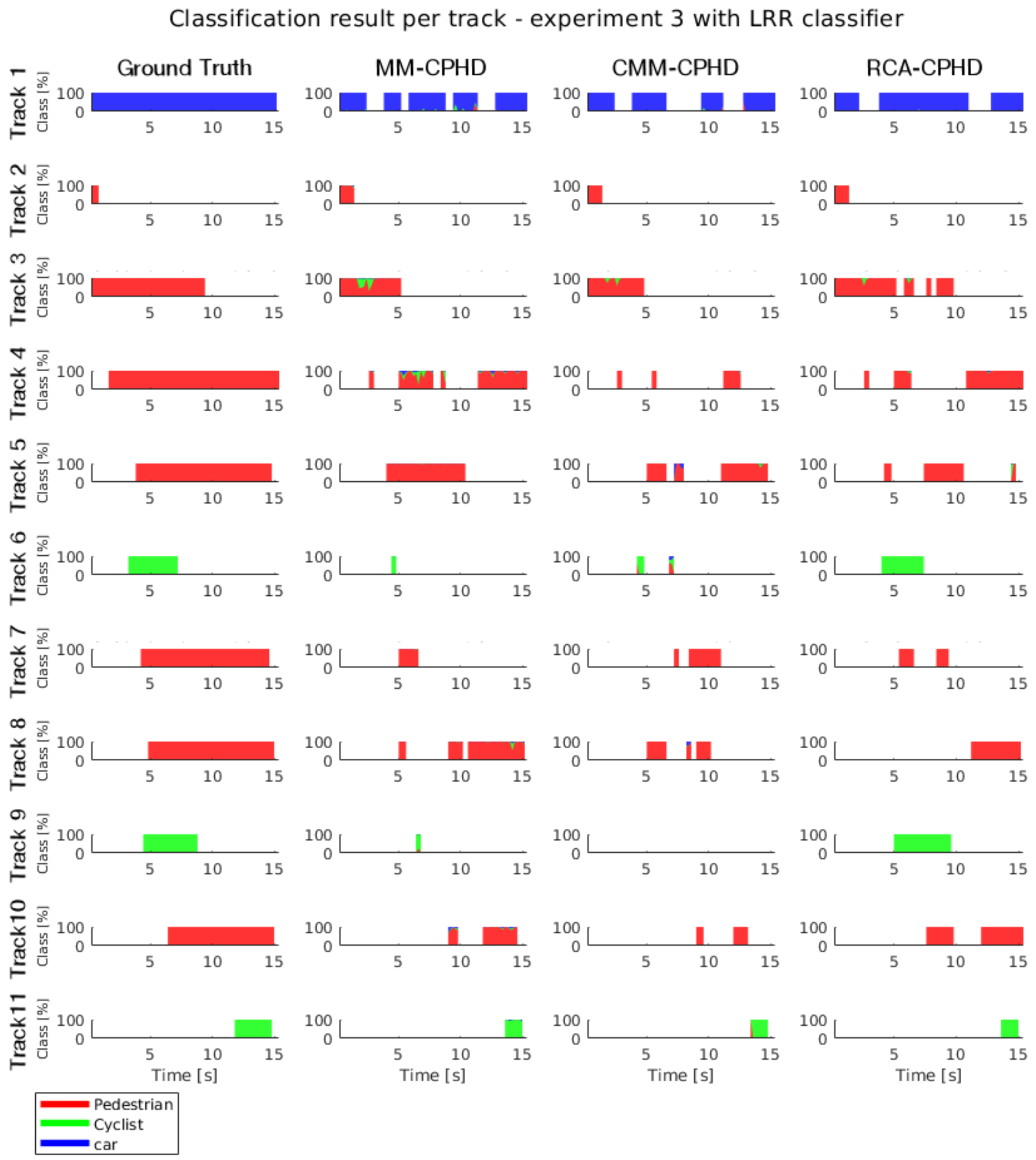


Figure 5.36: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

5.3.4 Experiment 4

Experiment 4 is a recording selected from the KITTI database, a database containing camera images, laser scans, high precision GPS measurements and IMU accelerations recorded while driving on Germand roads [22]. An overview of the intersection, from the viewpoint of the vehicle, is shown in figure 5.37.

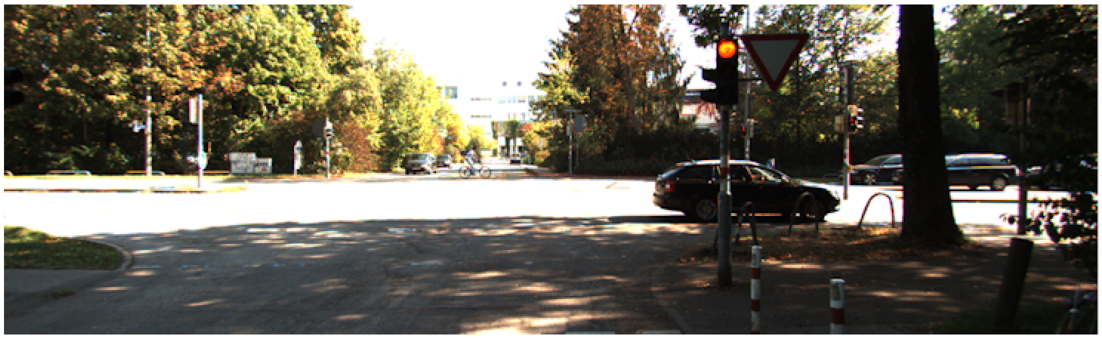
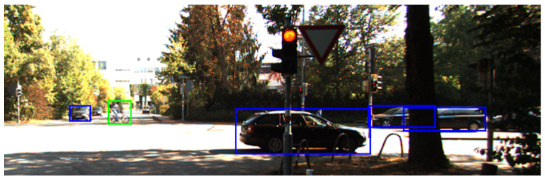


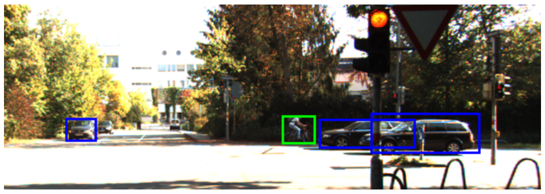
Figure 5.37: Experiment 4 is a traffic situation selected from the KITTI database. The ego vehicle is waiting in front of an intersection, while 10 cars, 1 motorcycle and 1 cyclist are crossing the intersection.

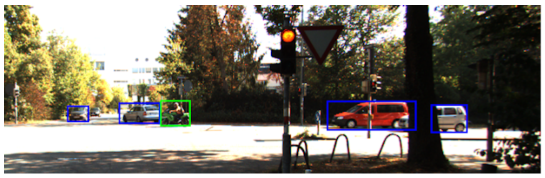
In this experiment, the stereo colour camera recording is used, recorded at 5 hz with a resolution of 1224x370 pixels. The experiment recording has a total length of 14 seconds and the weather condition of the recording were sunny. During the recording a total number of 12 objects are observed, consisting of 10 cars, 1 motorcycle and 1 cyclist with a maximum of 5 objects at the same time.

Course of recording The recording starts with car 1 crossing the road ahead, cars 3 and 4 waiting to cross the road ahead in front of a traffic light, another car 2 waiting in front of a traffic light on the other side of the road and a cyclist crossing the cycle road a head as shown in figure 5.38a. After 2 seconds, car 3 and 4 start driving and sequentially occlude the cyclist from 2.8 to 3.8 seconds as shown in figure 5.38b. Meanwhile at 2.4 second car 5 and 6 arrive in the FOV and occlude the cyclist from 4 to 5 seconds. At 4.8 seconds car 7 arrives in the field of view, which subsequently occludes the cyclist at 5.4 seconds as shown in figure 5.38c .

At 8 seconds another car 8 appears followed by a motorcycle at 8.6 seconds. Subsequently at 11.6 seconds car 8 turns off the main road, after which it is passed by the motorcycle as shown in figure 5.38d. The motorcycle then occludes car 1. Meanwhile at respectively 10.8, 11.4 and 13 seconds car 9, 10 and 11 arrive. Car 9 has a higher speed with respect to the rest and thereby overtakes the motorcycle at 13.2 seconds as shown in figure 5.38d.


(a) Start of recording with car 1 crossing the road, car 2 on the other side, car 3 and 4 waiting to cross the road and a cyclist on the cycle path


(b) From 2.8 to 3.8 seconds car 3 and 4 are occluding the cyclist


(c) At 11.6 seconds car 8 turns off the main road, after which it is passed by the motorcycle

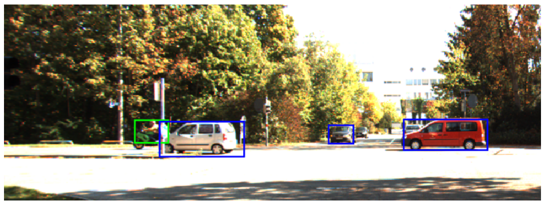

(d) At 13.2 seconds car 9 overtakes the motorcycle

Figure 5.38: Overview course and events of experiment 4

Analysis

The performance is evaluated by two different measures, namely the tracking accuracy and the classification accuracy explained in more depth in section 5.1. First the tracking accuracy results are analysed for both classifiers and subsequently the classification accuracy.

Tracking accuracy The tracking accuracy results are shown in figure 5.39 and figure 5.40 for respectively the SSD and LRR classifier. At 1.4 seconds the SSD classifier classifies the rider as pedestrian, while the bike is not detected. Due to the difference in classification, the RCA-CPHD is not using the rider for updating the cyclist's track. The MM-CPHD and CMM-CPHD only take into account the location and therefore use the incorrect measurement. This results in a higher OSPA distance of the MM-CPHD and CMM-CPHD with respect to the RCA-CPHD at 1.4 seconds with the SSD classifier.

After 2 seconds, car 3 and 4 start driving and sequentially occlude the cyclist from 2.8 to 3.8 seconds as shown in figure 5.38b. During the occlusion, the OSPA distance is lower for the RCA-CPHD with respect to the other approaches. The RCA-CPHD also initiates car 5 and 6 faster at 2.4 seconds with respect to the MM-CPHD and CMM-CPHD with the LRR classifier, resulting in a lower OSPA error.

At 8.6 seconds the motorcycle appears, which drives close behind the car in front of him. The MM-CPHD and CMM-CPHD are mixing up both tracks, resulting in a higher OSPA distance with respect to the RCA-CPHD. At 11.6 seconds, the car turns off the main road and is overtaken by the motorcycle. The RCA-CPHD is more able to distinguish both tracks, resulting in a lower OSPA distance with both classifiers.

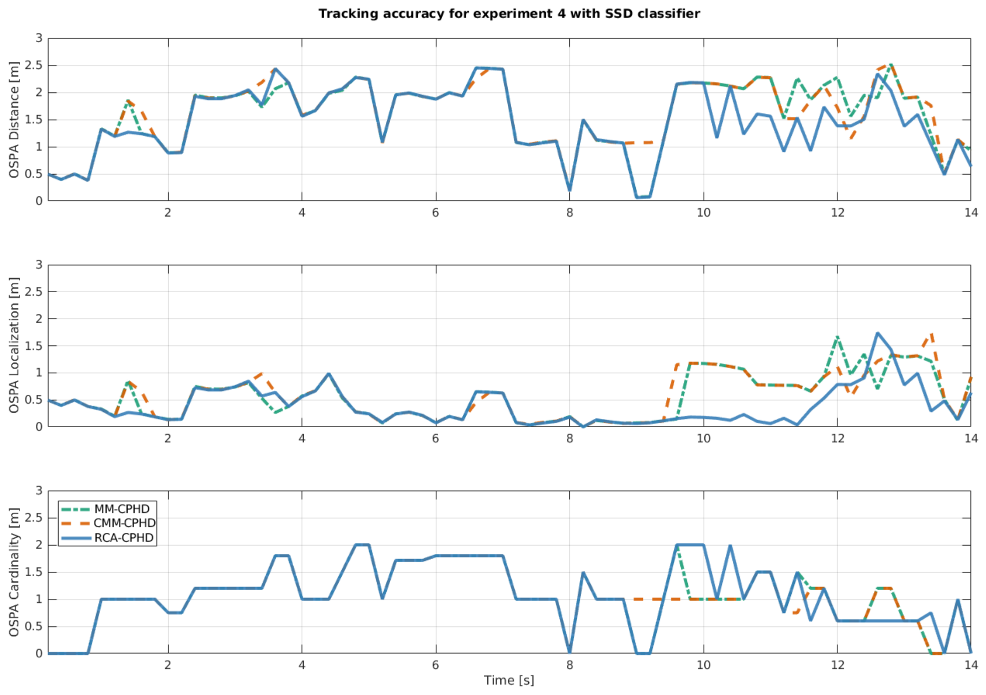


Figure 5.39: Tracking accuracy for experiment 4 using SSD classifier

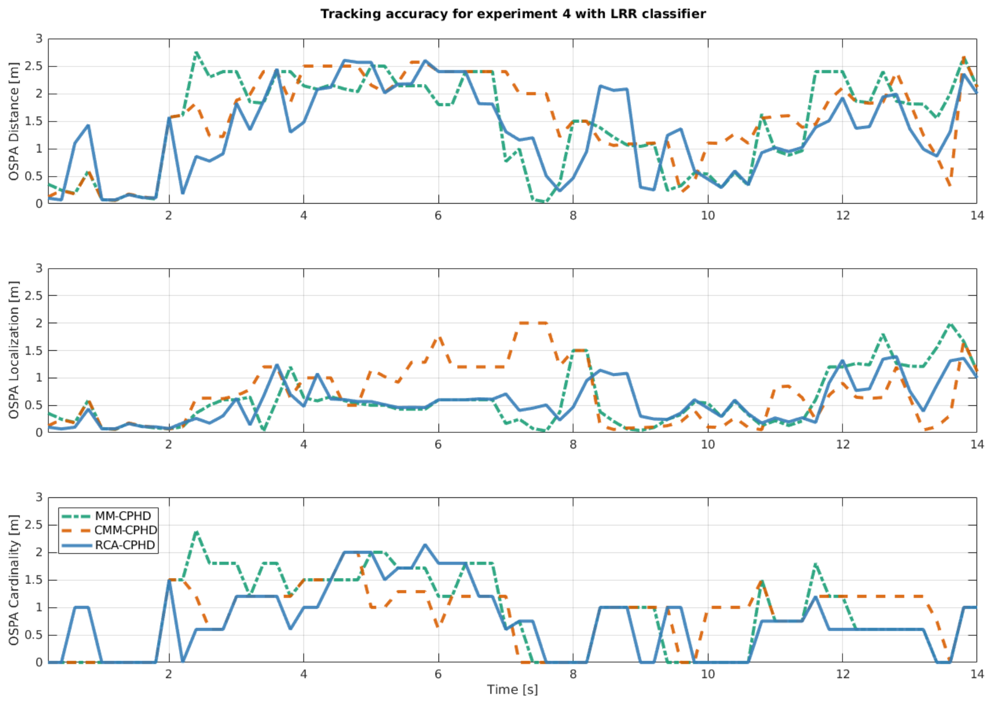


Figure 5.40: Tracking accuracy for experiment 4 using LRR classifier

Classification accuracy The classification accuracy for both the SSD and LRR classifier is shown in respectively figure 5.41 and 5.42. At 1.4 seconds the SSD classifier classifies the rider as pedestrian, while the bike is not detected. This results in a classification error for the CMM-CPHD and MM-CPHD, while the RCA-CPHD has no classification error.

