Thesis for the degree of Doctor of Philosophy

Filtering and modelling for automotive safety systems

by

Fredrik Sandblom

CHALMERS

Department of Signals and Systems
Signal Processing Group
Chalmers University of Technology
Göteborg, Sweden 2011

Filtering and modelling for automotive safety systems
Fredrik Sandblom
ISBN 978-91-7385-608-9

This thesis has been prepared using LaTeXand BibTeX.

Doktorsavhandlingar vid
Chalmers tekniska högskola
Ny serie Nr 3289
ISSN 0346-718X

Department of Signals and Systems
Signal Processing Group
Chalmers University of Technology
SE-412 96 Göteborg, Sweden
Telephone: +46 (0)31 772 1000

Fredrik Sandblom
Telephone: +46 (0)31 322 6951
E-mail: fredrik.sandblom.2@volvo.com,
fredrik.sandblom@chalmers.se

Printed by Chalmers Reproservice
Göteborg, Sweden, November 2011

To Lina and Sixten

Abstract

This thesis makes five important contributions to the development of an automotive safety system: filtering algorithms, three modelling frameworks concerning the usage of radar detections in tracking, vehicle motion, and decision-making for intervention decisions, and finally the implementation architecture. In the filtering context, we have developed a new sigma-point method for estimating the moments of a transformed Gaussian random variable. These estimates are derived from analytical expressions and are based on evaluations of the transforming function. The method is applied to the moment estimation task in a Gaussian filter and the resulting algorithm is denoted the marginalised Kalman filter (MKF).

Compared to traditional radar models, ours is specifically designed for vehicle radars, which often yield several measurements from each object. These measurements can provide useful information, such as vehicle orientation, if they are accurately modelled. We introduce a tracking filter using such a sensor model, and show how the complex data association problem can be facilitated by merging similar hypotheses into groups.

The presented vehicle motion model includes the control input from the driver. Uncertainties regarding, e.g., driver style, are formally treated with increased prediction accuracy as a result. Similar to this model, the third framework also takes the driver into consideration by allowing interventions only when the driver is believed to accept them. Our evaluations indicate an increased benefit in collision avoidance systems — particularly in traffic situations where the future trajectory of another road user is hard for the driver to predict.

Finally, we present a modular functional design for implementing a real-time data fusion system. We conclude that a tracking system, using modern estimation techniques, is well suited for sensor data fusion in an automotive environment.

Acknowledgment

First, I would like to thank professor Mats Viberg for allowing me join his research group; a remarkably talented group of people that I have really enjoyed working with. Of these, my main supervisor and friend Lennart Svensson stands out as an excellent coach and researcher. He also convincingly argues for celebrating results before cross-checking them — because "that way there's more celebration". I thank you for your positive attitude and support throughout these years.

I am grateful to Volvo for providing me with the possibility to conduct this research. Andreas Johnsson, my manager at Volvo during the majority of my work, not only introduced me to the topic, but also supported me during my project. I look forward to working together with my fellow colleagues at Volvo.

As luck would have it, I was fortunate enough to start my research in the best possible way, together with three colleagues with sharp minds and a sense of humour, Lars Hammarstrand, Joakim Sörstedt and Lennart Svensson. It has been a delight working with you. Contrary to the majority of you, however, I chose to change my name *before* writing my thesis, which amuses me.

Over the last year, I have also had the pleasure of working together with Mattias Brännström in a fruitful collaboration which I sincerely hope we can continue. I have also worked together with Malin Lundgren for some time and — one day — our periods of parental leave are bound not to overlap. I look forward to that.

My sincere thanks to Lennart Svensson, Daniel Svensson and Mattias Brännström for proofreading this thesis, it would certainly not be the same without you.

I go back a long way with some of my friends, and you have meant a lot to me during my studies. I would like to particularly express my gratitude to the crayfish committee for annually bringing us together.

The support I have received from my family is priceless — thank you Mum, Dad, my brother Magnus, and my family on my wife's side. I would especially like to thank my lovely son Sixten, my pride and joy, for bringing me perspective, occasionally conjuring me to a cat — or sometimes a monster, and for making up the part of our family who likes to do improv with me.

My biggest gratitude, however, is towards Lina, my wonderful wife. I cannot express in words what you have meant for me since the day I fell in love with you!

Fredrik Sandblom, Göteborg 2011

List of publications

The following publications constitute the foundation for this thesis:

Paper I

F. Sandblom and L. Svensson. *Moment estimation using a marginalized transform.* Submitted to IEEE Transactions on Signal Processing.

Paper II

M. Brännström, F. Sandblom and L. Hammarstrand. *A probabilistic framework for decision-making in collision avoidance systems.* Submitted to IEEE Transactions on Intelligent Transportation Systems.

Paper III

L. Hammarstrand, F. Sandblom, L. Svensson, and J. Sörstedt. *Extended object tracking using a radar resolution model.* Accepted for publication in IEEE Transactions on Aerospace and Electronic Systems.

Paper IV

J. Sörstedt, L. Svensson, F. Sandblom, and L. Hammarstrand. *A new vehicle motion model for improved predictions and situation assessment.* Accepted for publication in IEEE Transactions on Intelligent Transportation Systems.

Paper V

F. Bengtsson[1] and L. Danielsson. *A design architecture for sensor data fusion systems with application to automotive safety.* In Proceedings of Intelligent Transport Systems World Congress, 2008, New York, November 2008.

[1] Fredrik changed his name from Bengtsson to Sandblom in 2009

Other publications by the author, omitted in the thesis:

[1] F. Sandblom and L. Svensson, "Marginalized sigma-point filtering," in *Proc. 14th Int. Conference on Information Fusion*, Chicago, USA, July 2011, pp. 1–8.

[2] F. Sandblom and M. Brännström, "Probabilistic threat assessment and driver modeling in collision avoidance systems," in *Intelligent Vehicles Symposium, 2011 IEEE*, Baden-Baden, Germany, June 2011, pp. 914–919.

[3] M. Ahrholdt, F. Sandblom, L. Danielsson, and C. Lundquist, "SEFS results on sensor data fusion system development," in *ITS World Congress*, Stockholm, Sweden, September 2009.

[4] F. Bengtsson and L. Danielsson, "Designing a real time sensor data fusion system with application to automotive safety," Tech. Rep. R007/2008, Signals and Systems, Chalmers University of Technology, Göteborg, April 2008.

[5] J. Gunnarsson, L. Svensson, L. Danielsson, and F. Bengtsson, "Tracking vehicles using radar detections," in *Intelligent Vehicles Symposium, 2007 IEEE*, Istanbul, Turkey, June 2007, pp. 296–302.

[6] J. Gunnarsson, L. Svensson, F. Bengtsson, and L. Danielsson, "Joint driver intention classification and tracking of vehicles," in *Nonlinear Statistical Signal Processing Workshop 2006*, Cambridge, UK, September 2006.

Public patent applications:

PCT/SE2008/000634 F. Sandblom, "Method and system for combining sensor data", WO/2010/053410.

PCT/SE2008/000631 G. Markkula and F. Sandblom, "Method and system for determining road data", WO/2010/053408.

Contents

Part I
Introduction

Chapter 1

Introduction

AUTOMOTIVE SAFETY is in focus more today than ever and it is likely that research efforts within this area will be intensified. Traffic-related injuries cause personal tragedies and require considerable resources from society for rehabilitation. Today, traffic-related accidents are the 10^{th} largest cause of death, accounting for 2.1% of all deaths. This cause is predicted to advance to 8^{th} place by the year 2030 [7], and taking into account the loss of health from disability, the result appears even worse [8]. Road traffic injuries are predicted to be the third leading contributor to the global burden of disease and injury by 2020.

One way to reduce the number of injuries is to equip vehicles with systems that help the driver avoid accidents, or to mitigate their effects. An early example is the anti-lock braking system (ABS), whereas recent innovations include systems that automatically apply the brakes to mitigate the effects of an imminent collision. Systems that assess the traffic situation, in order to determine whether or not to take action, are generally called *active safety systems*. A characteristic property of these systems is that their intended domain of action is when the driving situation becomes critical, but they should not act under other circumstances. Therefore, the systems must continuously monitor the traffic environment and assess whether an accident is likely to occur or not. Usage of a statistical framework in this process is easily motivated as the assessment involves uncertainties regarding the future; by the time it can be determined that an accident will happen for certain, it is already unavoidable.

The research presented in this thesis has been carried out as a part of the author's work with automotive safety at the Volvo Group together with the Chalmers University of Technology. The goal has been to develop methods to provide an accurate description of traffic situations, so that actions can be initiated in time to break the course of events leading to an accident. This thesis presents an overview of suitable methods, provides a theoretical background for understanding their usage, and describes how to fuse sensor data from multiple sensors in real time. An outline of the author's research contributions is provided in Chapter 5.

1.1 Thesis outline

This thesis has two parts. The first part introduces the general problem at hand and describes the conceptual approach towards solving it. The intention is for the reader to be properly prepared for the second part, where the contributions are included. A more detailed description of the introductory chapters is provided below, whereas the content of the appended papers and the author's contributions are discussed in Section 5.1.

Chapter 1: Introduction

The first chapter contains an introduction to truck-related accidents, and automotive safety systems are described in terms of their purpose, behaviour, and design challenges. The purpose is to clarify the motivations for the research presented in this thesis.

Chapter 2: Filtering

The tracking filter in an automotive safety system is responsible for refining sensor measurements to form an accurate description of the traffic situation, and this chapter aims at providing an adequate, but detailed, background of the topic. The problem formulation, in all its simplicity, is expressed in terms of the system model, and the optimal solution to the problem formulation is expressed on a recursive form. It is discussed under which circumstances the formal solution is applicable, and examples are given of both exact recursive solutions and approximate recursive methods. Finally, moment estimation using sigma-points is explained in terms of quadrature rules, and commonly used sigma-point filters are discussed.

Chapter 3: Tracking

In order to do the filtering magic, previously discussed using a slightly more formal tone in Chapter 2, some difficulties need to be considered. This chapter clarifies real-world challenges such as the determination of which measurements should be used to update the different tracked objects (data association) and how to continuously evaluate track quality. The importance of having a good description of time-stamps is clarified and out-of-sequence measurements are described as an example.

Chapter 4: Modelling

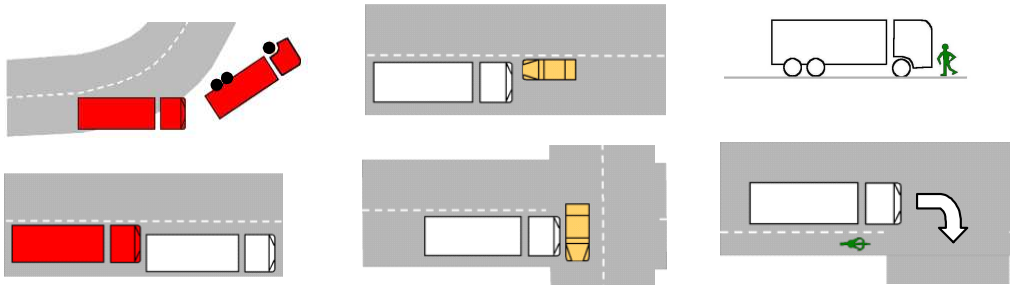
The fourth chapter explains in more detail the models used in an automotive tracking system, e.g., in the filter. The usage of stochastic components is motivated, and two commonly used linear process models are given as examples. Finally, the sensor model and its desired properties are discussed in terms of use in a tracking filter.

Chapter 5: Contributions and future work

Here, we present a summary of the author's contributions in the appended papers, and discuss possible extensions and ideas for future research.

1.2 A background to truck accidents

The consequences of an accident involving a truck are devastating due to the large vehicle weight. A frontal collision at high speed between a truck and a car is often fatal for the people travelling in the car. In a similar collision with two cars of the same size, the occupants normally survive if the collision speed of both vehicles is less than 70 km/h and provided that the occupants are wearing seat belts [9]. Also, a truck is a work environment used on a daily basis by professionals, which separates truck drivers from the general road user. These factors must be considered when planning safety systems for usage in trucks. Design for safety is a leading principle at Volvo and one important resource in this work is the accident research team (ART), formed in the sixties. An extensive database has been built up with information and analyses as to why accidents occur by visiting crash sites and interviewing drivers. Typical accidents involving trucks are shown in Figures 1.1–1.2, where the accident statistics originate from the ART database [10]. For a majority of collisions in Western Europe involving a truck, where the injuries are serious to fatal, the injured road user is travelling in a car. Truck drivers are more exposed to road departure accidents, accounting for 35% of accidents where truck drivers are seriously to fatally injured.