After 2 seconds, car 3 and 4 start driving and sequentially occlude the cyclist from 2.8 to 3.8 seconds as shown in figure 5.38b. During this occlusion, the CMM-CPHD and MM-CPHD are using the measurements from the car to update the cyclist's track. This results in a larger classification MSE with respect to the RCA-CPHD. The classification error is also clearly visible in the classification results of the individual tracks visualized in figure 5.43 between 2.8 and 3.8 seconds for track 3.

At 8.6 seconds the motorcycle appears, which drives close behind the car in front of him. The MM-CPHD and CMM-CPHD are mixing up both tracks, resulting in a classification error. The RCA-CPHD is able to separate both tracks and has no error during this period of time.

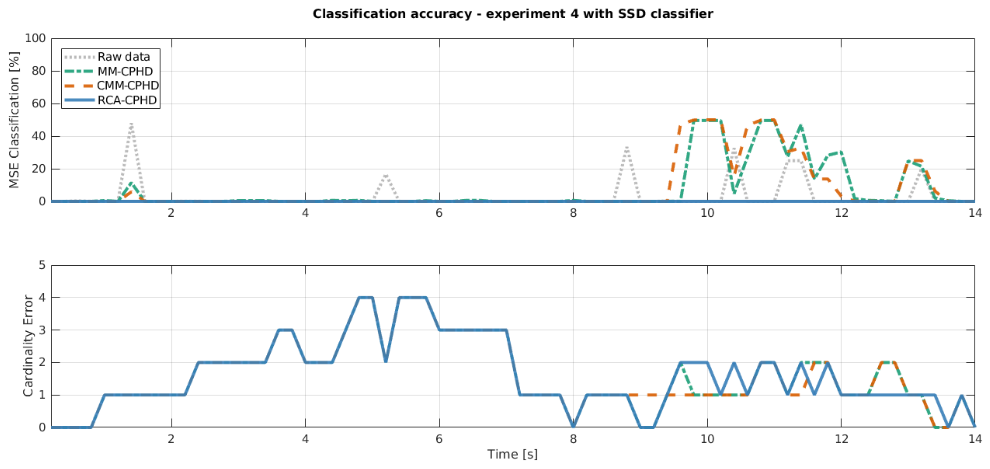


Figure 5.41: Classification accuracy for experiment 4 using SSD classifier

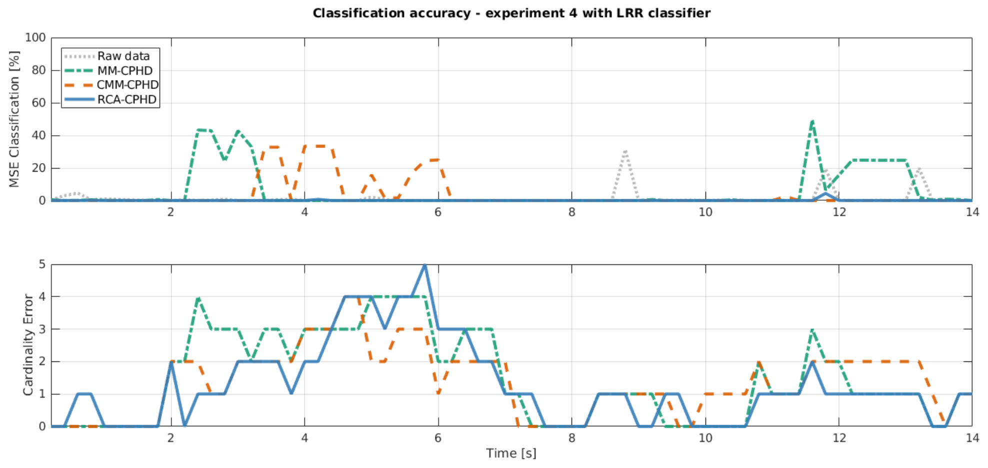


Figure 5.42: Classification accuracy for experiment 4 using LRR classifier

Summary of results

In the results of experiment 4 is seen that the RCA-CPHD has a higher tracking- and classification accuracy during classification uncertain conditions with respect to the MM-CPHD and CMM-CPHD. This is also seen in the average OSPA distances and average MSE over the entire recording as shown in table 5.10. The RCA-CPHD has a significantly lower OSPA distance and lower MSE for both classifiers with respect to the CMM-CPHD and MM-CPHD.

	Raw data	MM-CPHD	CMM-CPHD	**RCA-CPHD**
Total Average OSPA Distance LRR [m]	-	1.45	1.55	**1.31**
Total Average OSPA Distance SSD [m]	-	1.59	1.62	**1.46**
Total Average MSE LRR [%]	1.28	5.56	3.61	**0.08**
Total Average MSE SSD [%]	2.87	7.08	7.36	**0.00**

Table 5.10: The total average tracking and classification accuracy for experiment 4. In bold the lowest error is highlighted per performance measure.

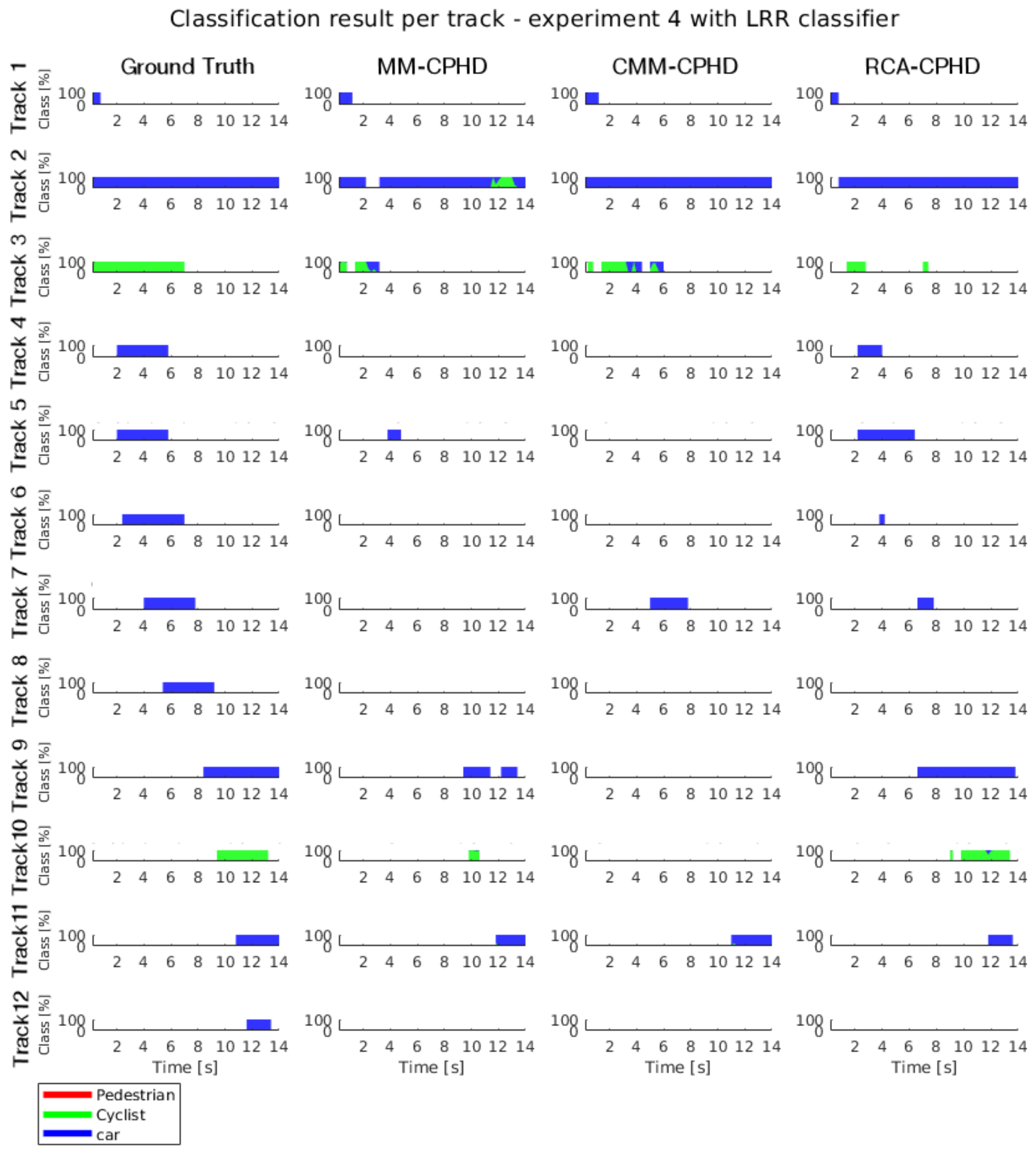


Figure 5.43: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

5.3.5 Experiment 5

The fifth experiment is a recording selected from the KITTI database, a database containing camera images, laser scans, high precision GPS measurements and IMU accelerations recorded while driving on Germand roads [22]. An overview of the intersection, from the viewpoint of the vehicle, is shown in figure 5.44.

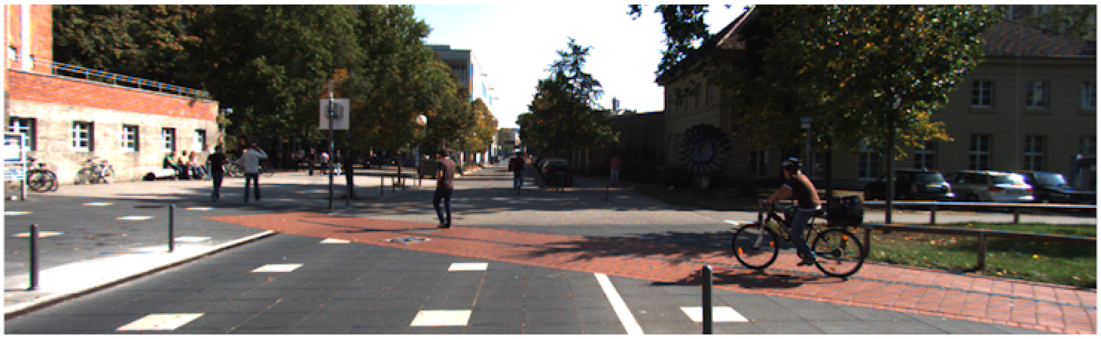


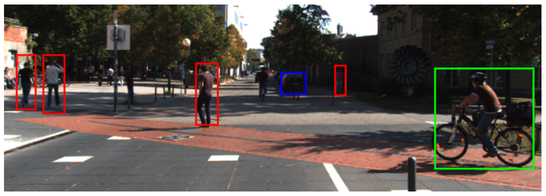
Figure 5.44: Experiment 5 is a traffic situation selected from the Kitti database. The ego vehicle is waiting in front of an intersection, while 5 pedestrians, 5 cyclist and 1 car pass.

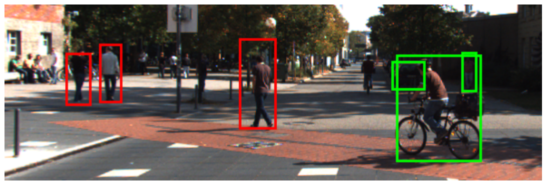
In this experiment, the stereo colour camera recording is used, recorded at 5 hz with a resolution of 1224x370 pixels. The experiment recording has a total length of 15 seconds and the weather condition of the recording were slightly cloudy with no rain or dust. During the recording a total number of 11 objects are observed, consisting of 5 pedestrians, 5 cyclists and 1 car with a maximum of 8 objects at the same time.

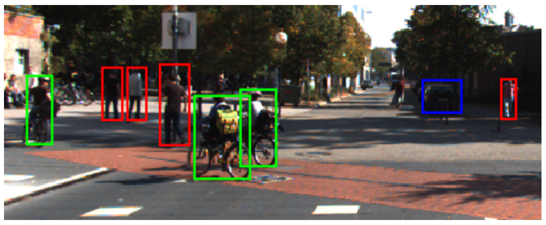
Course of recording The experiment starts with three pedestrians 1, 2 and 3 walking and cyclist 1 cycling on a cycle path in front of the car, a pedestrian 4 walking on the other side of the intersection and a car parked on the background as shown in figure 5.45a. After 0.6 seconds cyclist 1 occludes first pedestrian 4, subsequently the car at 1 second and crosses then pedestrian 1 at 1.8 seconds as shown in figure 5.45b.

Meanwhile at 1.2 seconds two cyclists 2 and 3 appear, which also occlude the pedestrian 4, car and pedestrian 1 at respectively 2.2, 2.4 and 3.6 seconds as shown in figure 5.45c. At 4 seconds cyclist 2 and 3 are crossing pedestrian 2 and 3 and subsequently disappear at 5.8 seconds.

At 6.6 seconds pedestrian 5 appears, followed by cyclist 4 at 9.6 seconds. Cyclist 4 crosses pedestrian 5at 10.4 seconds and occludes at the same time first pedestrian 4 and subsequently the parked car as shown in figure 5.45d. Meanwhile cyclist 5 appears in the FOV, which first occludes pedestrian 4 at 11 seconds, then occludes the car at 11.4 seconds and subsequently crosses pedestrian 4 at 11.6 seconds. All road users then continue their way without occluding or crossing each other.


(a) start of recording with four pedestrians, 1 cyclist and a parked car


(b) cyclist 1 occludes first pedestrian 4 and subsequently the parked car at 0.6 seconds


(c) Cyclists 2 and 3 first occlude both the car and pedestrian 4 and then pedestrian 2 and 3 at 5.8 seconds.

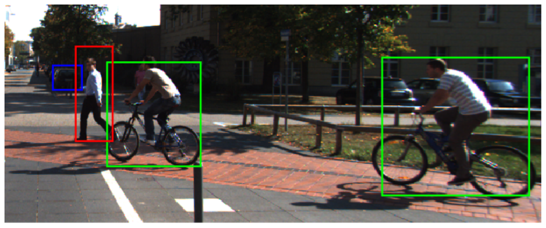

(d) Cyclist 4 crosses pedestrian 5 and is occluding pedestrian 4 and the car

Figure 5.45: Overview course and events of experiment 5

Analysis

The performance is evaluated by two different measures, namely the tracking accuracy and the classification accuracy explained in more depth in section 5.1. First the tracking accuracy results are analysed for both classifiers and subsequently the classification accuracy.