(a) The most common accident types causing severe injury to a truck driver (top not necessarily on a curved road).

(b) The most common accident types causing severe injury to car occupants.

(c) The most common accident types when vulnerable road users are severely injured.

Figure 1.1: The most common accident types for accidents involving a truck in Western Europe, leading to severe or fatal injuries. In 15-20% of these accidents the injured road user drives a truck, in 55-65% a car. Vulnerable road users account for 15-25% of the injuries [10].

Figure 1.2: Rear-end collisions account for 12% of all heavy truck accidents in western Europe. Furthermore, 20% of accidents where a truck driver is severely injured are of this type [10].

It has been agreed within the European union that all trucks must be equipped with lane departure warning systems and automatic braking systems. For trucks weighing over 8 tons, these rules apply to models introduced after 2013 [11]. For lighter trucks, down to 3.5 tons, the rules have been deferred 3 years to 2016.

1.3 In-vehicle safety systems

There are clearly good reasons for equipping vehicles with systems to protect the driver and other road users, and several methods have been proposed for reducing the number of injuries in the automotive environment. In-vehicle systems which try to achieve this are often classified as *passive* safety systems, *active* safety systems, or *integrated* safety systems, depending on which means of action they utilise. These systems aim to reduce either the impact *severity*, the *number* of collisions or the *injury risk* at a given severity. How these types of safety systems, and infrastructural improvements, relate to the means to decrease injuries is illustrated in Table 1.1 [12], although one can argue that infrastructure measures also affect the number of accidents. In some sense, we can regard active safety systems as an extension of improving the infrastructure, e.g., speed limits and crash-barriers — they contribute to reducing injuries in a similar fashion.

Different types of safety systems operate on various time-scales, and require quite different skills in design. Some require a sensing system and advanced control strategies, and continuously adapt their behaviour in order to provide increased safety (active), whereas some fulfil their purpose merely by being present at the time of the collision (passive).

Passive safety

Safety-oriented designs, such as crumple zones (energy-absorbing zones), and seat-belts, have been developed for decades and provide a high level of safety in modern vehicles. Passive safety systems have had a major impact in reducing the injury risk in collisions [13]. Although an airbag system makes use of sensors to detect a collision and trigger the airbag, it is usually classified as a passive safety system.

Active safety

Active safety systems use sensors to continuously observe the traffic environment, the equipped vehicle, or the driver, in order to avoid accidents or to mitigate their consequences. A system can, for example, monitor the surrounding environment to autonomously apply brakes when a collision is imminent, or inform a distracted driver of potential threats.

Early systems include ABS, which entered serial production in larger numbers in the 80's, and, a decade later, electronic stability control systems. Recent systems make use of sensors such as radar and cameras to continuously monitor the traffic-situation. If a situation becomes dangerous, the system may warn the driver or, if deemed necessary, intervene by autonomous braking or steering. Lane departure warning systems and auto-brake systems, which can provide, e.g., low-speed collision avoidance [14], are some examples. Active safety systems are also referred to as preventive systems.

Integrated safety

The term integrated safety is used to describe systems that use sensor information, e.g., regarding a pending collision, to attain the most from a combination of active and passive systems. An advanced integrated safety system could, for example, steer and brake the vehicle such that a deformation zone is fully utilised in a collision, whereas a very simple, yet efficient, example is the seat-belt reminder.

Improvement / Reduction	Active safety	Passive safety	Integrated safety	Infra-structure	Edu-cation
Impact severity	×		×	×	×
Number of accidents	×		×		×
Injury risk		×	×	×	

Table 1.1: There are several ways to contribute towards injury reduction. Effects can be evaluated in terms of reducing either the number of accidents, their severity, or the injury risk. Several in-vehicle safety systems contribute and supplement each other. Education measures are included in the table by the author for comparison.

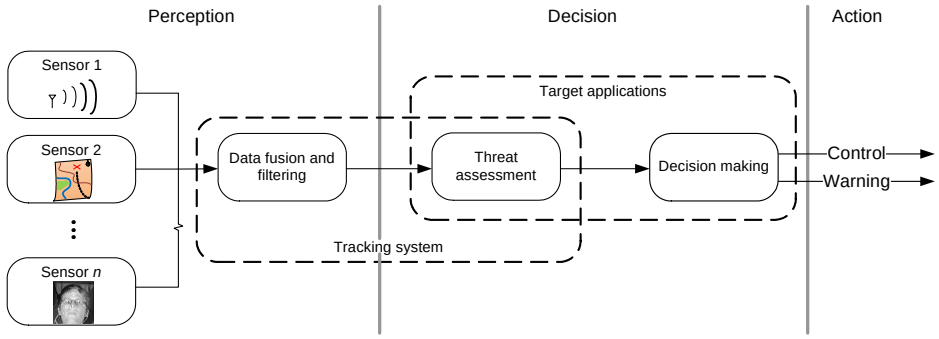


Figure 1.3: Signal processing in active safety systems. The environment and relevant sources of information are monitored, e.g., using radar, map databases and driver state sensors.

1.4 Active safety system design

The safety measures discussed in the previous section all contribute to reduce the number of traffic-related injuries. The work presented in this thesis has focused on how to provide an accurate description of the traffic environment, such that active safety systems can assess the situation and make robust intervention decisions. The purpose of this section is to provide a background on the difficulties in such a description, and explain why it is not straightforward to make decisions.

A three-layered data fusion process model was introduced in [15] to describe an active safety system. It represents the system using three layers; the *perception* layer, the *decision* layer, and the *action* layer. The layers follow a hierarchic structure: they are independent and restricted to one-way communication without feedback. In this sense, the model is somewhat limiting, but nevertheless provides a good abstraction of the components of an active safety system. A system model is provided in Figure 1.3, which has been slightly modified to illustrate that it may sometimes be useful for the perception layer and the decision layer to communicate both ways.

1.4.1 Perception

A vehicle hosting active safety systems to decrease or mitigate collisions is equipped with sensors that observe the surrounding traffic situation. Several sensors, each with different strengths and weaknesses, are generally used jointly to increase performance. The measurements provided by these sensors are usually associated with errors and ambiguities. Hence, a system that relates data from different sensors to each other and connects them to previous measurements is needed to provide a consistent description of the environment. Such a system tracks the observable objects and is consequently called a *tracking system*.

The efficiency of threat assessment and decision-making algorithms is dependent on the quality of the output of the tracking system. Typically, estimates of,

e.g., the position and the velocity of objects surrounding the vehicle are calculated. Improving the estimation accuracy is a motivation for using multiple sensors, so-called data fusion. An effective tracking system consists of several well-tuned interacting components, of which the filter is one. How to design such a system, especially the filter, is the main question at hand in this thesis.

Sensing the environment

There are several types of road-users — cars, trucks, pedestrians and animals to mention just a few. Together with lamp posts, guard rails, and other elements of the traffic infrastructure, the traffic environment is full of objects of different shapes and dynamic behaviour. The great variety of objects the system needs to observe often calls for a combined use of sensors.

Common sensor technologies used in this context are radar, laser ranging, and imaging systems operating in the visual and infrared spectra, all having different advantages and disadvantages. Radar can provide direct measurements of range rate and is relatively indifferent to bad weather, but may suffer from, e.g., low angular resolution; lidar is accurate in range but does not measure range rate and may be sensitive to rain, snow and fog. Vision systems measure angles relatively accurately and are powerful tools for object classification, but provide poor range measurements and are sensitive to bad weather and low light conditions. Additionally, the algorithms used by imaging systems systems can be computationally demanding and the measurements may be difficult to model.

Other sources of information can also be regarded as sensors from a fusion system's point of view. Vehicle-to-vehicle and vehicle-to-infrastructure communication are emerging technologies that are not widely used today, but may play important roles in the future as these would allow vehicles to "see behind corners". Map databases can provide information which cannot easily be measured with other sensors, such as merging of lanes or intersections ahead. So far, map databases are dependent on the accuracy of off-line databases and to a large extent lack information of particular interest to truck drivers, like road inclination and equipage height limitations.

Data fusion and tracking

Sensors make instantaneous observations of the traffic environment and report these to a sensor fusion system, typically a tracking system which provides a refined description, or estimates, to the threat assessment algorithms. To be the information provider to a safety application is a demanding task, taking into consideration the inconvenience and potential danger of a false intervention. Apart from employing robust estimation filters and accurate models, a tracking system must also provide self-assessment capabilities with respect to performance. A tracking filter incorporates different methods to achieve this. It handles events such as erroneous measurements (false alarms), measurements that have failed to

appear (missed detections), detection of new objects (track initialisation), separation of multiple objects, object occlusion, manoeuvering vehicles and noisy measurements. Most tracking filters are designed using statistical methods, based on the Bayesian philosophy, which comes natural given the inherent measurement and prediction uncertainties in the problem at hand — using new, but unreliable, measurements to describe a constantly changing scenario.

In order for the decision-layer to have every chance of succeeding in its assessment, a tracking system must not only provide reliable estimates of the traffic environment, but also describe how good these estimates are. Continuous assessment of the tracker performance is therefore required, and rapid detection of new objects is needed to enable quick decisions in highly dynamic situations; this can, for example, be seen in Paper II, where it is shown that very small differences in decision timing can have a major impact on the intervention outcome.

1.4.2 Decision layer

Using the description of the traffic situation provided by the tracking system, threat assessment algorithms calculate how critical a situation is, e.g. the lateral acceleration required to avoid driving into an object [16]. The purpose of a decision-making algorithm is then to continuously produce a series of decisions, as to whether an intervention of some kind is required or not. Even with a perfect description of the current traffic situation, active safety system decision algorithms still cannot make decisions with absolute confidence that the decision was motivated, at least not as a general rule. The reason is that, for a decision to have an effect on the outcome of a situation, it must be made such that the intended action has time to alter the chain of events to the better. As the chain of events is not known for certain during this time, it constitutes a source of uncertainty inherent in any system that predicts possible outcomes rather than measuring what has actually happened. These uncertainties are denoted *prediction uncertainties*, and together with the *measurement uncertainties* following the relaxation of the assumption of a perfect description of the traffic situation, make decision-making a challenging task. Careful consideration regarding modelling and treatment of these uncertainties will be of most importance. Having done so, the conditional[1] risk for a particular decision can be calculated, a statistic which, when properly interpreted, forms a good basis for decision-making in automotive safety systems. A good illustration of improper usage of such a statistic is given in [17].

1.4.3 The driver

The driver plays a natural and important part in active safety systems — Not only is the driver one of the stake-holders to be protected by the system, but also the judge as to whether the system behaves properly and, sometimes, the

[1] A threat assessment measure is conditioned not only on the measurements, but also the modelling assumptions.

direct cause of the dangerous situation. For this reason it is useful for decision-making algorithms to utilise more or less sophisticated driver models to assess the current need for assistance, as well as how that assistance should be delivered to be regarded by the driver as a correct system behaviour. Driver-state sensors, i.e., sensors capable of estimating, e.g., driver drowsiness or the gaze direction of the driver, could play an important part in decision timing and actuator usage.

Inattention

The reference to driver state sensors motivates a brief discussion on a topic which is not treated further in this thesis: driver awareness and inattention. The reasons for being inattentive are, at a high level, either a consequence of physical impairment such as drowsiness or a consequence of misdirected attention[2]. The impact of inattention in traffic can be intuitively understood; drivers clearly cannot react to threats they are not aware of. However, it is not straightforward to measure the actual impact on accidents. For one, several reasons can contribute to create a dangerous situation, and secondly, admitting to being inattentive or drowsy is discouraged, as it is typically punishable by law. Nevertheless, reports indicate that fatigue may be the underlying cause for 20-40% of all single-vehicle accidents [18], and it was shown in the well-known 100-car study [19] that inattention was a contributing factor in 78% of the recorded crashes.

Many safety measures and safety systems aim to improve driver performance in this sense, as a well-educated and attentive driver will reduce both impact severity and the number of collisions, see Table 1.1.

[2]It wasn't my fault officer, my followers on twitter demand constant updates!

Chapter 2

Filtering

FILTERING, in a wide sense, is the task of processing data such that unwanted components are removed and desired components are made clear — a filter produces estimates of the desired component. In the context of automotive active safety systems, these systems require knowledge of the current traffic situation in order to make intervention decisions. However, the observations made by sensors such as radar, cameras, and gyros, are affected by noise and cannot be trusted without reservation. Furthermore, as measurements are not available at all times there is a need for dynamic models. Taken together, this motivates using reliable and accurate filters in automotive safety systems.

In this chapter, we clarify the problem at hand and express the task in terms of recursive filtering of random processes, or recursive estimation; we use the terms interchangeably. It is made clear why accurate approximate solutions to integrals are important tools for designing such filters robustly, of which commonly used filtering techniques serve as examples. Introductions to estimation in general is given in, e.g., [20] and [21], whereas surveys of approximation techniques used for recursive filtering can be found in [22] and [23].

The formal description of the problem, and its conceptual solution, is given in Section 2.1. Unfortunately, the solution may be impossible or hard to calculate analytically, and it is often necessary to resort to approximations. The prospect of finding an analytical solution is discussed in Section 2.2, whereas common filtering techniques involving approximations are discussed in Section 2.3.

2.1 System model

A discrete-time nonlinear system, described by the state vector, $\mathbf{x}_k \in \mathbb{R}^n$, is assumed to be a first order Markov process, evolving according to the model:

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{w}_{k-1}). \tag{2.1}$$

The time index, $k \in \{0, 1, 2, \dots\}$, corresponds to a continuous-time instant, t_k, and observations, $\mathbf{y}_k \in \mathbb{R}^m$, are provided at these time instances:

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{v}_k). \tag{2.2}$$

In our context, the state vector, $\mathbf{x}_k$, describes the traffic situation. The process model (2.1) then explains (statistically) how the traffic situation changes over time, and the measurement model (2.2) describes how the sensors perceive the traffic situation. The noise terms $\mathbf{w}_k$ and $\mathbf{v}_k$ are random variables which account for model uncertainties. For example, the possibility that the driver of a vehicle hits the brake or initiates a turn is modelled by $\mathbf{w}_{k-1}$, whereas $\mathbf{v}_{k-1}$ explains why the observations are not exact. The process model and the measurement model are often expressed in terms of conditional probability density functions, $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ and $p(\mathbf{y}_k|\mathbf{x}_k)$. These can be calculated from models (2.1)–(2.2) respectively, and the descriptions are often used interchangeably.

The goal is to calculate the posterior distribution, $p(\mathbf{x}_k|\mathbf{Y}_k)$, using all available measurements,

$$\mathbf{Y}_k \triangleq \{\mathbf{y}_1, \dots, \mathbf{y}_k\}, \tag{2.3}$$

and we denote this the filter problem. The general solution for calculating the posterior distribution is given in Section 2.1.1, and is presented in recursive form in Section 2.1.2. Examples of two common estimates of the state, which can be calculated from the posterior distribution, are presented in Section 2.1.3 and conclude this section.

2.1.1 General solution

The posterior distribution can be used to derive estimates of the random variable $\mathbf{x}_k$ given the available measurements $\mathbf{Y}_k$, such as the posterior mean. Estimates are often denoted $\hat{\mathbf{x}}_{k|k}$, where the first subscript refers to the time index of the state and the latter to the time index of the last measurement used to update the state. Using Bayes' theorem, we can express the posterior distribution in terms of the likelihood function, $p(\mathbf{Y}_k|\mathbf{x}_k)$, and the prior, $p(\mathbf{x}_k)$:

$$p(\mathbf{x}_k|\mathbf{Y}_k) = \frac{p(\mathbf{Y}_k|\mathbf{x}_k)p(\mathbf{x}_k)}{\int p(\mathbf{Y}_k|\mathbf{x}_k)p(\mathbf{x}_k)d\mathbf{x}_k}. \tag{2.4}$$

The likelihood function of $\mathbf{x}_k$ is in fact the probability density function (pdf) for the measurement distribution, although it does not necessarily integrate to one with respect to $\mathbf{x}_k$. In other words, the likelihood function indicates the probability to observe the actual measurements, given a particular state. The prior, on the other hand, is the pdf for $\mathbf{x}_k$ before any measurements are made, whereas the denominator in the right-hand side of equation (2.4) scales the product such that the posterior pdf integrates to one. Equation (2.4) is the formal Bayesian solution to the filtering problem — full knowledge of the posterior distribution.

2.1.2 Conceptual recursive solution

The expression for the posterior distribution (2.4) is a function of all available measurements, $\mathbf{Y}_k$, which is usually not a feasible approach. For example, it requires a filter to store an increasing amount of measurements and process all of them whenever an estimate is needed. In practice, each observation, $\mathbf{y}_k$, will be available to the filter soon after the measurement was actually made. It is therefore convenient to formulate an algorithm to incorporate each new observation, $\mathbf{y}_k$, one at the time, without having to redo all previous calculations, which were based on $\mathbf{Y}_{k-1}$.

Assume the posterior distribution from the previous iteration is known to the filter, i.e., assume that $p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$ is known. The so-called prediction density, $p(\mathbf{x}_k|\mathbf{Y}_{k-1})$, can then be calculated by using the system equation (2.1). In other words, we calculate the density for $\mathbf{x}_k$ without using the new measurement, justifying the usage of the word prediction. In this sense, the predictive distribution constitutes a dynamic prior, and the need for this prior is made clear by rewriting the posterior density expression (2.4) in terms of new and old measurements:

$$p(\mathbf{x}_k|\mathbf{y}_k, \mathbf{Y}_{k-1}) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Y}_{k-1})}{\int p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Y}_{k-1})d\mathbf{x}_k}, \tag{2.5}$$

The measurement distribution $p(\mathbf{y}_k|\mathbf{x}_k)$ is known from the measurement model (2.2) and the predictive distribution, $p(\mathbf{x}_k|\mathbf{Y}_{k-1})$, can be calculated from the previous posterior distribution, $p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$, using the Chapman-Kolmogorov equation:

$$p(\mathbf{x}_k|\mathbf{Y}_{k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})d\mathbf{x}_{k-1}, \tag{2.6}$$

where the Markov property, $p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{Y}_{k-1}) = p(\mathbf{x}_k|\mathbf{x}_{k-1})$, is used. Hence, expressed in terms of the process model, the measurement model and the posterior distribution from the previous step, equation (2.5) constitute the formal Bayesian solution to the recursive filtering problem.

2.1.3 Estimates of the state

From a Bayesian perspective, the posterior distribution, $p(\mathbf{x}_k|\mathbf{Y}_k)$, contains all there is to know regarding the state, and — not in a particularly wide sense — "all" can be a lot. It is common to derive estimates, $\hat{\mathbf{x}}_{k|k}$, of $\mathbf{x}_k$ from the distribution, and use these estimates as filter output. An estimator is an algorithm for calculating estimates, and the posterior distribution can be used to find estimators corresponding to any optimality criterion. For example, a vector $\hat{\mathbf{x}}_{k|k}$ is said to be the minimum mean squared error (MMSE) estimate if it minimises the criterion:

$$\hat{\mathbf{x}}_{k|k} = \arg\min_{\hat{\mathbf{x}}_k} \mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)^T(\mathbf{x}_k - \hat{\mathbf{x}}_k)\middle|\mathbf{Y}_k\right], \tag{2.7}$$

where $\mathbb{E}[]$ denotes the expectation of the expression inside the brackets. The MMSE estimate is widely used and is given by the conditional mean, which is shown at the end of this section. Another useful estimate is the maximum a posteriori (MAP) estimate, which is the most likely value for $\mathbf{x}_k$ given $\mathbf{Y}_k$:

$$\hat{\mathbf{x}}_{k|k} = \arg\max_{\mathbf{x}_k} p(\mathbf{x}_k|\mathbf{Y}_k). \tag{2.8}$$

It must not be forgotten that estimates are random variables and should be treated as such[1] when used. In practice, though, the difference between a formal treatment of these uncertainties and a more simple approach may be negligible.

Derivation of the MMSE estimate

The MMSE estimate is given by equation (2.7), and corresponds to the conditional mean. The minimum is found by setting the derivative to zero (see [24] for expressions for derivatives of traces):

$$\frac{\partial}{\partial \hat{\mathbf{x}}_k}\mathbb{E}\left[(\mathbf{x}_k-\hat{\mathbf{x}}_k)^T(\mathbf{x}_k-\hat{\mathbf{x}}_k)|\mathbf{Y}_k\right] = \frac{\partial}{\partial \hat{\mathbf{x}}_k}\mathrm{Tr}\left\{\mathbb{E}\left[(\mathbf{x}_k-\hat{\mathbf{x}}_k)(\mathbf{x}_k-\hat{\mathbf{x}}_k)^T|\mathbf{Y}_k\right]\right\} \tag{2.9}$$

$$= \frac{\partial}{\partial \hat{\mathbf{x}}_k}\mathrm{Tr}\left\{\mathbb{E}\left[(\mathbf{x}_k\mathbf{x}_k^T+\hat{\mathbf{x}}_k\hat{\mathbf{x}}_k^T-\mathbf{x}_k\hat{\mathbf{x}}_k^T-\hat{\mathbf{x}}_k\mathbf{x}_k^T|\mathbf{Y}_k\right]\right\} \tag{2.10}$$

$$= 0 + 2\hat{\mathbf{x}}_k - \mathbb{E}[\mathbf{x}_k|\mathbf{y}_k] - \mathbb{E}[\mathbf{x}_k|\mathbf{Y}_k] = 0. \tag{2.11}$$

It follows from the last row that $\hat{\mathbf{x}}_{k|k} = \mathbb{E}[\mathbf{x}_k|\mathbf{Y}_k]$, which concludes the derivation of the estimator. It can be written as

$$\hat{\mathbf{x}}_{k|k} = \int \mathbf{x}_k p(\mathbf{x}_k|\mathbf{Y}_k)d\mathbf{x}_k. \tag{2.12}$$

However, it may be hard to calculate the estimate; it not only requires the expression for $p(\mathbf{x}_k|\mathbf{Y}_k)$, but also an expression for the integral (2.12). An important result is that if $\mathbf{x}_k, \mathbf{Y}_k$ are jointly Gaussian, the conditional density $p(\mathbf{x}_k|\mathbf{Y}_k)$ is also Gaussian and the MMSE estimator is linear in the data [21].

Derivation of the LMMSE estimator

The best estimator in the MMSE sense, constrained to be linear in data, is denoted the LMMSE estimator. Due to the simple form of the LMMSE estimator, it is very useful, e.g., if we have failed to derive closed-form expressions for the conditional mean. The linearity constraint corresponds to an affine transformation:

$$\hat{\mathbf{x}}_{k|k} = \mathbf{A}_k\mathbf{Y}_k + \mathbf{b}_k. \tag{2.13}$$

[1] Contrary to popular belief, randomness does not disappear just because it is not modelled.

The error should be orthogonal to $\mathbf{b}_k$, i.e., zero-mean when averaged over both the state and the measurements. Updating the form to fulfil this criterion yields:

$$\mathbb{E}[\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}] = \bar{\mathbf{x}}_k - \mathbf{A}_k\bar{\mathbf{Y}}_k - \mathbf{b}_k = 0 \tag{2.14}$$

$$\Rightarrow \mathbf{b}_k = \bar{\mathbf{x}}_k - \mathbf{A}_k\bar{\mathbf{Y}}_k. \tag{2.15}$$

Inserting the above expression for $\mathbf{b}_k$ into the affine transformation given by equation (2.13) yields the expression for the LMMSE estimator:

$$\hat{\mathbf{x}}_{k|k} = \mathbf{A}_k(\mathbf{Y}_k - \bar{\mathbf{Y}}_k) + \bar{\mathbf{x}}_k, \tag{2.16}$$

where $\bar{\mathbf{x}}_k$ denotes $\mathbb{E}[\mathbf{x}_k]$, and $\bar{\mathbf{Y}}_k$ denotes $\mathbb{E}[\mathbf{Y}_k]$. The question is how to select $\mathbf{A}_k$. Using the form (2.16) for the estimator, the MMSE criterion (2.7), to be minimised, takes the form:

$$\hat{\mathbf{x}}_{k|k} = \arg\min_{\mathbf{A}_k} \mathbb{E}\left[(\mathbf{x}_k - \bar{\mathbf{x}}_k - \mathbf{A}(\mathbf{Y}_k - \bar{\mathbf{Y}}_k))^T(\mathbf{x}_k - \bar{\mathbf{x}}_k - \mathbf{A}(\mathbf{Y}_k - \bar{\mathbf{Y}}_k))|\mathbf{Y}_k\right]. \tag{2.17}$$