Tracking accuracy Between 0.6 seconds and 1 second cyclist 1 occludes first pedestrian 4 and the parked car and subsequently crosses pedestrian 1 at 1.8 seconds. Meanwhile at 1.2 seconds two cyclists 2 and 3 appear, which also occlude the pedestrian 4, car and pedestrian 1 at respectively 2.2, 2.4 and 3.6 seconds as shown in figure 5.45c. During the occlusion and crossing, the OSPA distance is smaller for the RCA-CPHD with respect to the other approaches.

At 4 seconds cyclist 2 and 3 are crossing pedestrian 2 and 3, with as a consequence a lower OSPA distance with respect to the MM-CPHD and CMM-CPHD with the LRR classifier.

Cyclist 4 crosses pedestrian 5 at 10.4 seconds and occludes at the same time first pedestrian 4 and subsequently the parked car as shown in figure 5.45d. Meanwhile cyclist 5 appears in the FOV, which first occludes pedestrian 4 at 11 seconds, then occludes the car at 11.4 seconds and subsequently crosses pedestrian 4 at 11.6 seconds. During the occlusion and crossing, the OSPA distance is smaller for the RCA-CPHD with respect to the other approaches with both classifiers.

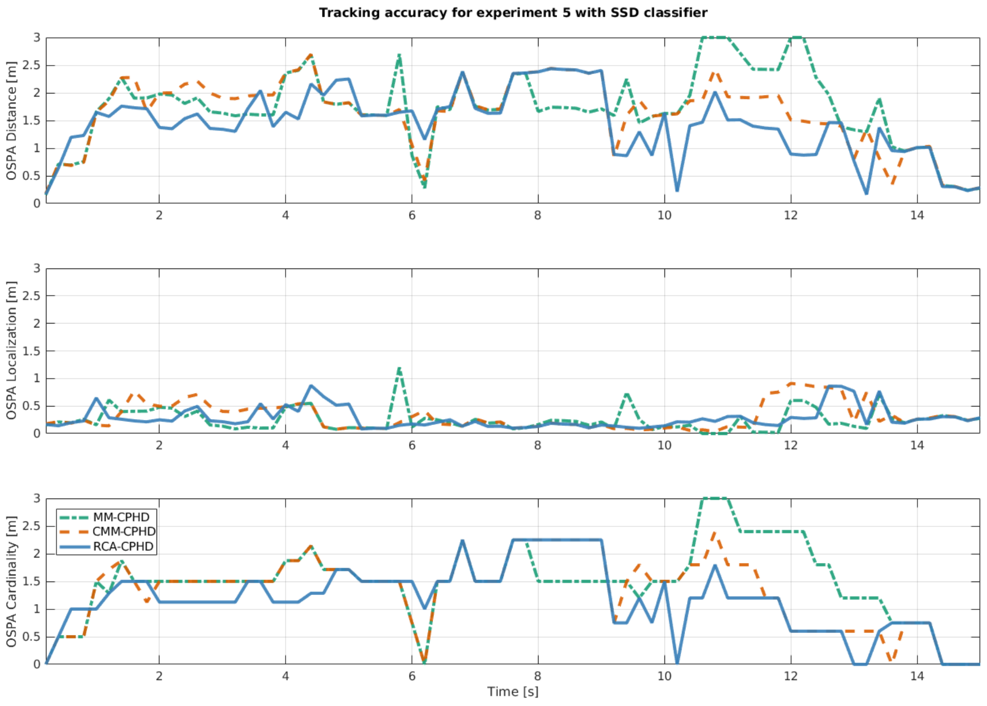


Figure 5.46: Tracking accuracy for experiment 5 using SSD classifier

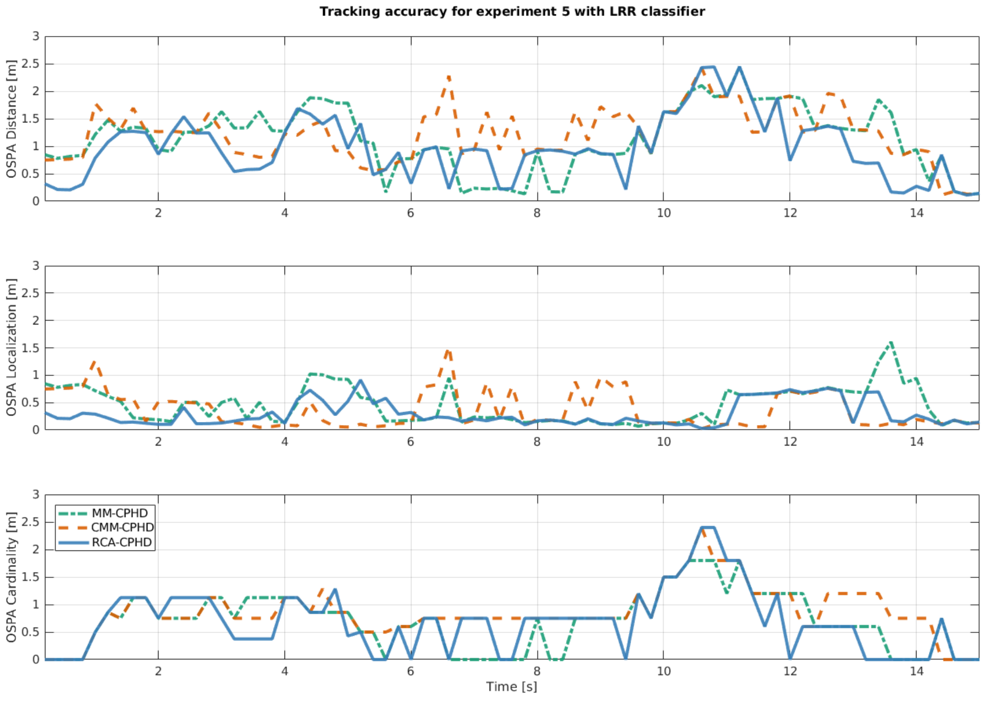


Figure 5.47: Tracking accuracy for experiment 5 using LRR classifier

Classification accuracy The classification accuracy for both the SSD and LRR classifier is shown in respectively figure 5.48 and 5.49. The multiple occlusions and crossings between 0.6 and 4.5 seconds, result in an classification error for the MM-CPHD and CMM-CPHD, while the RCA-CPHD has no error during these classification uncertain conditions. The classification error due to the occlusions is also clearly visible in the classification result for the individual tracks shown in figure 5.50 for track 1 and track 4.

Between 6 and 10 seconds the car's track is several times updated by measurements from pedestrian 4 with the LRR classifier. During these moments a classification error is visible for the CMM-CPHD shown in figure 5.49.

Between 10.4 and 11.6 seconds there are several occlusions and crossings after each other, with as a consequence a classification error for the MM-CPHD and CMM-CPHD with both classifiers. The RCA-CPHD however has no error during these classification uncertainty.

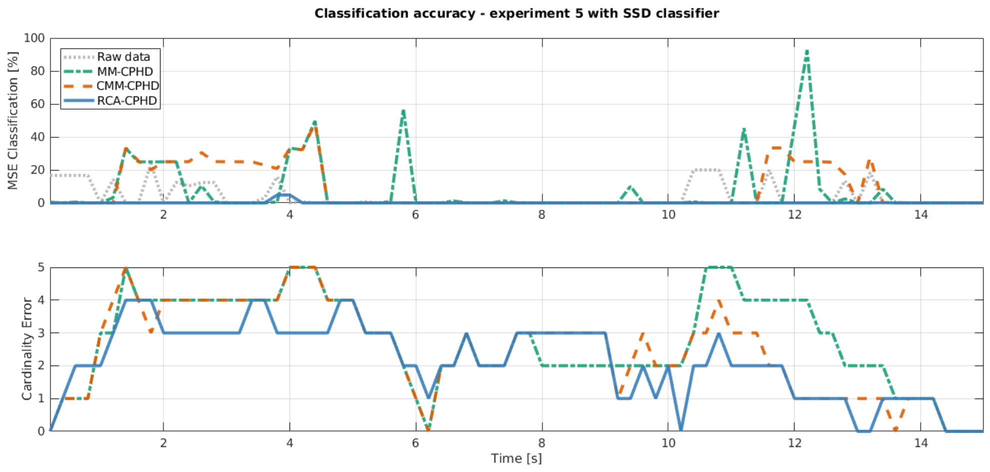


Figure 5.48: Classification accuracy for experiment 5 using SSD classifier

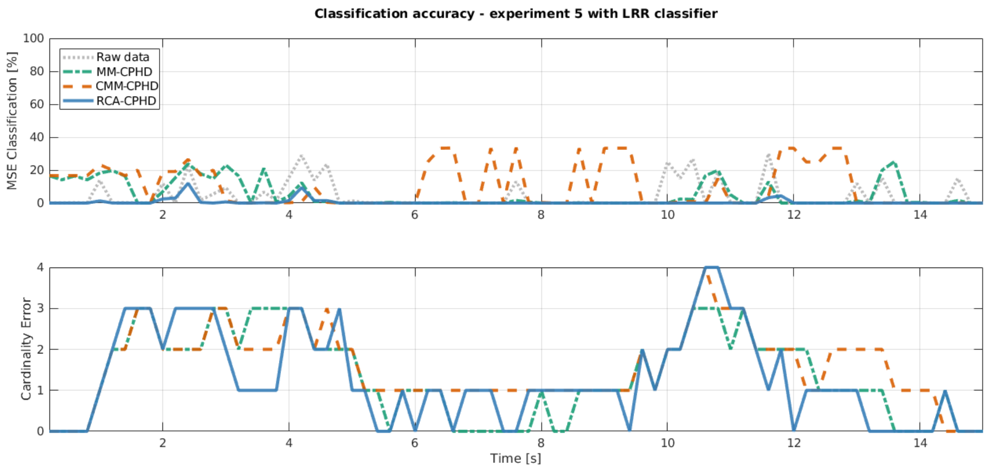


Figure 5.49: Classification accuracy for experiment 5 using LRR classifier

Summary of results

The results of experiment 5 show that the RCA-CPHD has a higher tracking- and classification accuracy during the high classification uncertainty moments with respect to the MM-CPHD and CMM-CPHD. This is also seen in the average OSPA distances and average MSE over the entire recording as shown in table 5.11. The RCA-CPHD has a significantly lower OSPA distance and lower MSE for both classifiers with respect to the CMM-CPHD and MM-CPHD.

	Raw data	MM-CPHD	CMM-CPHD	**RCA-CPHD**
Total Average OSPA Distance LRR [m]	-	1.14	1.21	**0.97**
Total Average OSPA Distance SSD [m]	-	1.72	1.64	**1.44**
Total Average MSE LRR [%]	4.43	5.14	10.11	**0.56**
Total Average MSE SSD [%]	3.81	7.19	8.72	**0.13**

Table 5.11: The total average tracking and classification accuracy for experiment 5. In bold the lowest error is highlighted per performance measure.

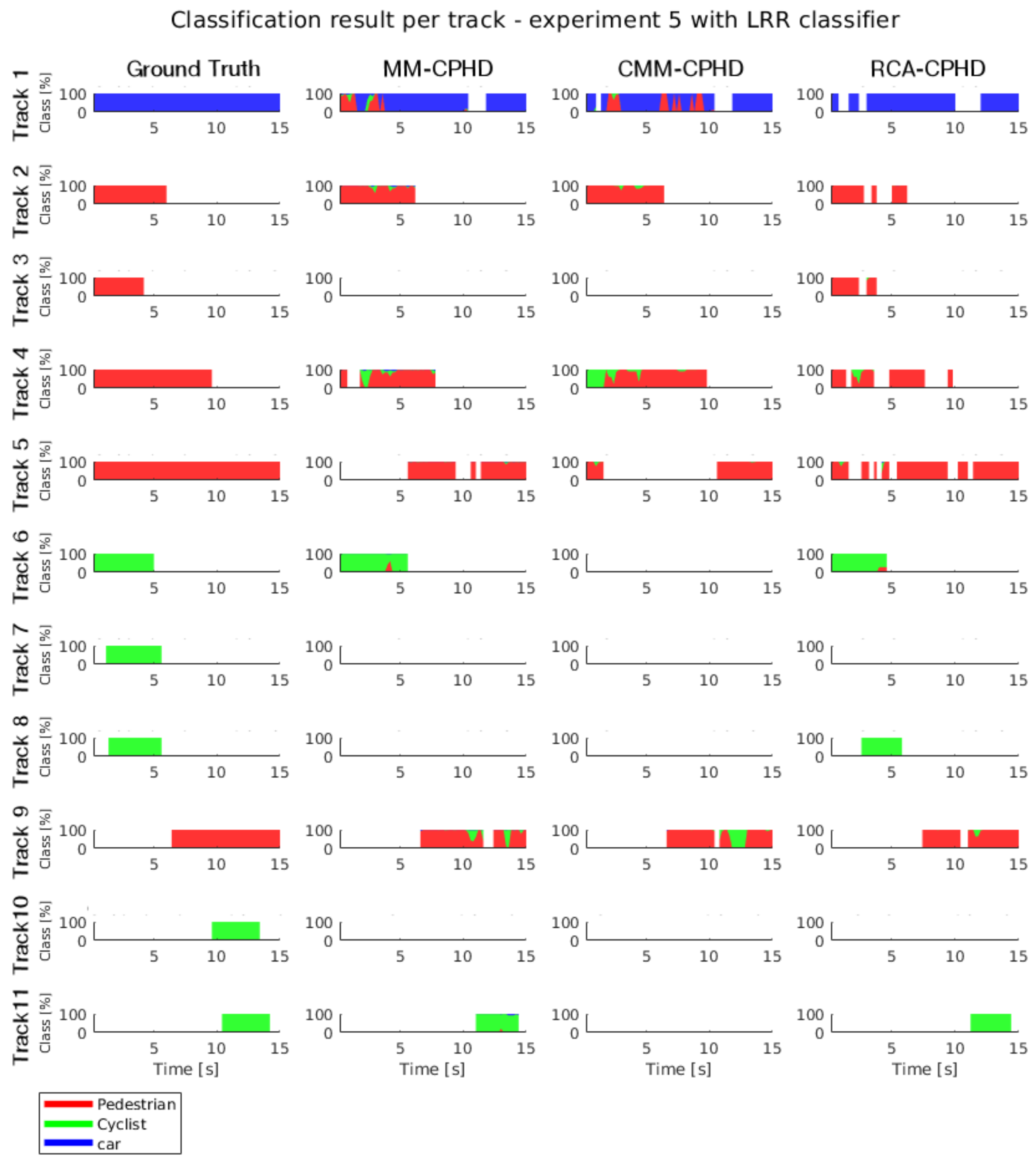


Figure 5.50: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

5.3.6 Experiment 6

Experiment 6 is a recording selected from the KITTI database, a database containing camera images, laser scans, high precision GPS measurements and IMU accelerations recorded while driving on Germand roads [22]. An overview of the intersection, from the viewpoint of the vehicle, is shown in figure 5.51.