Similar to the derivation of the MMSE estimator, we set the derivative to zero:

$$\begin{aligned}
&\frac{\partial}{\partial \mathbf{A}_k}\mathrm{Tr}\left\{\mathbb{E}\left[(\mathbf{x}_k - \bar{\mathbf{x}}_k - \mathbf{A}_k(\mathbf{Y}_k - \bar{\mathbf{Y}}_k))(\mathbf{x}_k - \bar{\mathbf{x}}_k - \mathbf{A}_k(\mathbf{Y}_k - \bar{\mathbf{Y}}_k))^T|\mathbf{Y}_k\right]\right\} \\
&\quad = \frac{\partial}{\partial \mathbf{A}_k}\mathrm{Tr}\left\{\mathbb{E}\left[(\mathbf{x}_k - \bar{\mathbf{x}}_k)(\mathbf{x}_k - \bar{\mathbf{x}}_k)^T + \mathbf{A}_k(\mathbf{Y}_k - \bar{\mathbf{Y}}_k)(\mathbf{Y}_k - \bar{\mathbf{Y}}_k)^T\mathbf{A}_k^T|\mathbf{Y}_k\right]\right\} \\
&\qquad + \frac{\partial}{\partial \mathbf{A}_k}\mathrm{Tr}\left\{\mathbb{E}\left[(\mathbf{x}_k - \bar{\mathbf{x}}_k)(\mathbf{Y}_k - \bar{\mathbf{Y}}_k)^T\mathbf{A}_k^T + \mathbf{A}_k(\mathbf{Y}_k - \bar{\mathbf{Y}}_k)(\mathbf{x}_k - \bar{\mathbf{x}}_k)^T\right]\right\} = 0
\end{aligned} \tag{2.18}$$

Let $\boldsymbol{P}_{\mathbf{xx}} = \mathrm{Cov}(\mathbf{x}_k, \mathbf{x}_k)$, $\boldsymbol{P}_{YY} = \mathrm{Cov}(\mathbf{Y}_k, \mathbf{Y}_k)$, $\boldsymbol{P}_{\mathbf{x}Y} = \mathrm{Cov}(\mathbf{x}_k, \mathbf{Y}_k)$, and $\boldsymbol{P}_{\mathbf{x}Y}^T = \mathrm{Cov}(\mathbf{Y}_k, \mathbf{x}_k)$. Replacing the expectations in (2.18) with these matrices makes it easy to calculate the derivative:

$$\frac{\partial}{\partial \mathbf{A}_k}\mathrm{Tr}\left\{\boldsymbol{P}_{\mathbf{xx}} + \mathbf{A}_k\boldsymbol{P}_{YY}\mathbf{A}_k^T + \boldsymbol{P}_{\mathbf{x}Y}\mathbf{A}_k^T + \mathbf{A}_k\boldsymbol{P}_{\mathbf{x}Y}^T\right\} = 2\mathbf{A}_k\boldsymbol{P}_{YY} + \boldsymbol{P}_{\mathbf{x}Y} + \boldsymbol{P}_{\mathbf{x}Y}.$$

For the right-hand side to be zero, i.e., $2\mathbf{A}_k\boldsymbol{P}_{YY} + \boldsymbol{P}_{\mathbf{x}Y} + \boldsymbol{P}_{\mathbf{x}Y} = 0$,

$$\Rightarrow \mathbf{A}_k = \boldsymbol{P}_{\mathbf{x}Y}\boldsymbol{P}_{YY}^{-1} \tag{2.19}$$

$$\Rightarrow \hat{\mathbf{x}}_{k|k} = \bar{\mathbf{x}}_k + \boldsymbol{P}_{\mathbf{x}Y}\boldsymbol{P}_{YY}^{-1}(\mathbf{Y}_k - \bar{\mathbf{Y}}_k), \tag{2.20}$$

which concludes the derivation of the LMMSE estimator; a simple update rule requiring only the first two moments of $\mathbf{x}_k$ and $\mathbf{Y}_k$, and their cross-covariance.

2.2 Exact recursive solutions

The solution (2.5) to the recursive filtering problem derived in the previous section is in general not easy to apply. There are several difficulties: First, closed-form

expressions for the prediction distribution and the measurement distribution must be derived. Second, even if the distributions have closed-form expressions, it must also be possible to evaluate the integral over their product. Finally, if the resulting posterior, $p(\mathbf{x}_k|\mathbf{Y}_k)$, is not on the same form as the posterior from the previous step, $p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$, for which the closed-form expressions discussed above were derived, all expressions must be derived once again for the new distribution.

In rare cases, the issues discussed above have known solutions. The most famous example is the Kalman filter [25], discussed in Section 2.2.1, which calculates the posterior distribution correctly when models are linear and the noise is Gaussian. Another approach is to use grid-based methods to calculate the optimal solution when the state space is discrete with a finite number of states, as briefly discussed in Section 2.2.2.

2.2.1 The Kalman Filter

The Kalman filter (KF) [25] is the optimal estimator in the MMSE sense, when models are linear and noise parameters are Gaussian. Under these conditions it is an efficient estimator, meaning that the posterior covariance matrix attains the Posterior Cramér-Rao lower bound [26].

The process model (2.1) and the measurement model (2.2) can be written

$$\mathbf{x}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{v}_{k-1} \tag{2.21}$$

$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{w}_k, \tag{2.22}$$

where $\mathbf{v}_{k-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k-1})$ and $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$ is zero mean white Gaussian noise. Although the noise does not have to be zero-mean, we assume so for notational convenience.

The Kalman filter assumes that the noise distributions and the initial distribution are Gaussian, and we recall two important results for the Gaussian distribution:

1. The distribution of a linearly-transformed Gaussian distributed random variable is also Gaussian.

2. A Gaussian prior is self-conjugate with respect to a Gaussian likelihood function [27].

A prior distribution and a posterior distribution are said to be conjugate if they belong to the same class of distributions. The first result means that given a Gaussian posterior distribution, the predictive distribution and the measurement likelihood are also Gaussian, and the second result states that, in that case, the

updated posterior distribution is also Gaussian:

$$p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}) = \mathcal{N}(\mathbf{x}_{k-1}; \hat{\mathbf{x}}_{k-1|k-1}, \boldsymbol{P}_{k-1}) \tag{2.23}$$

$$\Rightarrow p(\mathbf{x}_k|\mathbf{Y}_{k-1}) = \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \boldsymbol{P}_{k|k-1}) \tag{2.24}$$

$$\Rightarrow p(\mathbf{y}_k|\mathbf{Y}_{k-1}) = \mathcal{N}(\mathbf{y}_k; \hat{\mathbf{y}}_{k|k-1}, \mathbf{S}_{k|k-1}) \tag{2.25}$$

$$\Rightarrow p(\mathbf{x}_k|\mathbf{Y}_k) = \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k}, \boldsymbol{P}_{k|k}). \tag{2.26}$$

Consequently, the state is always Gaussian distributed and fully described by its conditional mean, $\hat{\mathbf{x}}_{k|\ell}$, and covariance matrix, $\boldsymbol{P}_{k|\ell}$, $(0 \geq \ell \geq k)$. The Kalman filter calculates these moments recursively in two steps: the prediction step and the measurement update step.

Prediction step

The first step is to temporally align the state vector with the new measurement and calculate the predictive distribution, $p(\mathbf{x}_k|\mathbf{Y}_{k-1})$. Recall that only the mean and covariance are needed. Using standard notation we have:

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-1|k-1} \tag{2.27}$$

$$\boldsymbol{P}_{k|k-1} = \mathbf{F}_{k-1}\boldsymbol{P}_{k-1|k-1}\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1} \tag{2.28}$$

The above results follow directly from (2.21) and (2.22), as $\mathbb{E}[(\mathbf{Ax})(\mathbf{Ax})^T] = \mathbf{A}\mathbb{E}[\mathbf{x}\mathbf{x}^T]\mathbf{A}^T$. The covariance normally grows when we predict future states, exemplifying the Bayesian formal treatment of the infamous art of fortune telling.

Measurement update step

The predicted state, described by $p(\mathbf{x}_k|\mathbf{Y}_{k-1})$, is to be updated with the new measurement, $\mathbf{y}_k$, to form the posterior distribution, $p(\mathbf{x}_k|\mathbf{Y}_k)$. Recall from Section 2.1.3 that the MMSE estimator is *linear* and produces the conditional mean, $\hat{\mathbf{x}}_{k|k}$. Consequently, the LMMSE estimator and the MMSE estimator coincide under the Kalman filter assumptions. The correction of the predicted mean is given by direct application of the LMMSE estimator, given by equation (2.20):

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1}\right) \tag{2.29}$$

$$\boldsymbol{P}_{k|k} = \boldsymbol{P}_{k|k-1} - \mathbf{K}_k\mathbf{S}_{k|k-1}\mathbf{K}_k^T. \tag{2.30}$$

The mean and covariance of the measurement distribution is given by

$$\hat{\mathbf{y}}_{k|k-1} = \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1} \tag{2.31}$$

$$\mathbf{S}_{k|k-1} = \mathbf{H}_k\boldsymbol{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k. \tag{2.32}$$

We shall soon motivate the calculation of $\boldsymbol{P}_{k|k}$, but first we turn to the optimal gain matrix, $\mathbf{K}_k$, denoted the *Kalman gain*, given by equation (2.19):

$$\mathbf{K}_k = \boldsymbol{P}_{\mathbf{xy},k|k-1}\mathbf{S}_{k|k-1}^{-1} \tag{2.33}$$

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

The Kalman filter is defined by equations (2.27)–(2.34), which are recursively executed for every new measurement, to calculate the posterior mean and covariance matrix that fully describe the posterior distribution.

Equation (2.30), the updated covariance matrix, is the result of simplifications following the usage of the optimal gain matrix (2.34). For any gain matrix, the posterior covariance matrix takes the form:

$$\begin{aligned}
\boldsymbol{P}_{k|k} &= \mathrm{Cov}\left(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1} - \mathbf{K}_k(\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1})\right) && (2.35)\\
&= \mathrm{Cov}\left(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1} - \mathbf{K}_k(\mathbf{H}_k\mathbf{x}_k + \mathbf{v}_{k-1} - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1})\right) && (2.36)\\
&= \mathrm{Cov}\left(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1} - \mathbf{K}_k\mathbf{H}_k(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}) - \mathbf{K}_k\mathbf{v}_{k-1}\right) && (2.37)\\
&= (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\boldsymbol{P}_{k|k-1}(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T, && (2.38)
\end{aligned}$$

where the last step follows from the independence assumption regarding the process noise. The covariance update form (2.38), sometimes referred to as the *Joseph form*, is valid for any gain matrix $\mathbf{K}_k$, but can be reduced to the form (2.30) when the optimal gain is used. The Joseph form is numerically more stable in presence of, e.g., round-off errors; a product $\mathbf{A}\mathbf{A}^T$ is guaranteed to be positive-semidefinite (see, e.g., Paper I, Appendix A).

The difference between the measurement and the predicted measurement, $\mathbf{y}_k - \hat{\mathbf{y}}_k$, is called the *residual*, or the *innovation*. The posterior mean is a weighted combination of the predicted mean and the innovation, intuitively explaining how model knowledge and evidence both contribute to the updated state. Contrary to the prediction step, the uncertainties are expected to decrease during the update process[2] — we definitely expect to be more certain after the inclusion of evidence into our calculations.

Finally, it should be mentioned that in the case of an unknown noise distribution, the Kalman filter is still the best *linear* estimator, which follows from the Bayesian Gauss-Markov Theorem [20]. Also, many filters take on the Kalman filter name, such as the extended Kalman filter and the unscented Kalman filter. These are in fact approximations of the one-step linear minimum mean squared error estimator discussed in Section 2.3.1.

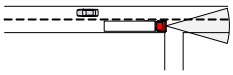
2.2.2 Grid based methods

If the state space can be partitioned into a grid with a finite number of states, $\mathbf{x}_k^{(1)}, \mathbf{x}_k^{(2)}, \ldots, \mathbf{x}_k^{(n)}$, we can use the following form for the posterior distribution:

$$p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}) = \sum_{i=0}^{n} w_{k-1|k-1}^{(i)} \delta(\mathbf{x}_{k-1} - \mathbf{x}_{k-1}^{(i)}) \tag{2.39}$$

$$w_{k-1|k-1}^{(i)} \triangleq \Pr\{\mathbf{x}_{k-1} = \mathbf{x}_{k-1}^{(i)}|\mathbf{Y}_{k-1}\}, \tag{2.40}$$

[2] Here we have assumed linear models; for nonlinear models it is possible for the state covariance matrix to grow during update. This is illustrated in Fig. 2.1, Section 2.3.1.



where $\delta(\mathbf{x}_{k-1} - \mathbf{x}_{k-1}^{(i)}) = 1$ if $\mathbf{x}_{k-1}^{(i)} = \mathbf{x}_{k-1}$ and zero elsewhere. The point in writing the distribution on this form is that the prediction and update procedure corresponds to re-calculating the weights in two simple steps. Firstly, the weights for the predicted state, $w_{k|k-1}^{(i)}$, can be expressed in terms of the transition probabilities, using the Chapman-Kolmogorov equation (2.6):

$$p(\mathbf{x}_k|\mathbf{Y}_{k-1}) = \sum_{i=0}^{n} w_{k|k-1}^{(i)} \delta(\mathbf{x}_k - \mathbf{x}_k^{(i)}) \tag{2.41}$$

$$w_{k|k-1}^{(i)} = \sum_{j=1}^{n} \Pr\{\mathbf{x}_k = \mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(j)}\}\, w_{k-1|k-1}^{(j)}. \tag{2.42}$$

Secondly, multiplication with the likelihood in (2.5) produces weights proportionate to $w_{k|k}^{(i)}$, so the final expression is straightforward:

$$w_{k|k}^{(i)} = \frac{p(\mathbf{y}_k|\mathbf{x}_k^{(i)}) w_{k|k-1}^{(i)}}{\sum_{i=1}^{n} p(\mathbf{y}_k|\mathbf{x}_k^{(i)}) w_{k|k-1}^{(i)}} \tag{2.43}$$

For a grid based method, it is enough that the transition probabilities, $\Pr\{\mathbf{x}_k = \mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(j)}\}$, and the likelihood function, $p(\mathbf{y}_k|\mathbf{x}_k^{(i)})$, can be evaluated. Therefore, the method enables exact filtering of very complex systems, but the price comes in having to partition the state space with sufficient accuracy. The computational demands increase with the number of partitions (weights), seen clearly in equation (2.42). Nevertheless, this method provides an optimal solution to the Bayesian state estimation problem in Eq. (2.5) and was used, e.g., to generate the posterior distribution in Fig. 2.1.

2.3 Approximate recursive solutions

In the previous section we discussed exact solutions to the recursive estimation problem (2.5), and the Kalman filter was introduced together with the grid-based technique. The Kalman filter is the optimal filter if the system can be described using linear equations and additive white Gaussian noise, whereas the grid-approach provides the exact solution if the state space is discrete. Unfortunately, from a filtering perspective, these criteria are rarely met in real-world applications and in this chapter we introduce some of the most commonly used approximations. The LMMSE estimator filtering framework employed by numerous algorithms, such as the extended Kalman filter and several sigma-point methods, is introduced in Section 2.3.1. These are of relatively low computational complexity, whereas the sequential Monte-Carlo methods, which provide asymptotically exact solutions at the price of high computational complexity, are described in Section 2.3.2.

2.3.1 Gaussian filters

The family of *Gaussian filters* solves the recursive estimation problem under the assumption that the concerned distributions are approximately Gaussian. A variety of Gaussian filters have been proposed to cope with nonlinear models [28], and the derivative-free filters [29], [30], [31], [32], [33] are particularly useful; with little or no adjustment, they can be applied to a wide range of problems. These filters use a transformed set of deterministically chosen points, often referred to as *sigma-points*, to approximate means and covariances. Although the perhaps most famous sigma-point method, the Unscented Transform (UT) [29] and [30], does not constrain the distribution to be Gaussian, it can be explained under this assumption. The techniques are closely related to the Gauss-Hermite quadrature rule for numerical integration, and are generally exact for integration over certain polynomial functions. An overview of the estimation techniques is given in Section 2.4, and an extensive analysis of the numerical integration perspective on Gaussian filters is given in [34].

The equations used to compute the posterior mean and covariance in the Gaussian filter are those of the one-step linear minimum mean square error (LMMSE) estimator, which coincide with the well known KF for linear systems. Similarly, the resulting algorithm can be divided into a prediction step and an update step. We begin by expressing the recursion in terms of these steps, and then turn to strategies for calculating the required moments.

The Gaussian filter algorithm and its application

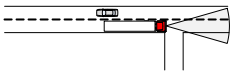
Expressed in terms of the mean and covariance of the distributions of interest, the Gaussian filter recursion is as simple as the KF:

Prediction step: Given $p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$, calculate the first two moments of the state distribution at the time of the next unused measurement:

$$\begin{aligned}\hat{\mathbf{x}}_{k|k-1} &= \mathbb{E}[\mathbf{x}_k|\mathbf{Y}_{k-1}] \\ &= \mathbb{E}[f(\mathbf{x}_{k-1},\mathbf{w}_{k-1})|\mathbf{Y}_{k-1}] \qquad (2.44)\\ \boldsymbol{P}_{k|k-1} &= \mathrm{Cov}(\mathbf{x}_k|\mathbf{Y}_{k-1}) \\ &= \mathbb{E}[f(\mathbf{x}_{k-1},\mathbf{w}_{k-1})f(\mathbf{x}_{k-1},\mathbf{w}_{k-1})^T|\mathbf{Y}_{k-1}] - \hat{\mathbf{x}}_{k|k-1}\hat{\mathbf{x}}_{k|k-1}^T \qquad (2.45)\end{aligned}$$

The predicted density is $p(\mathbf{x}_k|\mathbf{Y}_{k-1}) \approx \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \boldsymbol{P}_{k|k-1})$.

Update step: Correct the predicted density using the measurement, $\mathbf{y}_k$. Assuming that $\mathbf{x}_{k|k-1}$ and $\mathbf{y}_k$ are jointly Gaussian, the LMMSE estimator (2.20) gives



approximations to the posterior mean and covariance:

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \left(\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1}\right) \tag{2.46}$$

$$\boldsymbol{P}_{k|k} = \boldsymbol{P}_{k|k-1} - \mathbf{K}_k \mathbf{S}_{k|k-1} \mathbf{K}_k^T \tag{2.47}$$

$$\mathbf{K}_k = \boldsymbol{P}_{\mathbf{xy},k|k-1} \mathbf{S}_{k|k-1}^{-1}, \tag{2.48}$$

where $\mathbf{K}_k$ is the Kalman gain. The mean, $\hat{\mathbf{y}}_{k|k-1}$, covariance, $\mathbf{S}_{k|k-1}$, and cross-covariance matrix $\boldsymbol{P}_{\mathbf{xy},k|k-1}$ are defined as:

$$\begin{aligned}
\hat{\mathbf{y}}_{k|k-1} &= \mathbb{E}[\mathbf{y}_k|\mathbf{Y}_{k-1}] \\
&= \mathbb{E}[h(\mathbf{x}_k, \mathbf{v}_k)|\mathbf{Y}_{k-1}] && (2.49) \\
\mathbf{S}_{k|k-1} &= \mathrm{Cov}(\mathbf{y}_k|\mathbf{Y}_{k-1}) \\
&= \mathbb{E}[h(\mathbf{x}_k, \mathbf{v}_k) h(\mathbf{x}_k, \mathbf{v}_k)^T|\mathbf{Y}_{k-1}] - \hat{\mathbf{y}}_{k|k-1}\hat{\mathbf{y}}_{k|k-1}^T. && (2.50) \\
\boldsymbol{P}_{\mathbf{xy},k|k-1} &= \mathrm{Cov}(\mathbf{x}_k, \mathbf{y}_k|\mathbf{Y}_{k-1}) \\
&= \mathbb{E}[\mathbf{x}_k h(\mathbf{x}_k,\mathbf{v}_k)^T|\mathbf{Y}_{k-1}] - \hat{\mathbf{x}}_{k|k-1}\hat{\mathbf{y}}_{k|k-1}^T && (2.51)
\end{aligned}$$

The measurement density is $p(\mathbf{y}_k|\mathbf{Y}_{k-1}) \approx \mathcal{N}(\mathbf{y}_k; \hat{\mathbf{y}}_{k|k-1}, \mathbf{S}_{k|k-1})$, for which the predicted density is a conjugate prior. The updated density is therefore $p(\mathbf{x}_k|\mathbf{Y}_k) = \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k}, \boldsymbol{P}_{k|k})$.

There are several filters employing this algorithm, differing only in how they calculate the required moments. We shall give two examples, the extended Kalman filter (EKF), see, e.g., [21], and the so-called cubature Kalman filter (CKF) [33]. Let it be clear that the difficulty in calculating the above moments arise from nonlinear models (2.1)–(2.2), for which the expectations above are not known.

The Extended Kalman Filter **(EKF)**:
A common method for treating nonlinearities is to use a first order Taylor series expansion to approximate the nonlinear transformation with a linear one. The linearised relations can then be used in the standard Kalman filtering framework, resulting in essentially the same equations. The EKF equations are:

$$\hat{\mathbf{x}}_{k|k-1} = f_{k-1}(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{0}) \tag{2.52}$$

$$\boldsymbol{P}_{k|k-1} = \widehat{\mathbf{F}}_{k-1}^{\mathbf{x}} \boldsymbol{P}_{k-1|k-1} \widehat{\mathbf{F}}_{k-1}^{\mathbf{x}\,T} + \widehat{\mathbf{F}}_{k-1}^{\mathbf{v}} \mathbf{Q}_{k-1} \widehat{\mathbf{F}}_{k-1}^{\mathbf{v}\,T} \tag{2.53}$$

$$\hat{\mathbf{y}}_{k|k-1} = h_k(\hat{\mathbf{x}}_{k|k-1}, \mathbf{0}) \tag{2.54}$$

$$\mathbf{S}_{k|k-1} = \widehat{\mathbf{H}}_k^{\mathbf{x}} \boldsymbol{P}_{k|k-1} \widehat{\mathbf{H}}_k^{\mathbf{x}\,T} + \widehat{\mathbf{H}}_k^{\mathbf{w}} \mathbf{R}_k \widehat{\mathbf{H}}_k^{\mathbf{w}\,T} \tag{2.55}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1}) \tag{2.56}$$

$$\boldsymbol{P}_{k|k} = \boldsymbol{P}_{k|k-1} - \mathbf{K}_k \mathbf{S}_{k|k-1} \mathbf{K}_k^T \tag{2.57}$$

$$\mathbf{K}_k = \boldsymbol{P}_{k|k-1} \widehat{\mathbf{H}}_k^{\mathbf{x}} \mathbf{S}_{k|k-1}^{-1}. \tag{2.58}$$

where the linearised model is expressed in terms of the Jacobians $\widehat{\mathbf{F}}^{\mathbf{x}}_{k-1}$, $\widehat{\mathbf{H}}^{\mathbf{x}}_{k}$, and $\widehat{\mathbf{F}}^{\mathbf{v}}_{k-1}$, $\widehat{\mathbf{H}}^{\mathbf{w}}_{k-1}$, accounting for state and noise propagation respectively:

$$\widehat{\mathbf{F}}^{\mathbf{x}}_{k-1} = \frac{\partial}{\partial \mathbf{x}_{k-1}} f^T_{k-1}(\mathbf{x}_{k-1}, \mathbf{v}_{k-1})\big|_{\mathbf{x}_{k-1}=\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{v}_{k-1}=0} \tag{2.59}$$

$$\widehat{\mathbf{F}}^{\mathbf{v}}_{k-1} = \frac{\partial}{\partial \mathbf{v}_{k-1}} f^T_{k-1}(\mathbf{x}_{k-1}, \mathbf{v}_{k-1})\big|_{\mathbf{x}_{k-1}=\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{v}_{k-1}=0} \tag{2.60}$$

$$\widehat{\mathbf{H}}^{\mathbf{x}}_{k} = \frac{\partial}{\partial \mathbf{x}_{k}} h^T_{k}(\mathbf{x}_{k}, \mathbf{w}_{k})\big|_{\mathbf{x}_{k}=\hat{\mathbf{x}}_{k|k-1}, \mathbf{w}_{k}=0} \tag{2.61}$$

$$\widehat{\mathbf{H}}^{\mathbf{w}}_{k} = \frac{\partial}{\partial \mathbf{w}_{k}} h^T_{k}(\mathbf{x}_{k}, \mathbf{w}_{k})\big|_{\mathbf{x}_{k}=\hat{\mathbf{x}}_{k|k-1}, \mathbf{w}_{k}=0}. \tag{2.62}$$

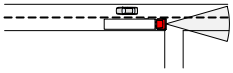
We can regard the linearization procedure as a way to approximate the models by their varying linear ones, to which we can apply the optimal KF equations. This is equivalent to using equations (2.52)–(2.55) to approximate the moments needed in the general Gaussian-filter algorithm.

When the nonlinearities are relatively large, the moment estimates will be inaccurate. The intended LMMSE estimator reduces to a suboptimal linear estimator, of whose performance we know little. The occurrence and origin of errors in the EKF framework are exemplified and analysed in [35]. One can of course use approximations of higher order than one for increased accuracy, the drawbacks being the effort of deriving the filter and increased computational complexity at runtime. In [36], a derivation and evaluation of a second order filter are presented and compared with a first order approximation.