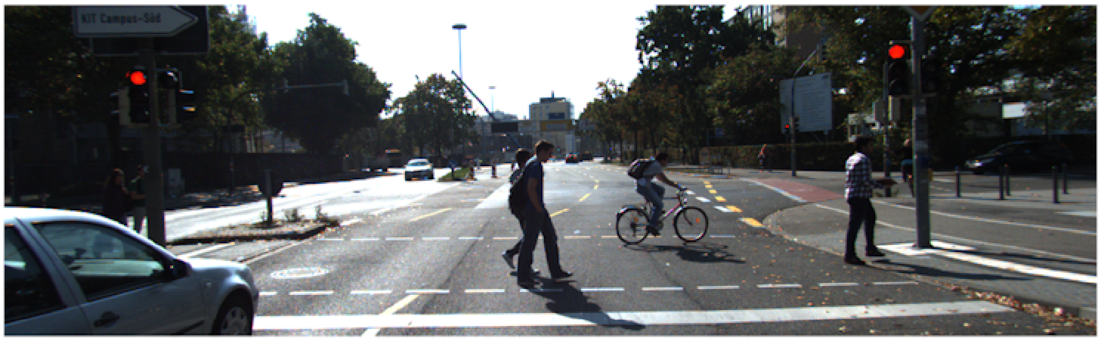
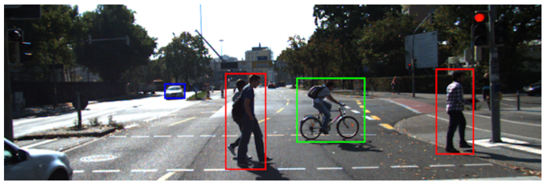


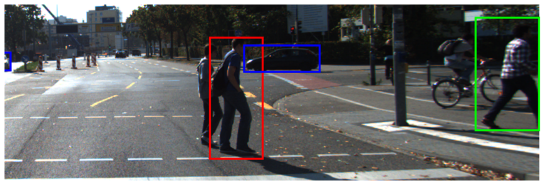
Figure 5.51: Experiment 6 is a traffic situation selected from the Kitti database. The ego vehicle is waiting in front of a traffic light, while 5 pedestrians, 1 cyclist and 2 cars are passing.

In this experiment, the stereo colour camera recording is used, recorded at 5 hz with a resolution of 1224x370 pixels. The experiment recording has a total length of 15 seconds and the weather condition of the recording were cloudy with no rain or dust. During the recording a total number of 8 objects are observed, consisting of 5 pedestrians, 1 cyclist and 2 cars with a maximum of 7 objects at the same time.

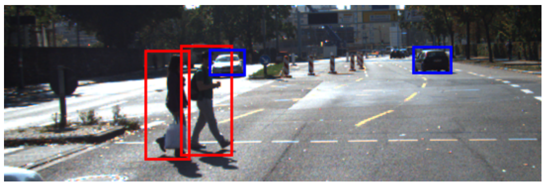
Course of recording The recording starts with pedestrian 1 ,2 and 3 and cyclist 1 crossing a pedestrian crossover in front of the ego vehicle and car 1 is waiting in front of a traffic light at the other side of the intersection as shown in figure 5.52a. After 1.4 seconds, two pedestrians 4 and 5 and a car 2 appear in the FOV. At 1.8 seconds car 2 is occluded by pedestrian 1 and 2 and at the same time pedestrian 3 and cyclist 1 are crossing each other as shown in figure 5.52b.



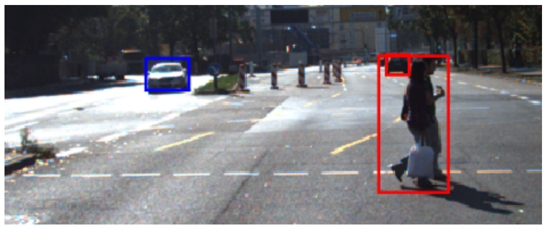
(a) start of recording with pedestrians 1,2,3 crossing the pedestrian crossover and car 1 waiting in front of a traffic light



(b) At 1.8 seconds car 2 is occluded by pedestrian 1 and 2, while at the same time pedestrian 3 and cyclist 1 are crossing each other



(c) At 3.8 seconds pedestrians 4 and 5 are occluding car 1



(d) At 5.8 seconds pedestrian 4 and 5 are occluding car 2

Figure 5.52: Overview course and events of experiment 6

At 3.8 seconds pedestrian 4 and 5 are occluding car 1 while crossing the pedestrian crossover as shown in figure 5.52c. The same happens for car 2, when it is occluded by pedestrian 4 and 5 at 5.8 seconds as shown in figure 5.52d. Subsequently all road users continue their way without occluding or crossing each other. At 7.6 seconds car 1 starts driving and turns into the side street.

Analysis

The performance is evaluated by two different measures, namely the tracking accuracy and the classification accuracy explained in more depth in section 5.1. First the tracking accuracy results are analysed for both classifiers and subsequently the classification accuracy.

Tracking accuracy The tracking accuracy results are shown in figure 5.53 and figure 5.54 for respectively the SSD and LRR classifier. At 1.4 seconds the number of objects is increased with 3 objects, resulting in a cardinality error. The RCA-CPHD initiates the new objects faster, resulting in a lower OSPA distance with respect to the other approaches. At 1.8 seconds the class uncertainty increases due to an occlusion of car 2 by pedestrian 1 and 2 and at the same time a crossing of pedestrian 3 and cyclist 1 as shown in figure 5.52b. During these classification uncertain period of time, the RCA-CPHD has a lower OSPA distance with respect to the MM-CPHD and CMM-CPHD.

At 5.8 seconds car 2 is occluded by pedestrian 3 and 5. During the occlusion, the RCA-CPHD has a more accurate cardinality estimate with the SSD classifier, resulting in a lower OSPA distance.

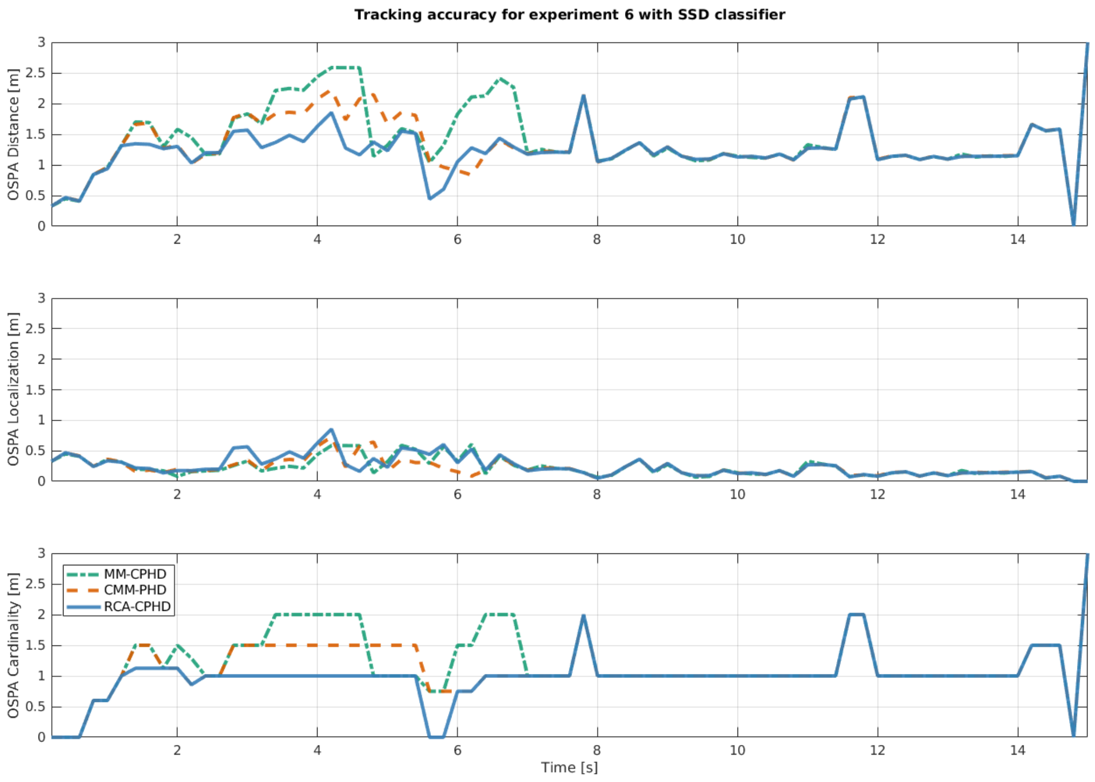


Figure 5.53: Tracking accuracy for experiment 6 using SSD classifier

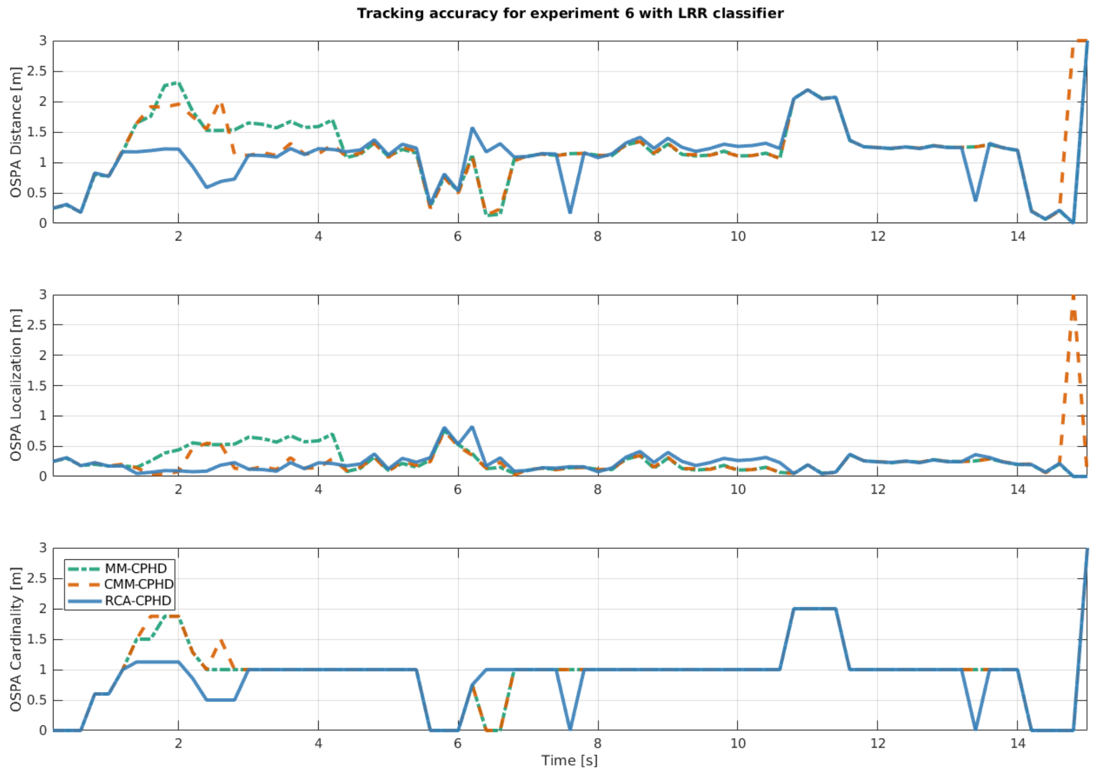


Figure 5.54: Tracking accuracy for experiment 6 using LRR classifier

Classification accuracy The classification accuracy for both the SSD and LRR classifier is shown in respectively figure 5.55 and 5.56. At 0.6 seconds the SSD classifier misclassified the cyclist as a pedestrian, resulting in a classification error for the MM-CPHD and CMM-CPHD. The RCA-CPHD, however, is not using the misclassification in the update step and has therefore no error.

At 1.8 seconds the occlusion is clearly visible with the LRR classifier, where both the MM-CPHD and the CMM-CPHD have a classification error. The other occlusion at 5.8 second also results in a MSE for the MM-CPHD and CMM-CPHD, while the RCA-CPHD has no error at all.

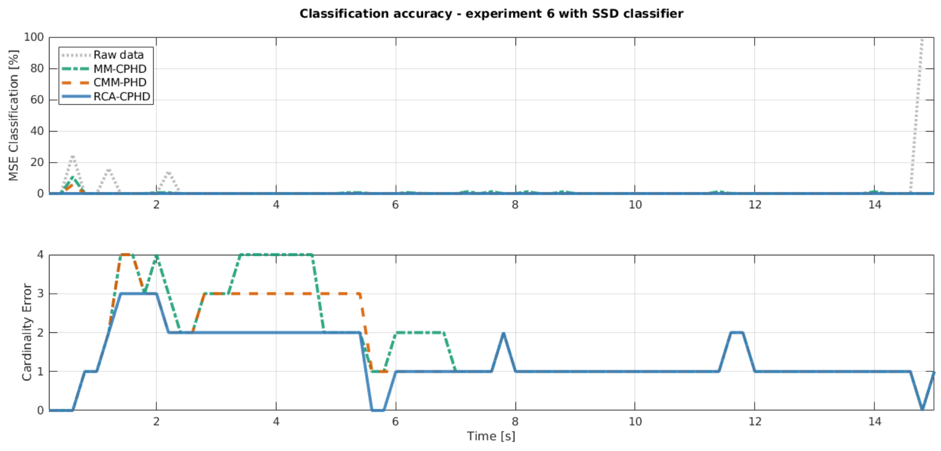


Figure 5.55: Classification accuracy for experiment 6 using SSD classifier

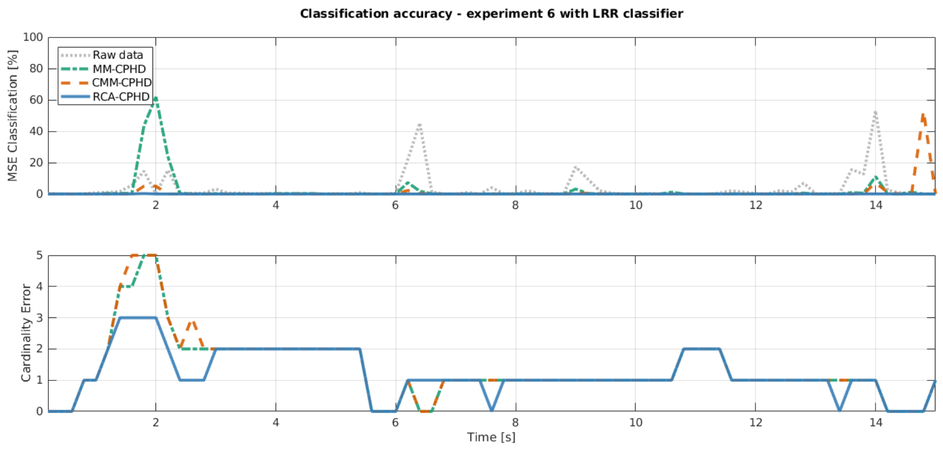


Figure 5.56: Classification accuracy for experiment 6 using LRR classifier

Summary of results

The results of experiment 6 show that the RCA-CPHD has a higher tracking- and classification accuracy during the high classification uncertainty moments, e.g. occlusions, crossings and misclassifications, with respect to the MM-CPHD and CMM-CPHD. This is also seen in the average OSPA distances and average MSE over the entire recording as shown in table 5.12. The RCA-CPHD has a significantly lower OSPA distance and lower MSE for both classifiers with respect to the CMM-CPHD and MM-CPHD.

	Raw data	MM-CPHD	CMM-CPHD	**RCA-CPHD**
Total Average OSPA Distance LRR [m]	-	1.24	1.23	**1.09**
Total Average OSPA Distance SSD [m]	-	1.44	1.34	**1.24**
Total Average MSE LRR [%]	3.67	4.73	3.33	**0.01**
Total Average MSE SSD [%]	2.07	0.29	0.08	**0.00**

Table 5.12: The total average tracking and classification accuracy for experiment 6. In bold the lowest error is highlighted per performance measure.

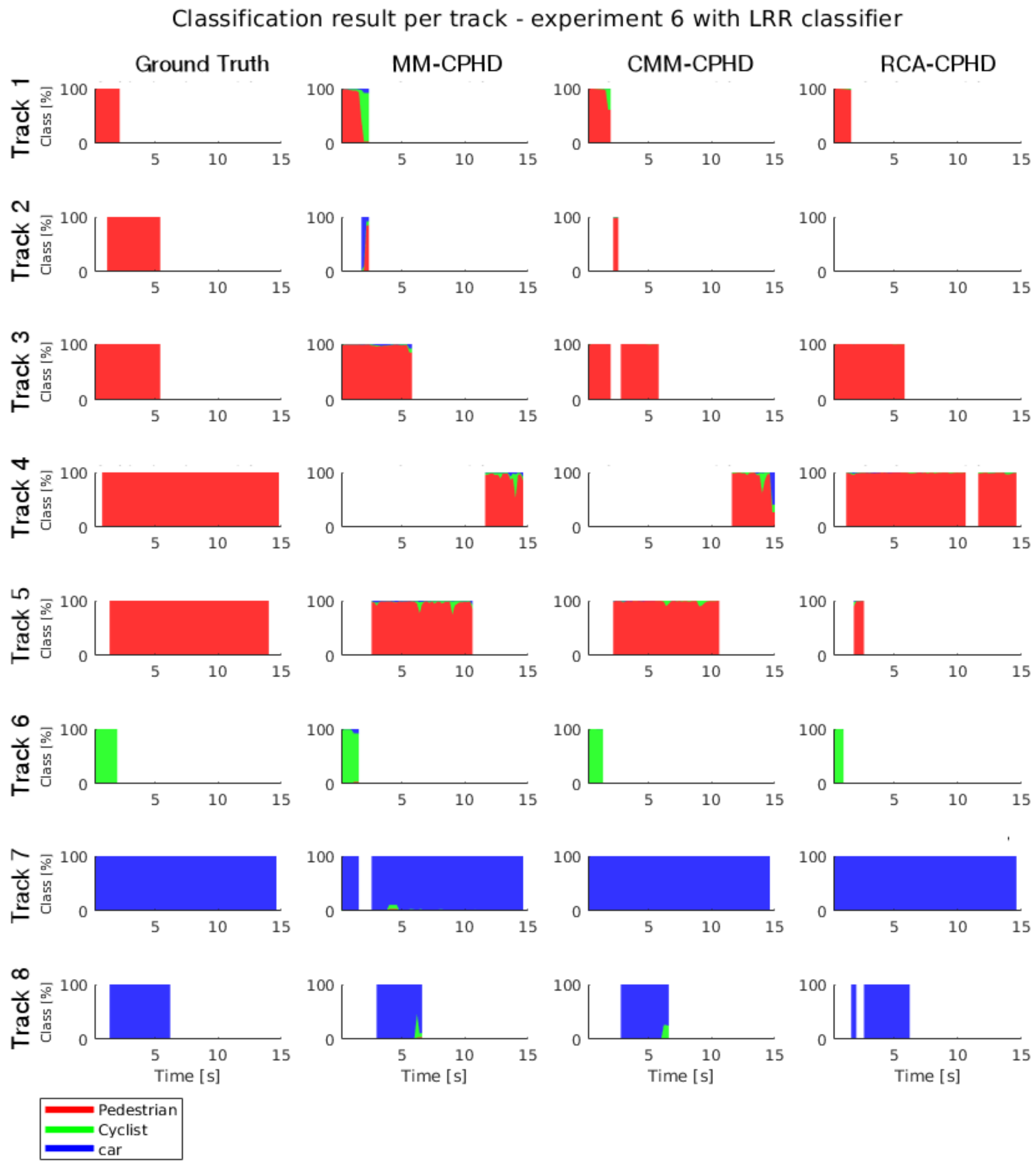


Figure 5.57: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

5.3.7 Validation Conclusion

The results of the validation show that for every individual experiment the RCA-CPHD has a significant higher tracking accuracy compared to the MM-CPHD and CMM-PHD during classification uncertain moments, like occlusions, crossings and misclassifications:

	experiment 1	Experiment 2	Experiment 3	Experiment 4	Experiment 5	Experiment 6
RCA-CPHD	**1.22 m**	**1.08 m**	**1.39 m**	**1.31 m**	**0.97 m**	**1.09 m**
CMM-CPHD	1.67 m	1.29 m	1.59 m	1.55 m	1.21 m	1.23 m
MM-CPHD	1.80 m	1.30 m	1.46 m	1.45 m	1.14 m	1.24 m

Table 5.13: Summary of OSPA error for all experiments with LRR classifier. The lowest OSPA error is highlighted in bold.

	experiment 1	Experiment 2	Experiment 3	Experiment 4	Experiment 5	Experiment 6
RCA-CPHD	**1.21 m**	**1.78 m**	**1.36 m**	**1.46 m**	**1.44 m**	**1.24 m**
CMM-CPHD	1.43 m	2.10 m	1.64 m	1.62 m	1.64 m	1.34 m
MM-CPHD	1.71 m	2.11 m	1.56 m	1.59 m	1.72 m	1.44 m

Table 5.14: Summary of OSPA error for all experiments with SSD classifier. The lowest OSPA error is highlighted in bold.

In addition, the results also show that for every individual experiment, the RCA-CPHD has a significant higher classification accuracy during the classification uncertain:

	experiment 1	Experiment 2	Experiment 3	Experiment 4	Experiment 5	Experiment 6
RCA-CPHD	**0.40 %**	**0.09 %**	**0.35 %**	**0.08 %**	**0.56 %**	**0.01 %**
CMM-CPHD	0.62 %	0.33 %	3.19 %	3.61 %	10.11 %	3.33 %
MM-CPHD	1.94 %	4.54 %	1.81 %	5.56 %	5.14 %	4.73 %

Table 5.15: Summary of classification MSE for all experiments with LRR classifier. The lowest classification MSE is highlighted in bold.

	experiment 1	Experiment 2	Experiment 3	Experiment 4	Experiment 5	Experiment 6
RCA-CPHD	**0.00 %**	**0.45 %**	**0.00 %**	**0.00 %**	**0.13 %**	**0.00 %**
CMM-CPHD	0.72 %	9.17 %	1.18 %	7.36 %	8.72 %	0.08 %
MM-CPHD	7.21 %	9.50 %	2.81 %	7.08 %	7.19 %	0.29 %

Table 5.16: Summary of classification MSE for all experiments with SSD classifier. The lowest classification MSE is highlighted in bold.

6 Concluding Remarks

6.1 Conclusion

Autonomous driving could be a unique opportunity to increase these traffic safety, traffic flow efficiency and to reduce emissions in future [30]. In order to operate reliably and accurately, autonomous driving vehicles and autonomous features require an accurate perception of the infrastructure and other road users in the surrounding. Multi-object tracking is the process concerned with the estimation of the states of the objects in the environment, given the noisy measurements from the sensors. In addition to the estimation of the states, it is also necessary to estimate the classification of the objects e.g. for a correct situation analysis and path planning. Classifiers are used to detect and classify objects from image frames, however the classification is sometimes incorrect or uncertain. This results in a decrease of tracking accuracy and classification accuracy in classification uncertain conditions. The contribution in this work is, by keeping the classification uncertainty in the tracking approach and using it in all steps, to jointly improve the tracking accuracy as well as the classification.

The main challenge in multi-object tracking lies in the correct association of the measurements with the corresponding tracks of the objects. This association is challenging due to undetected or false alarms of objects by the sensors, closely packed or crossing objects and dense clutter measurements. Also objects appear and disappear continuously from the vehicle's field of view in traffic situations. This creates an uncertainty about the number of objects, making the measurement-to-track association even harder. Most traditional multi-object tracking approaches are focused on the explicit association of the measurements and tracks. Due to the explicit measurement-to-track association, these approaches are computational intensive in general [60] [74]. Mahler proposed the Probability Hypothesis Density (PHD) [47], an object tracking approach based on Random Finite Sets (RFS), which explicitly avoids the association of measurements. The PHD filter is computational tractable due to its propagation of the first order approximation of the Multi Object Bayes recursion. However the PHD filter assumes a Poisson distribution in the number of targets, resulting in a high variance in the cardinality when the number of objects increases. The Cardinalized PHD (CPHD) solves the instability in the number of objects by simultaneously propagation the probability hypothesis density and the cardinality distribution. The more stable estimation comes however at the cost of a higher computational complexity.

Classifiers are used to detect and classify objects from image frames. This classification is used for a correct situation analysis and path planning, the next steps after multi-object tracking. The classification is also used in the MM-PHD proposed by Mahler et al. [45] and CMM-PHD proposed by Meissner et al. [50] to improve the prediction step of the PHD filter, using class-specific motion models. The classification results from the classifiers are, however, sometimes incorrect. In that case the incorrect motion model is selected in the prediction step, resulting in a less accurate tracking. In addition, in situations with closely packed or crossing objects, tracks of unequal class can be mixed up, causing the classification of the object to change incorrectly. So during classification uncertain conditions and the possibility of updating tracks with measurements of unequal class result in a reduction of the tracking accuracy and the classification accuracy.

In this thesis a more robust multi-sensor multi-object tracking approach for dealing with classification uncertainty, called the Robust Classification Aided CPHD (RCA-CPHD), is proposed. Instead of using binary classifications, the RCA-CPHD is making use of the classification probabilities in the filter. In the MM-CPHD, the classification is only used in the prediction step of the filter. In the RCA-CPHD, however, the class-probabilities are used in all steps of the filter. The prediction is done using class-specific motion models based on the classification uncertainty, resulting in a more accurate prediction. In addition, the likelihood of measurements and tracks is not only based on localisation, but also on similarity in classification in the update step. A track with a certain class is therefore not influenced by a measurement of another class, preventing a mix up of tracks e.g. during crossing objects. Also the gating process is also based on likelihood in classification, instead of localisation only. New appearing objects close to another object, but with a different classification, are therefore faster initiated.

The performance of the proposed RCA-CPHD, defined by the tracking accuracy and the classification accuracy, is evaluated in a two step approach; verification and validation. First a verification is done by comparing the performance of the RCA-CPHD with the MM-CPHD and the CMM-CPHD on simulated data from one radar and one camera. Two different scenarios are simulated and the performance is determined by averaging over 100 Monte Carlo runs. Subsequently the performance is validated by comparing the performance of the RCA-CPHD with the original MM-CPHD and the CMM-PHD on six real world data experiments from a stereo camera and the two different classifiers. Two experiments are recorded with a stereo camera on the DAVI-vehicle and four experiments are selected from the KITTI database [22]. The validation results show that the RCA-CPHD outperforms the MM-CPHD and CMM-PHD in both tracking accuracy and classification accuracy for all six experiments.

In addition, the verification and the validation of the performance is done for two different classifiers. The two classifiers are the LRR classifier developed by Ghiasi et al. [25] and the SSD developed by Wei Liu et al. [36]. The two classifiers differ in the way of dealing with classification uncertainty. Both the code of the LRR from Ghiasi et al. [25] and the code of the SSD from Wei Liu et al. [36] are adjusted to output the class probabilities for the three considered classes. The RCA-CPHD outperforms the MM-CPHD and CMM-PHD in both tracking accuracy and classification accuracy during classification uncertain conditions for both classifiers in the simulations and the real data scenarios.

6.2 Discussion

For the evaluation of the RCA-CPHD, the output of the classifier has been adjusted to output the probability of the observed classes only and the other considered classes are neglected. The class probabilities contain therefore not the actual uncertainty of the classifier, but given a detection, the corresponding probabilities of being a pedestrian, cyclist and car. This creates the possibility of an object to be classified high for an observed class, while it was actually lower classified due to an uncertainty in classification with non-observed classes. This however does not affect the performance of the compared filters in classification uncertain conditions.

Besides the RCA-CPHD is evaluated on urban traffic situations. The performance evaluation is therefore particularly valid for urban environments. For a more complete overview of the performance in automotive application, the evaluation should be extended with driving tests on the highway.

6.3 Future Work

In the evaluation of the RCA-CPHD three different types of road users are considered, namely pedestrians, cyclists and cars. In future work this could be extended with more road users like trucks, motorcycles, mopeds, etc. and their corresponding motion models. The use of more models in the RCA-CPHD would result in an increase in computational load, since for each class a duplicate is created. However since the RCA-CPHD uses uncertainties, instead of a transition matrix with jump probabilities as used in the MM-CPHD and CMM-PHD, it is no longer needed to predict and propagate the very unlikely classes.

For example if an object is classified as 40% pedestrian, 60% cyclist and 0% car, it is not necessary to propagate a duplicate for the car. If only the duplicates are propagated from the uncertain classes, an advantage will be gained in computational load.

In addition, the RCA-CPHD uses a constant detection probability based on the sensor modalities. When using a classifier, the detection probability is, however, different per class. For example pedestrians and cars often have a higher detection probability than cyclists. Since the class-probabilities are propagated, a unique detection probability per GM component could be composed based on the corresponding class probabilities.

Information about the classification and corresponding uncertainty has been obtained from measurement features from the sensors. However kinematics features of the estimated track could also give information about the classification. For example very recently Fortin et al. [19] proposed the Credal Classification MM-PHD filter which uses kinematic behaviour for classification and Meissner et al. [50] uses the maximum speed of an object as information about the classification. The use of kinematic features in the RCA-CPHD in addition to the measurement features would therefore be interesting for future research.

The implementation of the RCA-CPHD is evaluation with one radar and one camera sensor. In order to use the RCA-CPHD for autonomous driving, the implementation should be extended with multiple of these sensor couples covering the entire surrounding of the vehicle. Besides the evaluation is done in urban environment. In future research, driving tests on the highway would be interesting to obtain a more complete overview of the performance.