Sigma-point filters:
The family of Sigma-point filters have two significant advantages over the EKF approximation. They are derivative-free and do not require the tedious calculation of Jacobians, and they are generally more accurate. Even the computational complexity is about the same, [30], [37]. The associated cost is in terms of understanding how they should be applied, and how to avoid the unfortunate property of some methods not to guarantee positive-semidefinite covariance matrices. In other words, there is little reason in using the EKF when better methods are available at no particular cost, but the joy of learning a new filter. Having this said, the EKF is a widely applied filter found in a large number of applications — it is clearly a very useful filter and regarded of many as more intuitive. Moreover, certain functions are especially hard to approximate using sigma-point methods, see Section 2.4.3, and in these cases the EKF can be a better alternative.

The family of sigma-point filters approximate integrals, such as the calculations of mean and covariance, using a weighted sum:

$$\int_{\mathbb{R}^n} \mathcal{N}(\mathbf{x}\,; \boldsymbol{\mu}_{\mathbf{x}}, \boldsymbol{P}_{\mathbf{x}}) g(\mathbf{x}) d\mathbf{x} \approx \sum_{i=1}^{\kappa} w_i g(\mathbf{x}^i). \tag{2.63}$$



The κ so-called sigma-points, $\{\mathbf{x}^1, \ldots, \mathbf{x}^\kappa\}$, and the associated weights, w_i, are chosen according to a deterministic scheme. This integral approximation strategy yields the estimator

$$\mathbb{E}[g(\mathbf{x})] \approx \sum_{i=1}^{\kappa} w_i g(\mathbf{x}^i) \triangleq \bar{\mathbf{y}}, \tag{2.64}$$

under the assumption that $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x})$. It is clear that this algorithm for calculating expected values can be directly applied to equations (2.44)–(2.45) and (2.49)–(2.51) in the Gaussian filter algorithm. All we have to do is to evaluate the transforming function in the points, $\{\mathbf{x}^1, \ldots, \mathbf{x}^\kappa\}$. First, we show how to handle the state vector and the noise terms jointly.

Augmenting the state vector:
Nonlinear noise terms in the transforming function need to be included in the calculation of the mean and covariance. A common strategy is to augment the state vector to also include these terms, such that the noise uncertainties are treated jointly with the state uncertainties. For the process model (2.1), the substitution is:

$$\breve{\mathbf{x}}^\mathbf{w}_{k-1|k-1} = \begin{bmatrix} \hat{\mathbf{x}}_{k-1|k-1} \\ \mathbb{E}[\mathbf{w}_{k-1}] \end{bmatrix} \tag{2.65}$$

$$\breve{\boldsymbol{P}}^\mathbf{w}_{k-1|k-1} = \begin{bmatrix} \boldsymbol{P}_{k-1|k-1} & \mathbf{0} \\ \mathbf{0} & \mathrm{Cov}(\mathbf{w}_{k-1}) \end{bmatrix}. \tag{2.66}$$

The corresponding models are expressed in terms of the augmented state but, for convenience, the same notation is used:

$$f(\breve{\mathbf{x}}^\mathbf{w}_{k-1|k-1}) = f(\mathbf{x}_{k-1|k-1}, \mathbf{w}_{k-1}). \tag{2.67}$$

The estimation task is much easier to express in terms of the augmented state vector, and the procedure ensures formal treatment of nonlinear noise terms.

A simple sigma point filter, the Cubature Kalman filter (CKF) is presented here as an example of a sigma-point Kalman filter. Given the posterior distribution from the previous iteration, $p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$, do:

Prediction step

1. Generate sigma points using the n-element state vector, augmented with the process noise term, and their joint covariance matrix:

$$\mathbf{x}^i_{k-1|k-1} = \begin{cases} \breve{\mathbf{x}}^\mathbf{w}_{k-1|k-1} + \left(\sqrt{n\,\breve{\boldsymbol{P}}^\mathbf{w}_{k-1|k-1}}\right)_i & 1 \leq i \leq n \\ \breve{\mathbf{x}}^\mathbf{w}_{k-1|k-1} - \left(\sqrt{n\,\breve{\boldsymbol{P}}^\mathbf{w}_{k-1|k-1}}\right)_{i-n} & n < i \leq 2n \end{cases} \tag{2.68}$$

   where $i = 1, \ldots, 2n$ and $(\sqrt{\boldsymbol{P}})_i$ is the i^{th} column of a matrix square root such that $\sqrt{\boldsymbol{P}}\sqrt{\boldsymbol{P}}^T = \boldsymbol{P}$.

2. Propagate the points through the process model and estimate the mean and covariance of the predictive distribution:

$$\hat{\mathbf{x}}_{k|k-1} = \sum_{i=1}^{2n} w_i \, f(\mathbf{x}^i_{k-1|k-1}) \tag{2.69}$$

$$\boldsymbol{P}_{k|k-1} = \sum_{i=1}^{2n} w_i \, f(\mathbf{x}^i_{k-1|k-1}) f(\mathbf{x}^i_{k-1|k-1})^T - \hat{\mathbf{x}}_{k-1|k-1}\hat{\mathbf{x}}^T_{k-1|k-1}, \tag{2.70}$$

where each weight, $w_i = \frac{1}{2n}$.

Update step

3. Generate sigma points using the η-element state vector, augmented with the measurement noise term, and their joint covariance matrix:

$$\mathbf{x}^i_{k|k-1} = \begin{cases} \breve{\mathbf{x}}^{\mathbf{v}}_{k|k-1} + \left(\sqrt{\eta \, \breve{\boldsymbol{P}}^{\mathbf{v}}_{k|k-1}}\right)_i & 1 \leq i \leq \eta \\ \breve{\mathbf{x}}^{\mathbf{v}}_{k|k-1} - \left(\sqrt{\eta \, \breve{\boldsymbol{P}}^{\mathbf{v}}_{k|k-1}}\right)_{i-\eta} & \eta < i \leq 2\eta \end{cases}. \tag{2.71}$$

4. Propagate the points through the measurement model and estimate the mean and covariance of the measurement distribution, as well as the cross-covariance matrix:

$$\hat{\mathbf{y}}_{k|k-1} = \sum_{i=1}^{2\eta} w_i \, h(\mathbf{x}^i_{k|k-1}) \tag{2.72}$$

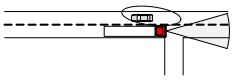
$$\mathbf{S}_{k|k-1} = \sum_{i=1}^{2\eta} w_i \, h(\mathbf{x}^i_{k|k-1}) h(\mathbf{x}^i_{k|k-1})^T - \hat{\mathbf{y}}_{k|k-1}\hat{\mathbf{y}}^T_{k|k-1} \tag{2.73}$$

$$\boldsymbol{P}_{\mathbf{xy},k|k-1} = \sum_{i=1}^{2\eta} w_i \, \mathbf{x}^i_{k|k-1} f(\mathbf{x}^i_{k|k-1})^T - \hat{\mathbf{x}}_{k|k-1}\hat{\mathbf{y}}^T_{k|k-1}, \tag{2.74}$$

where each weight, $w_i = \frac{1}{2\eta}$.

5. Calculate the Kalman gain, the LMMSE estimate, and the matrix-MSE to approximate the mean and covariance of the posterior distribution, $p(\mathbf{x}_k|\mathbf{Y}_k)$, using equations (2.49)–(2.51).

Several integration approximations on the desired form (2.63) are known from literature, and there is a variety of filters employing them. Roughly speaking, these filters can be divided into two categories: those employing a single approximation technique for both first and second order moments, and those who treat these integrals differently to reflect that a covariance matrix is positive-semidefinite. The Unscented Kalman filter (UKF) [30] and the CKF [33] are examples of the



former, whereas the first and second order divided difference filters (DD1, DD2) [31] and the Marginalised Kalman Filter (MKF) described in Paper I belong to the second category. There is a version of the UT, called the scaled UT [38], which applies a different set of rules for calculating the covariance matrix, than those used to calculate the mean. However, since the technique does not guarantee a valid covariance matrix, the method does not qualify in the second category. At the present time these (quite simple) estimations techniques are not detailed as they are discussed in Section 2.4.

Performance of the Gaussian filter

The Gaussian filter makes use of the LMMSE estimator to update the state, and various Gaussian filters apply different methods to approximate the required statistical moments. These filters can generally be tuned such that results are satisfactory also when the underlying assumptions are violated. The purpose of this section is to provide the reader with a deeper understanding of these filters and their implications.

First of all, when the concerned distributions are not jointly Gaussian it is possible to derive better estimators than the LMMSE estimator, albeit not linear. The main reason that the state is assumed Gaussian distributed is because it facilitates the calculation of the mean and covariance of the predicted distribution and the measurement distribution. When this approximation is not accurate, the estimates of the mean and covariance matrix of these distributions will contain errors even if models are linear. In other words, we are generally feeding a suboptimal estimator with false information.

Second, since the filter is based on the LMMSE estimator one may think that it is linear in data. However, the previous measurements are used in a nonlinear fashion, e.g., to calculate the optimal gain matrix. The estimator is therefore linear only in the most recent measurement.

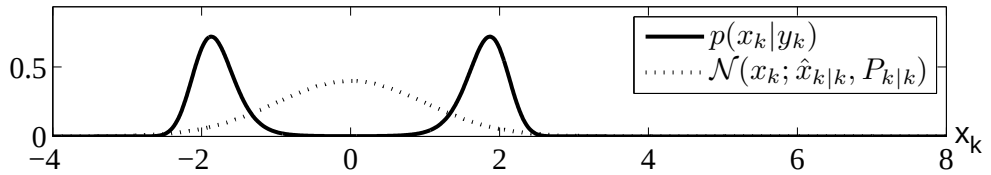
Third, as we will show here, the posterior covariance matrix $\boldsymbol{P}_{k|k}$ is approximated by the matrix-MSE which is sometimes a poor approximation. An illustrative example of the difference between the covariance matrix and the matrix-MSE is the following system: the state, $x_k \sim \mathcal{N}(0,1)$, is to be updated with the observation $y_k = 4$, using the model,

$$y_k = x_k^2 + v, \quad v \sim \mathcal{N}(0, \sigma_v^2). \tag{2.75}$$

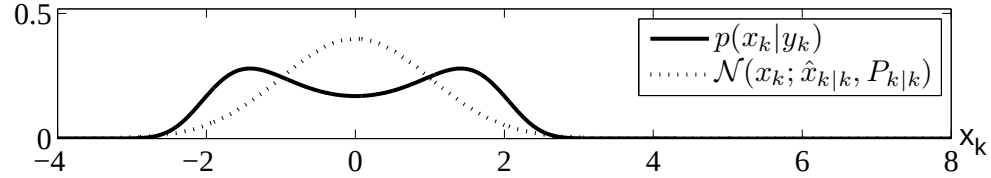
The likelihood function is clearly bi-modal. Fig. 2.1 shows the posterior distribution compared to the Gaussian filter approximation for three different noise models. The MSE resembles the true variance only if the variance of the noise term, σ_v^2, is large enough (recall that $y_k = 4$ in all figures).

The calculation of $\boldsymbol{P}_{k|k}$ is independent of the most recent observation, which is why the Gaussian approximation is the same in the three examples in the Fig. 2.1. The posterior covariance matrix is given by

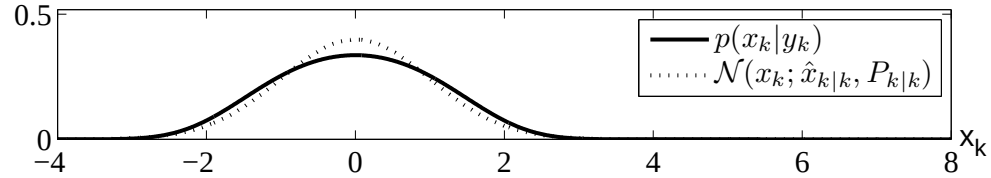
$$\mathrm{Cov}(\mathbf{x}_k|\mathbf{Y}_k) = \mathbb{E}\left[[\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}][\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}]^T \middle| \mathbf{Y}_k\right], \tag{2.76}$$



(a) $\mathbf{y}_k = 4$, $\sigma_v = 1$, and $\mathrm{var}(x_k|y_k) = 3.3$.



(b) $\mathbf{y}_k = 4$, $\sigma_v = 2$, and $\mathrm{var}(x_k|y_k) = 1.8$.



(c) $\mathbf{y}_k = 4$, $\sigma_v = 4$, and $\mathrm{var}(x_k|y_k) = 1.2$.

Figure 2.1: Comparison of the posterior distribution, $p(x_k|y_k = 4)$, to the Gaussian approximation using the LMMSE estimate as mean and its MSE for variance. The measurement model is given by equation (2.75) and the true variance, $\mathrm{var}(x_k|y_k = 4)$, depends on the noise model.

where $\hat{\mathbf{x}}_{k|k}$ is the conditional mean $\mathbb{E}[\mathbf{x}_k|\mathbf{Y}_k]$. However, in the Gaussian filter algorithm, the posterior covariance matrix is approximated as the expected error,

$$\boldsymbol{P}_{k|k} \approx \mathbb{E}\left[[\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}][\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}]^T \middle| \mathbf{Y}_{k-1}\right], \tag{2.77}$$

i.e., the matrix MSE. Conditioned on $\mathbf{Y}_{k-1}$, the LMMSE estimate is $\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1})$. By evaluating the expectation (2.77) it is clear that the final expression corresponds to the Gaussian filter covariance update (2.47):

$$\begin{aligned}
&\mathbb{E}\big[[\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}][\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}]^T|\mathbf{Y}_{k-1}\big] \\
&\qquad = \mathrm{Cov}\left(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1} - \mathbf{K}_k(\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1})|\mathbf{Y}_{k-1}\right) \\
&\qquad = \boldsymbol{P}_{k|k-1} + \mathbf{K}_k\mathbf{S}_{k|k-1}\mathbf{K}_k^T - \boldsymbol{P}_{\mathbf{xy},k|k-1}\mathbf{K}_k^T - \mathbf{K}_k\boldsymbol{P}_{\mathbf{yx},k|k-1} \\
&\qquad = \boldsymbol{P}_{k|k-1} - \mathbf{K}_k\mathbf{S}_{k|k-1}\mathbf{K}_k^T
\end{aligned} \tag{2.78}$$

where we used the relation $\boldsymbol{P}_{\mathbf{xy},k|k-1} = \boldsymbol{P}_{\mathbf{yx},k|k-1}^T = \mathbf{K}_k\mathbf{S}_{k|k-1}$.

We draw three conclusions from the above discussion. First, the matrix-MSE can be a poor substitute for the posterior covariance matrix, as illustrated in

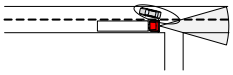


Fig. 2.1. Second, the errors introduced in the required moment approximations are not accounted for in the model. This leads to over-confident filters and it can be beneficial to exaggerate the noise terms in the model to compensate for this. Third, it is somewhat difficult to use the Gaussian filter when distributions are multi-modal, as such distributions are not well described by the mean and covariance.

A practical aspect of Gaussian filtering involves using numerically stable methods. Some of the operations involved in calculating the covariance matrices are sensitive to numerical errors, which may affect the long-term stability of the filter. For example, the subtraction in equation (2.47) is not guaranteed to produce a positive definite covariance matrix. Many sigma point filters have so-called square-root versions, e.g., [39], [40], and [33], designed for improved numerical stability.

2.3.2 Particle filters

The approximate filters described in the previous section provide a framework for estimating mean and covariance of the posterior distribution, using the LMMSE estimation algorithm. However, a general rule is that when nonlinearities in the model become more severe, the performance or the linear estimator deteriorates. For some systems, we may have to use more exact methods and, at some point, we would like to compare our filter to a much better filter for benchmark purposes. This section shows how to use computationally demanding methods to provide asymptotically exact solutions to the recursive estimation problem (2.5).

The idea is to let a set of randomly generated samples, referred to in this context as particles, represent the whole distribution. This representation has its roots in Monte Carlo methods for numerical integration — an integral over the distribution can be written as a sum of the particles.

Monte Carlo integration

For a probability density, $\pi(\mathbf{x})$, the integral approximation takes the form:

$$\mathbf{I} = \int g(\mathbf{x})\pi(\mathbf{x})d\mathbf{x} \approx \frac{1}{N}\sum_{i=1}^{N} g(\mathbf{x}^{(i)}) = \mathbf{I}_N \tag{2.79}$$

$$\mathbf{I}_N \xrightarrow{a.s} \mathbf{I} \quad \text{as} \quad N \to \infty \tag{2.80}$$

which holds if the set $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)}\}$ contains samples drawn from the actual pdf, $\pi(\mathbf{x})$. For some distributions it is straightforward to generate samples, either directly or by using, e.g., Markov chain Monte Carlo methods such as the Metropolis-Hastings algorithm [41]. For other distributions, such as those typically encountered in a tracking filter, it is difficult to generate a representative set using these methods. Fortunately, so-called importance sampling can be employed to substitute the desired set of samples with one we *can* generate samples from. It is a useful method that can be formulated in a sequential manner suitable for use in a tracking system, yielding the particle filter.

Importance sampling

Importance sampling uses weighted samples from one density to represent another density, in the integral approximation (2.79). Instead of generating samples from the unknown density, in our case $p(\mathbf{x}_k|\mathbf{Y}_k)$, which we want to calculate, we define a density $q(\mathbf{x}_k|\mathbf{Y}_k)$ from which we can generate samples, a so-called *importance density*. Introducing the so-called *importance weights*,

$$\tilde{w}^{(i)} = \frac{p(\mathbf{x}_k^{(i)}|\mathbf{Y}_k)}{q(\mathbf{x}_k^{(i)}|\mathbf{Y}_k)}, \tag{2.81}$$

we can create two sets of samples to represent the distribution, $p(\mathbf{x}_k|\mathbf{Y}_k)$, which are equivalent in the sense that they both calculate the integral (2.79) asymptotically correct:

$$\{\mathbf{x}_p^{(i)}\}_{i=1}^N \quad : \mathbf{x}_p^{(i)} \sim p(\mathbf{x}_k|\mathbf{Y}_k) \tag{2.82}$$

$$\{\mathbf{x}_q^{(i)}\}_{i=1}^N \quad : \mathbf{x}_q^{(i)} \sim q(\mathbf{x}_k|\mathbf{Y}_k). \tag{2.83}$$

The integral approximations using these sets respectively, are identical when the influence of the weights are taken into aspect:

$$\mathbf{I}_N = \frac{1}{N}\sum_{i=1}^{N} g(\mathbf{x}_p^{(i)}) = \frac{1}{N}\sum_{i=1}^{N} \tilde{w}^{(i)} g(\mathbf{x}_q^{(i)}). \tag{2.84}$$

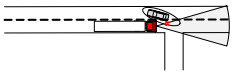
By changing to the set (2.83), we can represent the unknown distribution, $p(\mathbf{x}_k|\mathbf{Y}_k)$, without having to generate samples from it. We do have to calculate the weights (2.81) but, in fact, it is not necessary to evaluate $p(\mathbf{x}_k|\mathbf{Y}_k)$ to do so, it is enough to evaluate a function proportional to $p(\mathbf{x}_k|\mathbf{Y}_k)$ and normalise the weights:

$$w^{(i)} = \frac{\tilde{w}^{(i)}}{\sum_{i=1}^{N} \tilde{w}^{(i)}}. \tag{2.85}$$

We are free to choose any suitable importance density, which will be further discussed below, but it is however required that the support of $q(\mathbf{x}_k|\mathbf{Y}_k)$ contains the support of $p(\mathbf{x}_k|\mathbf{Y}_k)$, i.e.:

$$q(\mathbf{x}_k|\mathbf{Y}_k) > 0 \quad \text{if} \quad p(\mathbf{x}_k|\mathbf{Y}_k) > 0. \tag{2.86}$$

The constraint on the support can be understood from the following example: if there exists a region, $\mathcal{S}$, for which $p(\mathbf{x}_k \in \mathcal{S}) > 0$ and $q(\mathbf{x}_k \in \mathcal{S}) = 0$, the probability for generating samples in the region $\mathcal{S}$ is zero when using $q(\mathbf{x}_k)$. Consequently, $p(\mathbf{x}_k)$, which can generate samples in $\mathcal{S}$, cannot possibly be represented by $q(\mathbf{x}_k)$.



Sequential importance sampling and the particle filter algorithm

Several strategies for implementing the recursive filter described by Eq. (2.5) in Section 2.1.2 are known from literature, see, e.g., textbooks [42], [22], and comprehensive overviews in [43] and [44]. Most methods make use of a sequential importance sampling (SIS) based strategy. Given a set of particles from a previous recursion, $\{\mathbf{x}_{k-1}^{(1)}, \mathbf{x}_{k-1}^{(2)}, \ldots, \mathbf{x}_{k-1}^{(N)}\}$, and associated weights, $\{w_{k-1}^{(1)}, w_{k-1}^{(2)}, \ldots, w_{k-1}^{(N)}\}$, the strategy is to, for each iteration:

1. generate new particles from the existing set such that

$$\mathbf{x}_k^{(i)} \sim q(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}\mathbf{y}_k).$$

2. update the associated weights and approximate the posterior distribution

$$p(\mathbf{x}_k|\mathbf{Y}_k) \approx \sum_{i=1}^{N} w_k^{(i)} \delta(\mathbf{x}_k - \mathbf{x}_k^{(i)}).$$

A filter that performs these two steps is commonly denoted a particle filter (PF), and the complete derivation is given, e.g., in [22]. If we can choose an importance density that only requires knowledge of $\mathbf{x}_{k-1}^{(i)}$ and $\mathbf{y}_k$, the weights are updated as

$$\tilde{w}_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(\mathbf{y}_k|\mathbf{x}_k^{(i)}) p(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})}{q(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{y}_k)} \tag{2.87}$$

$$w_k^{(i)} = \frac{\tilde{w}_k^{(i)}}{\sum_{i=1}^{N} \tilde{w}_k^{(i)}}. \tag{2.88}$$

It is straightforward to calculate estimates from these distributions. For example, the conditional mean (and MMSE estimate) is:

$$\mathrm{E}[\mathbf{x}_k|\mathbf{Y}_k] = \sum_{i=1}^{N} w_k^{(i)} \mathbf{x}_k^{(i)}. \tag{2.89}$$

Other estimates can also be generated from these samples, see for example [45] for a method on how to calculate the MAP estimate.

Choosing importance density

We claimed that any density with the right support can be used as an importance density. However, the numerical approximation is exact only when $N \to \infty$, and any embodiment must naturally rely on a limited set of particles. The better the importance density, the more accurate estimate is given, for a fixed number

of particles. A capable importance density is the Gaussian approximation of the posterior distribution given by an LMMSE estimator,

$$q(\mathbf{x}_k^{(i)}|\mathbf{Y}_k) = \mathcal{N}(\mathbf{x}_k^{(i)}; \hat{\mathbf{x}}_k, P_{k|k}) \tag{2.90}$$

from which we can easily generate samples. The moments $\hat{\mathbf{x}}_k$ and $P_{k|k}$ can be calculated, e.g., using the EKF or a sigma-point filter. The resulting filter is sometimes denoted the local linearization particle filter (LLPF) [22].

A density that is tempting to use is the prior, i.e., to generate the particles from the predicted distribution:

$$\mathbf{x}_k^{(i)} \sim p(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}). \tag{2.91}$$

Obviously this makes the filtering process much easier, since $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ in the tracking framework corresponds to a dynamic motion model, from which it is generally easy to generate samples. Furthermore, the particle weight update (2.87) is reduced to evaluating the likelihood, as the predictive models cancel out:

$$\tilde{w}_k^{(i)} \propto w_{k-1}^{(i)} p(\mathbf{y}_k|\mathbf{x}_k^{(i)}). \tag{2.92}$$

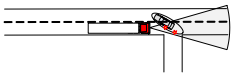
A drawback with this method, compared to the LLPF, is that the importance density is independent of the measurements. This technique is used in the bootstrap filter [46], an algorithm that cleared the way for the numerous variants of particle filters developed today. The main contribution in the article is a method for preventing the degradation of estimates over time. This so-called degeneracy problem means that after some time, usually only a few iterations, a few particle weights are dominant and the rest are near zero. It can be shown [43] that the unconditional variance of the importance weights can only increase over time, hence this problem is unavoidable. The solution presented in [46] was to resample particles from the distribution, before degeneration becomes critical. Thus the number of particles with negligible weights is kept low. A somewhat violent metaphor is that particles with low weights are killed whereas the good ones are multiplied. A good criterion to measure degeneracy is the effective mass:

$$J_k^{\text{eff}} = \frac{1}{\sum_{i=1}^{N} (w_k^{(i)})^2}. \tag{2.93}$$

The efficient mass was introduced in [47] and resampling is initiated whenever $J_k^{\text{eff}} < J^{\text{Threshold}}$. It is important to understand that resampling decreases the accuracy of the approximation, but lays the foundation for improved future results. Therefore, in an iteration where the resampling criterion is met, desired moments should be calculated before the resampling takes place.

Rao-Blackwellization

If a part of the posterior distribution can be analytically calculated, it is possible to reduce the total number of particles, as they do not need to represent these



dimensions. This is known as Rao-Blackwellization, to which a good tutorial and derivation is given in [48]. The idea is illustrated by a simple example.