Bibliography

[1] ANP. Autonomous toyota prius developed by davi. `https://www.anpfoto.nl/`, 2016. [Online; accessed Jun 06, 2016].

[2] Yaakov Bar-Shalom. Multitarget-multisensor tracking: advanced applications. *Norwood, MA, Artech House, 1990, 391 p.*, 1, 1990.

[3] Yaakov Bar-Shalom and Edison Tse. Tracking in a cluttered environment with probabilistic data association. *Automatica*, 11(5):451–460, 1975.

[4] James O Berger. *Statistical decision theory and Bayesian analysis*. Springer Science & Business Media, 2013.

[5] W Dale Blair, Theodore R Rice, Ali T Alouani, and P Xia. Asynchronous data fusion for target tracking with a multitasking radar and optical sensor. In *Orlando'91, Orlando, FL*, pages 234–245. International Society for Optics and Photonics, 1991.

[6] Glenn W Brier. Verification of forecasts expressed in terms of probability. *Monthey Weather Review*, 78(1):1–3, 1950.

[7] Daniel E Clark, Kusha Panta, and Ba-Ngu Vo. The gm-phd filter multiple target tracker. In *information Fusion, 2006 9th International Conference on*, pages 1–8. IEEE, 2006.

[8] SAE On-Road Automated Vehicle Standards Committee et al. Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems. *SAE Standard J3016*, pages 01–16, 2014.

[9] Continental. *Ars 30x datasheet*, 5 2012. Version 09.

[10] Verband der Automobilindustrie eV. Automation: From driver assistance systems to automated driving. *VDA Magazine-Automation*, 2015.

[11] Hugh Durrant-Whyte. Multi sensor data fusion. *The Australian Center for Field Robotics, The University of Sydney, Sydney, Australia*, 2001.

[12] Hugh Durrant-Whyte and Thomas C Henderson. Multisensor data fusion. In *Springer Handbook of Robotics*, pages 585–610. Springer, 2008.

[13] Ozgur Erdinc, Peter Willett, and Yaakov Bar-Shalom. Probability hypothesis density filter for multitarget multisensor tracking. In *Information Fusion, 2005 8th International Conference on*, volume 1, pages 8–pp. IEEE, 2005.

[14] Ozgur Erdinc, Peter Willett, and Yaakov Bar-Shalom. A physical-space approach for the probability hypothesis density and cardinalized probability hypothesis density filters. In *Signal and Data Processing of Small Targets 2006*, volume 6236, page 623619. International Society for Optics and Photonics, 2006.

[15] Ozgur Erdinc, Peter Willett, and Yaakov Bar-Shalom. The bin-occupancy filter and its connection to the phd filters. *IEEE Transactions on Signal Processing*, 57:4232–4246, 2009.

[16] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[17] César Ferri, José Hernández-Orallo, and R Modroiu. An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, 30(1):27–38, 2009.

[18] Mathieu Flament, Gilles Fleury, and Marie-Eve Davoust. Particle filter and gaussian-mixture filter efficiency evaluation for terrain-aided navigation. In *2004 12th European Signal Processing Conference*, 2004.

[19] Benoît Fortin, Samir Hachour, and François Delmotte. Multi-target phd tracking and classification using imprecise likelihoods. *International Journal of Approximate Reasoning*, 90:17–36, 2017.

[20] Michael Gabb, Otto Lohlein, Raimar Wagner, Antje Westenberger, Martin Fritzsche, and Klaus Dietmayer. High-performance on-road vehicle detection in monocular images. In *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*, pages 336–341. IEEE, 2013.

[21] Fernando Garcia, Antonio Prioletti, Pietro Cerri, Alberto Broggi, Arturo de la Escalera, and Jose M Armingol. Visual feature tracking based on phd filter for vehicle detection. In *Information Fusion (FUSION), 2014 17th International Conference on*, pages 1–6. IEEE, 2014.

[22] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3354–3361. IEEE, 2012.

[23] Ramona Georgescu and Peter Willett. Classification aided cardinalized probability hypothesis density filter. *SPIE Signal Processing, Sensor Fusion, and Target Recognition XXI*, 2012.

[24] Ramona Georgescu and Peter Willett. The multiple model cphd tracker. *IEEE Transactions on Signal Processing*, 60(4):1741–1751, 2012.

[25] Golnaz Ghiasi and Charless C Fowlkes. Laplacian pyramid reconstruction and refinement for semantic segmentation. In *European Conference on Computer Vision*, pages 519–534. Springer, 2016.

[26] Afzal Godil, Roger Eastman, and T Hong. Ground truth systems for object recognition and tracking. Technical report, 2013.

[27] Irwin R Goodman, Ronald P Mahler, and Hung T Nguyen. *Mathematics of data fusion*, volume 37. Springer Science & Business Media, 2013.

[28] Karl Granström and Christian Lundquist. On the use of multiple measurement models for extended target tracking. In *Information Fusion (FUSION), 2013 16th International Conference on*, pages 1534–1541. IEEE, 2013.

[29] G Homem De Almeida Rodriguez Correia. Wepods project: finding the potential for automated road public transport. *Fly over, 25 (2), 2016*, 2016.

[30] Raymond Hoogendoorn, Bart van Arem, Riender Happee, Manuel Mazo Espinoza, and Dimitrios Kotiadis. Towards safe and efficient driving through vehicle automation: The dutch automated vehicle initiative, 2013.

[31] Tao Jiang, Srdjan Petrovic, Uma Ayyer, Anand Tolani, and Sajid Husain. Self-driving cars: Disruptive or incremental? *Applied Innovation Review (AIR)*, Issue 1:3–22, 2015.

[32] Laetitia Lamard, Roland Chapuis, and Jean-Philippe Boyer. A comparison of two different tracking algorithms is provided for real application. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 414–419. IEEE, 2012.

[33] Laetitia Lamard, Roland Chapuis, and Jean Philippe Boyer. Multi target tracking with cphd filter based on asynchronous sensors. In *Information Fusion (FUSION), 2013 16th International Conference on*, pages 892–898. IEEE, 2013.

[34] M langer. Lecture notes in fundamentals of computer vision, Fall 2010.

[35] Stéphanie Lefèvre, Dizan Vasquez, and Christian Laugier. A survey on motion prediction and risk assessment for intelligent vehicles. *Robomech Journal*, 1(1):1, 2014.

[36] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[37] Malin Lundgren, Lennart Svensson, and Lars Hammarstrand. A cphd filter for tracking with spawning models. *IEEE Journal of Selected Topics in Signal Processing*, 7(3):496–507, 2013.

[38] Christian Lundquist. Sensor fusion for automotive applications. *Linköping University Electronic Press*, 2011.

[39] Wing-Kin Ma, Ba-Ngu Vo, Sumeetpal S Singh, and Adrian Baddeley. Tracking an unknown time-varying number of speakers using tdoa measurements: a random finite set approach. *Signal Processing, IEEE Transactions on*, 54(9):3291–3304, 2006.

[40] Mirko Maehlisch, Roland Schweiger, Werner Ritter, and Klaus Dietmayer. Multisensor vehicle tracking with the probability hypothesis density filter. In *Information Fusion, 2006 9th International Conference on*, pages 1–8. IEEE, 2006.

[41] Ronald Mahler. Engineering statistics for multi-object tracking. In *Multi-Object Tracking, 2001. Proceedings. 2001 IEEE Workshop on*, pages 53–60. IEEE, 2001.

[42] Ronald Mahler. Random set theory for target tracking and identification, 2001.

[43] Ronald Mahler. A theory of phd filters of higher order in target number. In *Signal Processing, Sensor Fusion, and Target Recognition XV*, volume 6235, page 62350K. International Society for Optics and Photonics, 2006.

[44] Ronald Mahler. Phd filters of higher order in target number. *IEEE Transactions on Aerospace and Electronic Systems*, 43(4):1523–1543, 2007.

[45] Ronald Mahler. On multitarget jump-markov filters. In *Information Fusion (FUSION), 2012 15th International Conference on*, pages 149–156. IEEE, 2012.

[46] Ronald Mahler. "statistics 102" for multisource-multitarget detection and tracking. *Selected Topics in Signal Processing, IEEE Journal of*, 7(3):376–389, 2013.

[47] Ronald PS Mahler. Multitarget bayes filtering via first-order multitarget moments. *Aerospace and Electronic Systems, IEEE Transactions on*, 39(4):1152–1178, 2003.

[48] Ronald PS Mahler. "statistics 101" for multisensor, multitarget data fusion. *Aerospace and Electronic Systems Magazine, IEEE*, 19(1):53–64, 2004.

[49] Daniel Meissner, Stephan Reuter, and Klaus Dietmayer. Road user tracking at intersections using a multiple-model phd filter. In *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pages 377–382. IEEE, 2013.

[50] Daniel Meissner, Stephan Reuter, Elias Strigel, and Klaus Dietmayer. Intersection-based road user tracking using a classifying multiple-model phd filter. *IEEE Intelligent Transportation Systems Magazine*, 6(2):21–33, 2014.

[51] Daniel Meissner, Stephan Reuter, Benjamin Wilking, and Klaus Dietmayer. Road user tracking using a dempster-shafer based classifying multiple-model phd filter. In *Information Fusion (FUSION), 2013 16th International Conference on*, pages 1236–1242. IEEE, 2013.

[52] Daniel Alexander Meißner. *Intersection-based road user tracking using a classifying multiple-model PHD filter*. PhD thesis, Universität Ulm, 2016.

[53] Michael Munz, Mirko Mahlisch, and Klaus Dietmayer. Generic centralized multi sensor data fusion based on probabilistic sensor and environment models for driver assistance systems. *IEEE Intelligent Transportation Systems Magazine*, 2(1):6–17, 2010.

[54] P Muthumanikandan, S Vasuhi, and V Vaidehi. Multiple maneuvering target tracking using mht and nonlinear non-gaussian kalman filter. In *Signal Processing, Communications and Networking, 2008. ICSCN'08. International Conference on*, pages 52–56. IEEE, 2008.

[55] Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 625–632. ACM, 2005.

[56] World Health Organization. *Global status report on road safety 2015*. World Health Organization, 2015.

[57] Umit Ozguner, Tankut Acarman, and Keith Redmill. *Autonomous ground vehicles*. Artech House, 2011.

[58] Kusha Panta, Daniel E Clark, and Ba-Ngu Vo. Data association and track management for the gaussian mixture probability hypothesis density filter. *IEEE Transactions on Aerospace and Electronic Systems*, 45(3), 2009.

[59] Kusha Panta, Ba-Ngu Vo, Sumeetpal Singh, and Arnaud Doucet. Probability hypothesis density filter versus multiple hypothesis tracking. In *Defense and Security*, pages 284–295. International Society for Optics and Photonics, 2004.

[60] Syed Ahmed Pasha, Ba-Ngu Vo, Hoang Duong Tuan, and Wing-Kin Ma. A gaussian mixture phd filter for jump markov system models. *IEEE Transactions on Aerospace and Electronic Systems*, 45(3), 2009.

[61] John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[62] GW Pulford. Taxonomy of multiple target tracking methods. In *Radar, Sonar and Navigation, IEE Proceedings-*, volume 152, pages 291–304. IET, 2005.

[63] Changzhen Qiu, Zhiyong Zhang, Huanzhang Lu, and Huiwu Luo. A survey of motion-based multitarget tracking methods. *Progress In Electromagnetics Research B*, 62:195–223, 2015.

[64] Donald Reid. An algorithm for tracking multiple targets. *IEEE transactions on Automatic Control*, 24(6):843–854, 1979.

[65] Stephan Reuter. *Multi-object tracking using random finite sets*. PhD thesis, Universität Ulm. Fakultät für Ingenieurwissenschaften und Informatik, 2014.

[66] Stephan Reuter and Klaus Dietmayer. Pedestrian tracking using random finite sets. In *Information Fusion (FUSION), 2011 Proceedings of the 14th International Conference on*, pages 1–8. IEEE, 2011.

[67] Stephan Reuter, Klaus Dietmayer, and Sebastian Handrich. Real-time implementation of a random finite set particle filter. *Sensor Data Fusion: Trends, Solutions, Applications (SDF), Lecture Notes in Informatics*, 192, 2011.

[68] Stephan Reuter, Benjamin Wilking, Jurgen Wiest, Michael Munz, and Klaus Dietmayer. Real-time multi-object tracking using random finite sets. *IEEE Transactions on Aerospace and Electronic Systems*, 49(4):2666–2678, 2013.

[69] Rijksoverheid. Road traffic: volume trends and environmental pressure,1990-2016, 2017.

[70] Dominic Schuhmacher, Ba-Tuong Vo, and Ba-Ngu Vo. A consistent metric for performance evaluation of multi-object filters. *IEEE transactions on signal processing*, 56(8):3447–3457, 2008.

[71] Milos Vasic and Alcherio Martinoli. A collaborative sensor fusion algorithm for multi-object tracking using a gaussian mixture probability hypothesis density filter. In *Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on*, pages 491–498. IEEE, 2015.

[72] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2001.

[73] B.-T. Vo, B.-N. Vo, and A. Cantoni. Analytic implementations of the cardinalized probability hypothesis density filter. *IEEE Transactions on Signal Processing*, 55(7):3553–3567, July 2007.

[74] Ba-Ngu Vo and Wing-Kin Ma. The gaussian mixture probability hypothesis density filter. *Signal Processing, IEEE Transactions on*, 54(11):4091–4104, 2006.

[75] Ba-Ngu Vo, Sumeetpal Singh, and Arnaud Doucet. Sequential monte carlo implementation of the phd filter for multi-target tracking. In *Proc. Int. Conf. on Information Fusion*, pages 792–799, 2003.

[76] Ba-Ngu Vo, Sumeetpal Singh, and Arnaud Doucet. Sequential monte carlo methods for multitarget filtering with random finite sets. *Aerospace and Electronic Systems, IEEE Transactions on*, 41(4):1224–1245, 2005.

[77] Ba-Tuong Vo, Ba-Ngu Vo, and Antonio Cantoni. The cardinalized probability hypothesis density filter for linear gaussian multi-target models. In *Information sciences and systems, 2006 40th annual conference on*, pages 681–686. IEEE, 2006.

[78] Ba-Tuong Vo, Ba-Ngu Vo, and Antonio Cantoni. Analytic implementations of the cardinalized probability hypothesis density filter. *IEEE Transactions on Signal Processing*, 55(7):3553–3567, 2007.

[79] Carl Vondrick, Donald Patterson, and Deva Ramanan. Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision*, pages 1–21. 10.1007/s11263-012-0564-1.

[80] Daniel Weimer, Sebastian Köhler, Christian Hellert, Konrad Doll, Ulrich Brunsmann, and Roland Krzikalla. Gpu architecture for stationary multisensor pedestrian detection at smart intersections. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 89–94. IEEE, 2011.

[81] WEPods.nl. Wepod vehicle developed by davi. `https://wepods.nl/pages/media`, 2016. [Online; accessed Jun 06, 2016].

[82] Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 694–699. ACM, 2002.

Appendix A Plots with classification results per track

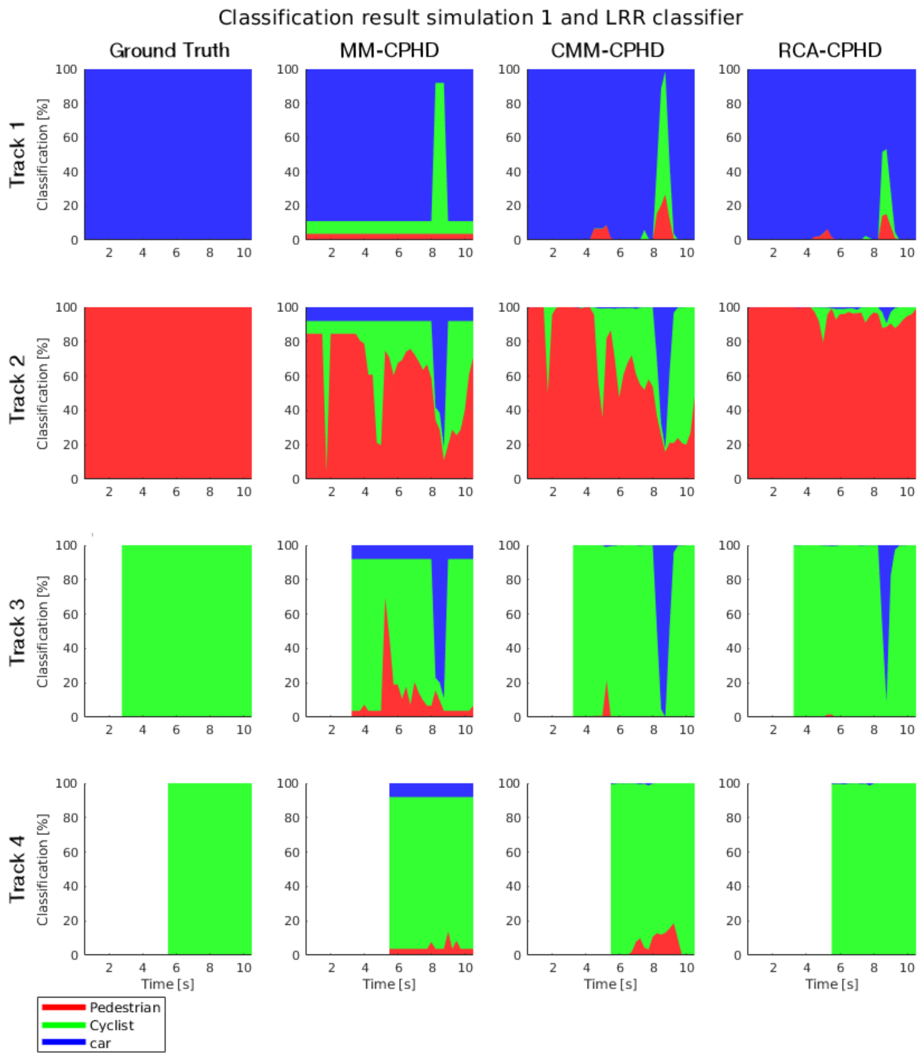


Figure A.1: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

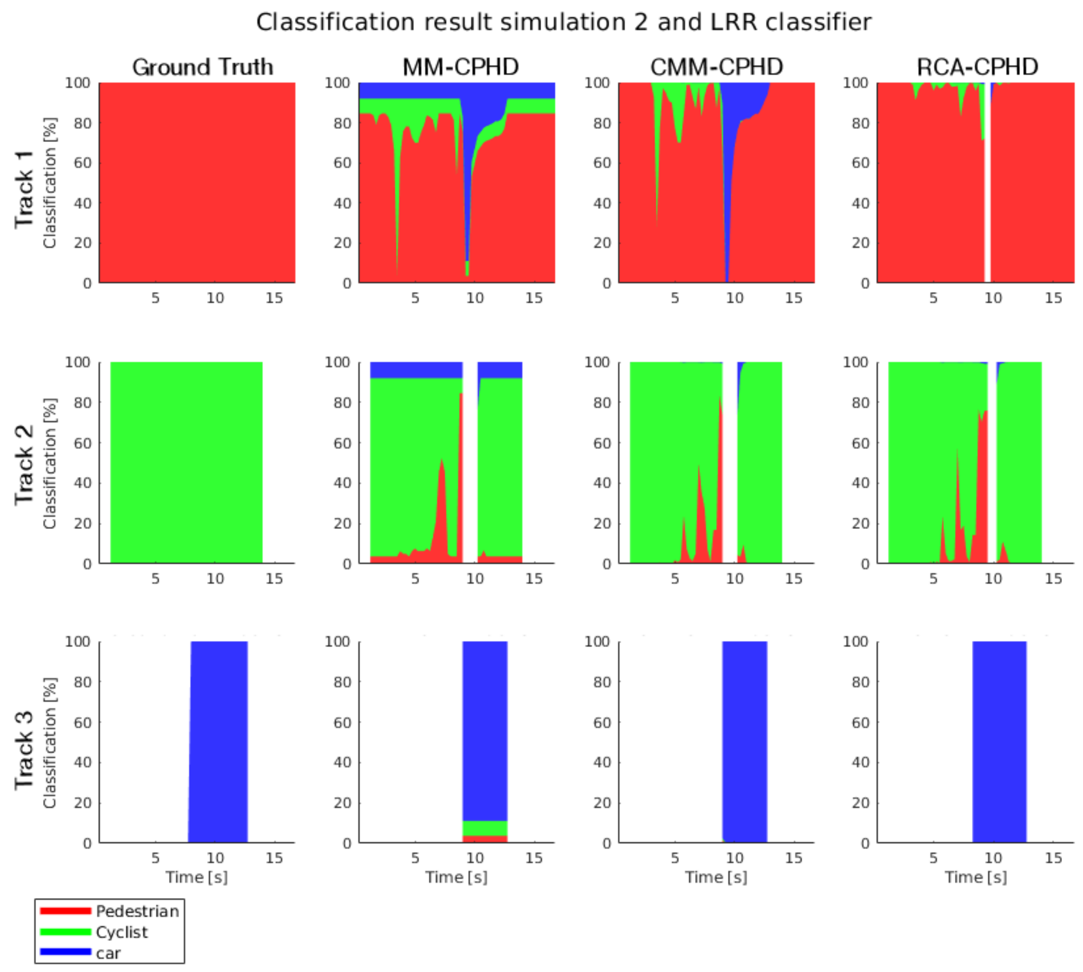


Figure A.2: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

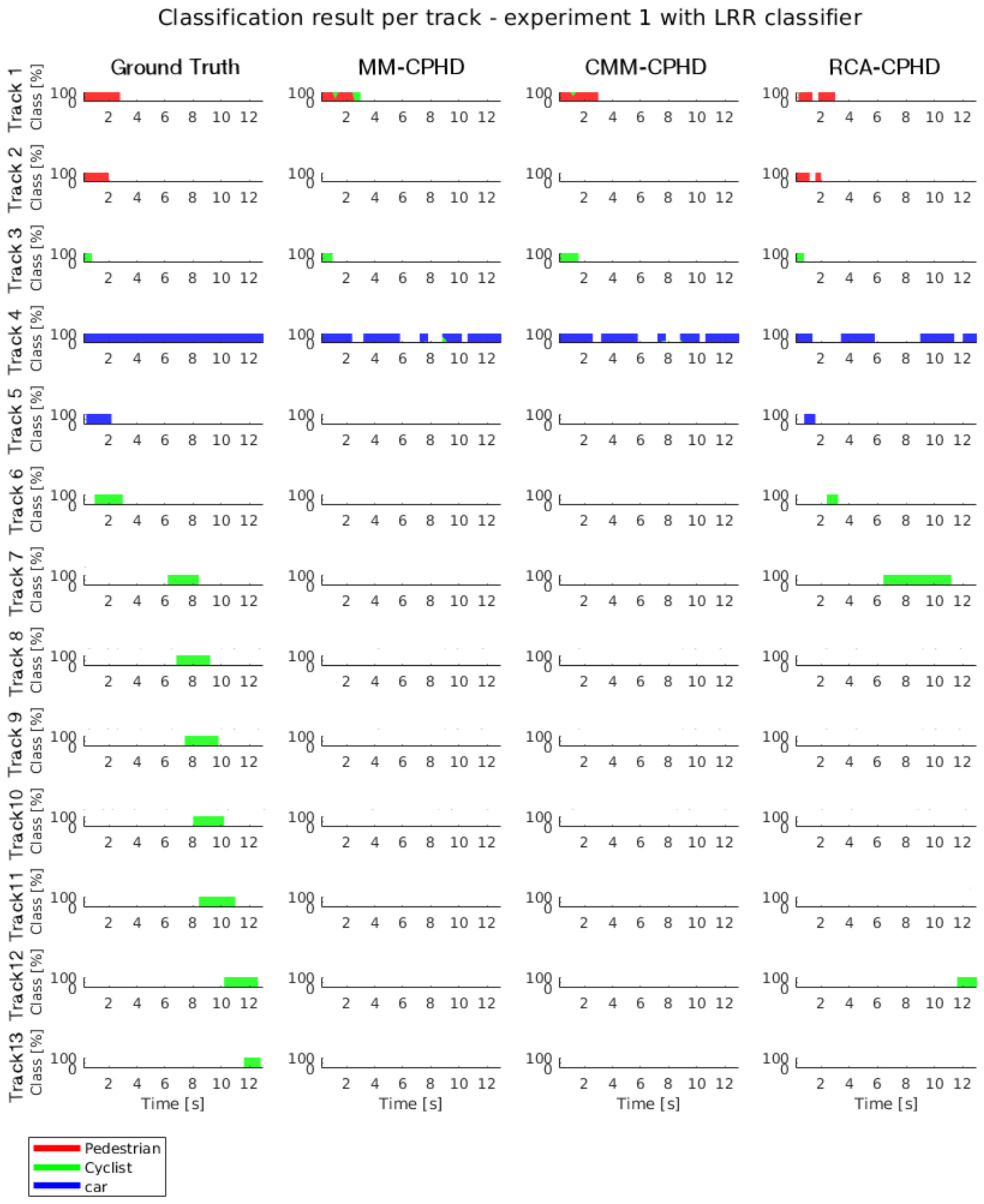


Figure A.3: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

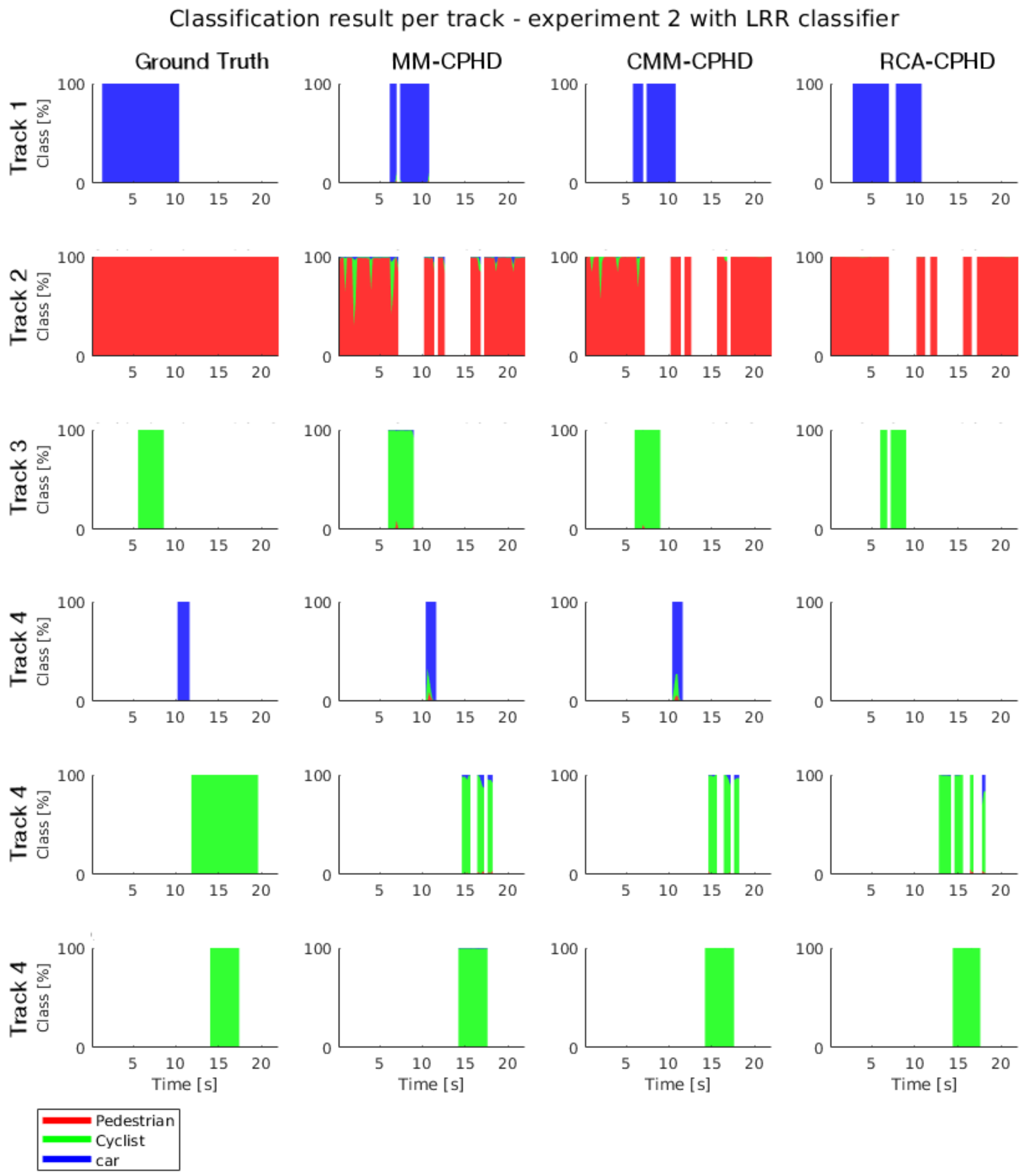


Figure A.4: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

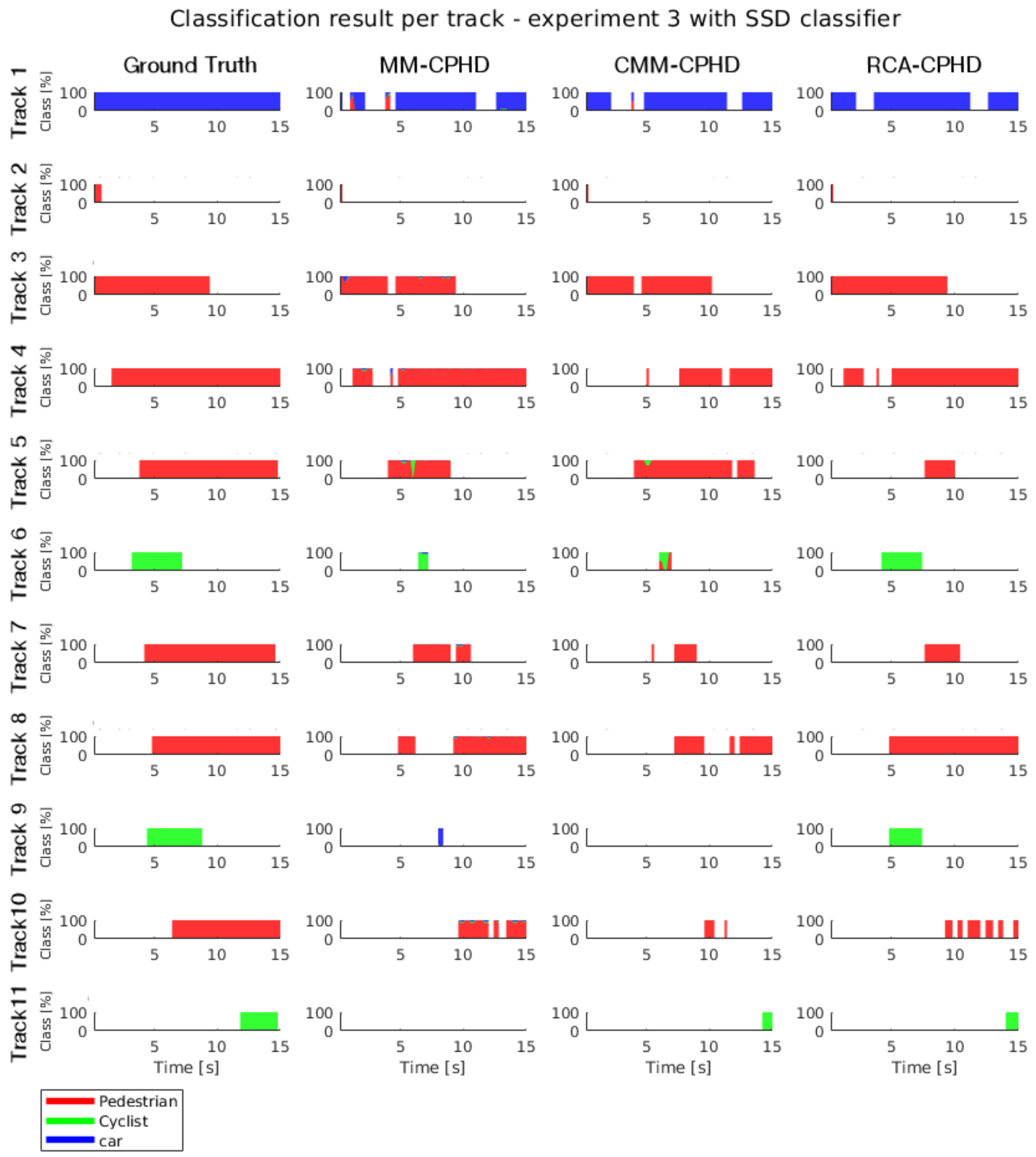


Figure A.5: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

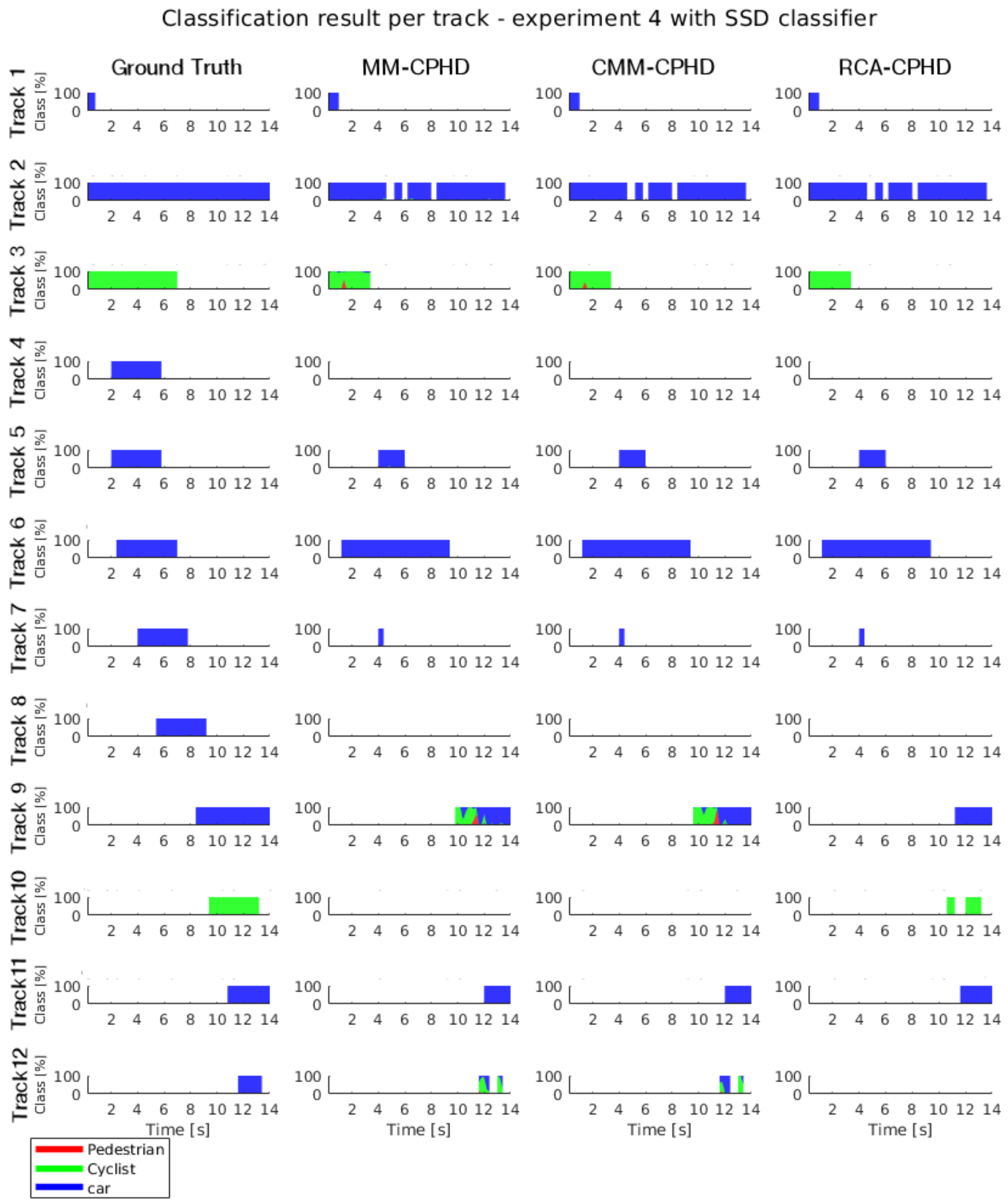


Figure A.6: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

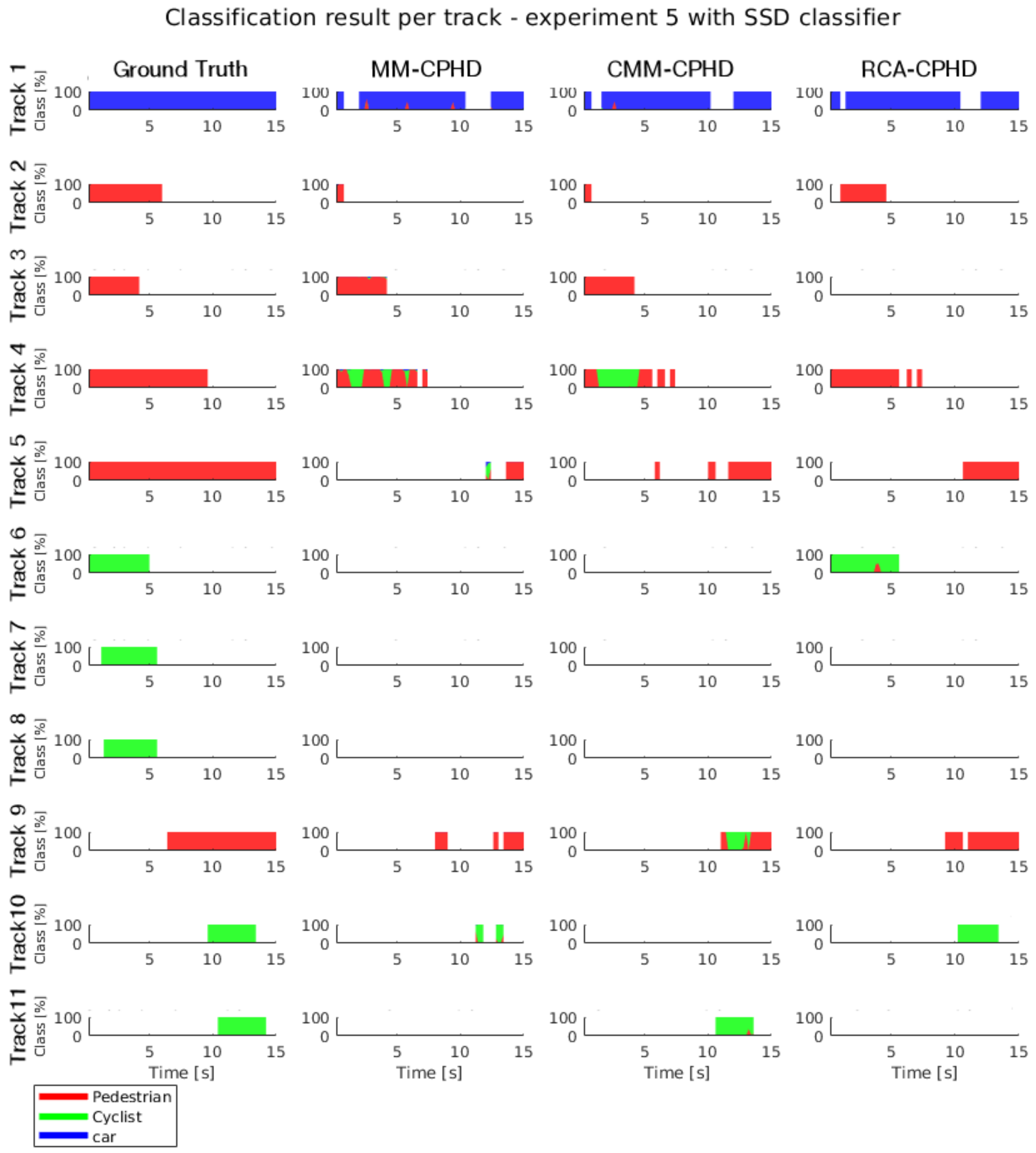


Figure A.7: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.

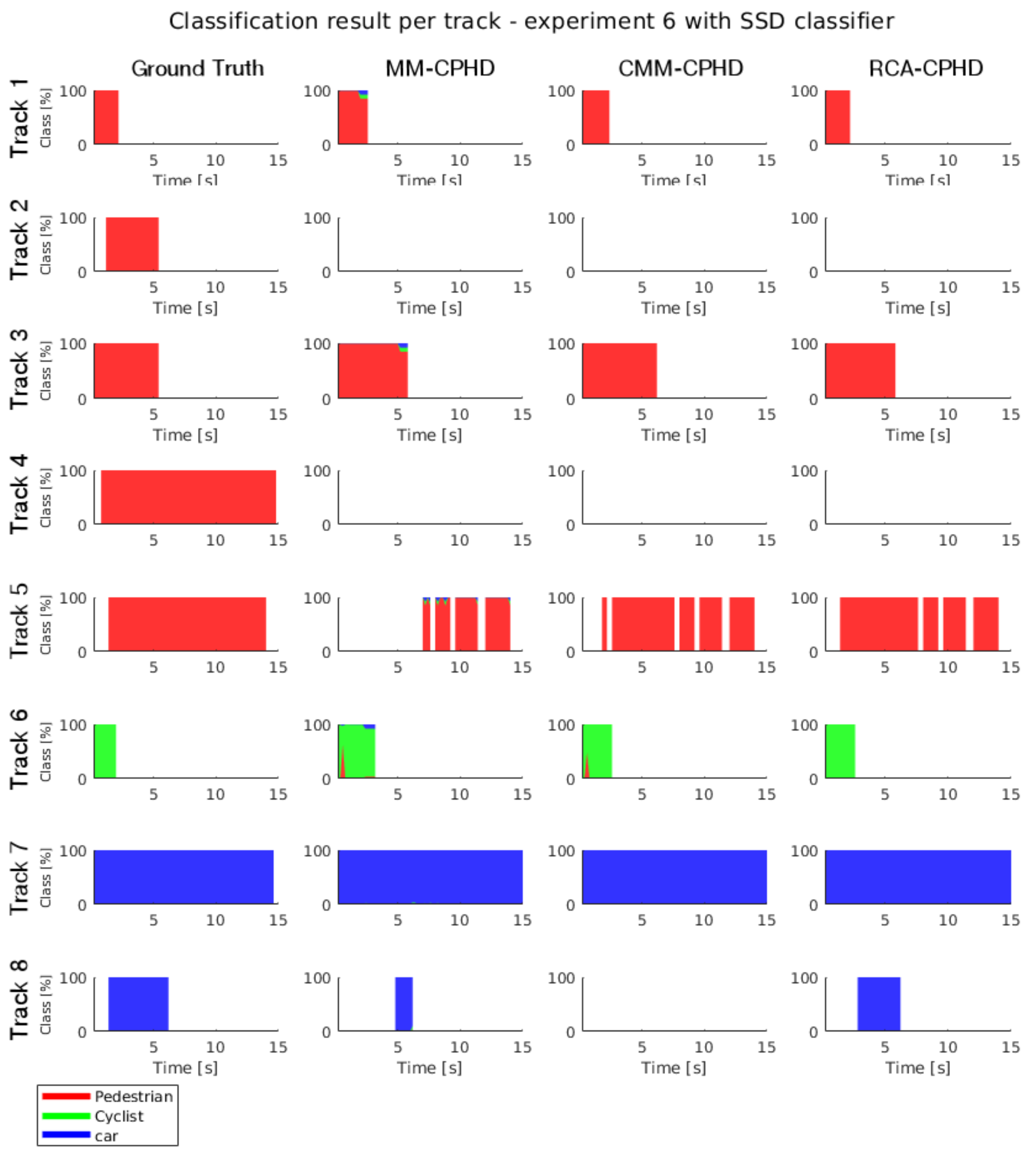


Figure A.8: Visualization of the classification result for the individual tracks per filter. In horizontal direction are the individual tracks shown and in vertical direction the result per filter, with the most left tracks being the ground truth.