Example: *Assume the state vector can be partitioned into a conditionally linear and a nonlinear part, such that*

$$\mathbf{x}_{k+1} = \begin{bmatrix} \mathbf{x}_{k+1}^l \\ \mathbf{x}_{k+1}^n \end{bmatrix} = \begin{bmatrix} F(\mathbf{x}_k^n)\mathbf{x}_k^l \\ f(\mathbf{x}_k^n) \end{bmatrix} + \begin{bmatrix} \mathbf{v}_k^l \\ \mathbf{v}_k^n \end{bmatrix}, \quad \begin{bmatrix} \mathbf{v}_k^l \\ \mathbf{v}_k^n \end{bmatrix} \sim \mathcal{N}(0, \mathbf{Q})$$

and

$$\mathbf{y}_k = h(\mathbf{x}_{k+1}^n) + H(\mathbf{x}_{k+1}^n)\mathbf{x}_k^l + \mathbf{w}, \quad \mathbf{w} \sim \mathcal{N}(0, \mathbf{R}).$$

This could represent, e.g., a scenario where measurements are biased. The posterior distribution is:

$$p(\mathbf{x}_k|\mathbf{Y}_k) = p(\mathbf{x}_k^l|\mathbf{x}_k^n, \mathbf{Y}_k)p(\mathbf{x}_k^n|\mathbf{Y}_k).$$

The left density is conditionally linear, and can be calculated, e.g., using the Kalman filter, whereas the right distribution is calculated using a Particle filter:

$$p(\mathbf{x}_k|\mathbf{Y}_k) = \sum_{i=1}^{N} \mathcal{N}(\mathbf{x}_k^l; \hat{\mathbf{x}}_k^{l(i)}, \boldsymbol{P}_k^{(i)}|\mathbf{x}_k^n)w^{(i)}\delta(\mathbf{x}_k^n - \mathbf{x}_k^{n(i)}). \tag{2.94}$$

The resulting filter is often denoted as a *marginalized particle filter*, as the linear state variables are analytically marginalized out.

2.4 Moment estimation using sigma-points

The Gaussian filter framework, discussed in Section 2.3.1, provides a solution to the recursive estimation problem under the assumption that the concerned distributions are approximately Gaussian, but requires the calculation of several expectation integrals (2.44)–(2.45), (2.49)–(2.51). In this section, we present a brief overview of sigma-point methods for calculating such integrals. A more thorough examination is presented in [49], whereas [34] sheds light upon the numerical integration perspective. A new approach towards using the sigma-points to calculate these moments is the topic of Paper I appended to this thesis, which presents the marginalised transform (MT) and the Marginalised Kalman Filter (MKF).

In Section 2.4.1, we discuss numerical integration methods and their properties, especially concerning integration over the Gaussian distribution. Two different approaches for using such rules in a filtering framework are discussed in Section 2.4.2 and, last, we provide some examples in Section 2.4.3.

2.4.1 Exact integration of polynomial functions

Consider a transformation, $g: \mathbb{R}^n \rightarrow \mathbb{R}^m$ and a stochastic variable $\mathbf{x} \in \mathbb{R}^n$ with probability density function

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}),$$

where g, $\boldsymbol{\mu}_\mathbf{x}$ and $\boldsymbol{P}_\mathbf{x}$ are all known. We wish to calculate the expected value of the transformed variable

$$\mathbf{y} = g(\mathbf{x}), \quad \mathbf{y} \in \mathbb{R}^m.$$

This first moment is given by the integral expression

$$\mathbb{E}[\mathbf{y}] = \int_{\mathbb{R}^n} \mathcal{N}(\mathbf{x}\,; \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) g(\mathbf{x}) d\mathbf{x}. \tag{2.95}$$

Sigma-point methods provide approximate solutions to integrals such as this one and have attractive properties with respect to performance and simplicity. The approximation can be written in terms of a weighted sum of the function values in κ points, with κ associated weights:

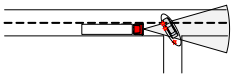
$$\int \mathcal{N}(\mathbf{x}\,; \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) g(\mathbf{x}) d\mathbf{x} \approx \sum_{i=1}^{\kappa} w_i g(\mathbf{x}^i). \tag{2.96}$$

A rule for selecting these points and weights is called a *quadrature*, or a quadrature rule, and are designed to calculate the integral correctly for some particular functions, not limited to products with probability density functions such as in our case. A Gaussian quadrature, for example, calculates (2.96) correctly when the product $\mathcal{N}(\mathbf{x}\,; \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) g(\mathbf{x})$ is a polynomial of some order. The Gauss-Hermite quadrature, however, is adapted to be exact for polynomials that are multiplied with the Gaussian pdf, meaning that the integral (2.96) is correctly calculated if $g(\mathbf{x})$ is a polynomial. In the literature, an important concept used to summarise the performance of a quadrature is the *precision*. The definition of the precision of an integration rule is [34]:

'A rule is said to have precision p if it integrates monomials up to degree p exactly, that is, monomials $\Pi_{i=1}^d x_i^{k_i}$ *with* $k_i \geq 0$ *and* $\sum_{i=1}^d k_i \leq p$*, but not exactly for some monomials of degree* $\sum_{i=1}^d k_i = p + 1$*'.*

Gauss-Hermite quadrature

If g is a polynomial, integrals such as (2.96) are calculated exactly by the Gauss-Hermite quadrature rule. The evaluation points $\mathbf{x}^i = [x_1^i, \ldots, x_n^i]^T$, for $i \in$



$\{1, 2, ..., \kappa\}$ and their associated weights w_i are given, e.g., in [50]. The Gauss-Hermite quadrature of precision $p = 2\zeta - 1$ is written:

$$\int \mathcal{N}(\mathbf{x}\,;\boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) g(\mathbf{x}) d\mathbf{x} \approx \sum_{\ell=1}^{\zeta} w_\ell \dots \sum_{j=1}^{\zeta} w_j g([x_1^j, \dots, x_n^\ell]^T)$$
$$= \sum_{i=1}^{\kappa} w_i g(\mathbf{x}^i), \tag{2.97}$$

where the total number of points $\kappa = \zeta^n$ relates to the precision according to $\zeta = \frac{p+1}{2}$. Unfortunately, the quadrature suffers from the curse of dimensionality; the number of required function evaluations using a precision 3 rule for $\mathbf{x} \in \mathbb{R}^2$ is only 4, but for $\mathbf{x} \in \mathbb{R}^8$ the quadrature requires 256 evaluations.

The Gauss-Hermite quadrature is more thoroughly explained in [32]. The Gauss-Hermite filter (GHF) [28] applies this quadrature to the Gaussian filter calculations described in Section 2.3.1.

The unscented transform and the cubature rule

The unscented transform was introduced in the mid 90's [29], [51] as a novel approach for moment estimation in nonlinear systems. The corresponding Gaussian filter is denoted UKF and an introduction to the method is given in, e.g., [30].

The rule has precision 3 and makes use of $\kappa = 2n + 1$ evaluations:

$$\int \mathcal{N}(\mathbf{x}\,;\boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) g(\mathbf{x}) d\mathbf{x} \approx \sum_{i=1}^{2n+1} w_i g(\mathbf{x}^i). \tag{2.98}$$

The evaluated points are denoted *sigma-points*, and it was shown in [52] that the UT realises a version of the fully symmetric integration formula presented almost three decades earlier in [53]. The version employed by the UT is exact for integration over certain polynomial functions, weighted by the Gaussian pdf. The sigma-points and their associated weights are selected as follows:

$$\mathbf{x}^0 = \quad \mathbb{E}[\mathbf{x}] \tag{2.99}$$

$$\mathbf{x}^i = \begin{cases} \mathbb{E}[\mathbf{x}] + \left(\sqrt{\frac{n}{(1-w_0)}\boldsymbol{P}_\mathbf{x}}\right)_i, & 1 \le i \le n \\ \mathbb{E}[\mathbf{x}] - \left(\sqrt{\frac{n}{(1-w_0)}\boldsymbol{P}_\mathbf{x}}\right)_{i-2n/2}, & n < i \le 2n \end{cases} \tag{2.100}$$

$$w_i = \frac{1 - w_0}{2n}, \tag{2.101}$$

where $i = 1, \dots, 2n$ and $(\sqrt{\boldsymbol{P}_\mathbf{x}})_i$ is the i^{th} column of a matrix square root such that $\sqrt{\boldsymbol{P}_\mathbf{x}}\sqrt{\boldsymbol{P}_\mathbf{x}}^T = \boldsymbol{P}_\mathbf{x}$. The mean weight w_0 is a free variable but it is suggested that $w_0 = 1 - n/3$, which corresponds to a rule that integrates monomials without

cross-terms up to order 5. More specifically, the rule has precision 3, but calculates terms on the form x_i^κ exactly for $\kappa \leq 5$.

The so called cubature rule, used in the CKF presented in Section 2.3.1, can be obtained from this scheme by setting $w_0 = 0$, effectively removing $\mathbf{x}^0$ from the set. The cubature rule, consequently, has precision 3 and requires $2n$ points.

Apart from numerous tracking and filtering tasks, the UT has been used for, e.g., sensitivity analysis for a variety of systems including antenna characterisation [54], power system analysis [55] and circuit design [56]. Contrary to the Gauss-Hermite quadrature, both the UT and the cubature rule scale very well with the number of dimensions, using $2n+1$ and $2n$ points respectively. Where the Gauss-Hermite quadrature required 256 points, for $\mathbf{x} \in \mathbb{R}^8$, the UT needs only 17.

The marginalised transform

A different approach towards exploiting the information in the Sigma-points was taken in [1], and is further developed in Paper I. The proposed method aims at being optimal on average for a larger family of polynomials than what the methods discussed above are exact for. However, the method can be designed such that it yields the integration rules of these methods, albeit covariance matrices are calculated differently. Some examples are included in Section 2.4.3.

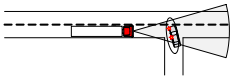
2.4.2 Application in a filtering framework

There are three different types of integrals in the Gaussian filter framework; calculation of means, covariance matrices and cross-covariance matrices. Some sigma-point filters treat all moment integrals the same, regardless of whether the first or the second order moment is calculated. Other filters make use of models to incorporate the known relation between the first and second order moment of a random variable.

In Section 2.4.1 we considered the integration over a generic function, $g(\mathbf{x})$, see (2.95). This function can represent, e.g., the process model used in the prediction step, and the integral would then correspond to the predicted mean. The integral for the covariance matrix of the predicted state is:

$$\begin{aligned}\mathrm{Cov}(f(\mathbf{x})) &= \int_{\mathbb{R}^n} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) \big[f(\mathbf{x}) - \mathbb{E}[f(\mathbf{x})]\big] \big[f(\mathbf{x}) - \mathbb{E}[f(\mathbf{x})]\big]^T d\mathbf{x} \\ &= \int_{\mathbb{R}^n} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) f(\mathbf{x}) f(\mathbf{x})^T d\mathbf{x} - \mathbb{E}[f(\mathbf{x})]\mathbb{E}[f(\mathbf{x})]^T. \end{aligned} \tag{2.102}$$

Clearly, the integral can be expressed on the familiar form (2.96), with $g(\mathbf{x}) = f(\mathbf{x})f(\mathbf{x})^T$. In other words, one approach is to approximate the non-central second order moment and obtain the covariance matrix by subtracting the outer product of the mean, approximated in the previous step. Two different strategies can be applied here: either the same rule is applied to both integrals, $\int \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) f(\mathbf{x}) d\mathbf{x}$ and $\int \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) f(\mathbf{x}) f(\mathbf{x})^T d\mathbf{x}$, or some measures are taken to distinguish the two integrals.



Direct application

The Gaussian filters employing the UT (the UKF) and the Cubature rule (the CKF) both apply the integration formula directly to the integral expressions for the mean and the non-central second order moment. This is often a successful approach, but if the mean is correctly calculated using a *minimum* number of evaluations, the covariance matrix estimate will in general not be exact. This is not a major drawback — we are making approximations after all — what's worse is that with negative weights, the covariance matrix estimate may not even be positive-semidefinite (see proof in Paper I, Appendix A).

The risk for the latter is a common objection against using the UKF and one of the reasons for employing the cubature rule, which has only positive weights. On the other hand, the UKF can be tuned by moving the sigma-points, whereas the CKF cannot be tuned at all. The scaled UT, mentioned in Section 2.3.1, does employ different weights when calculating the covariance matrix, but not in a fashion that matters in this aspect.

Polynomial models

Another approach is to model the transforming function and calculate the covariance matrix conditioned on the model. The central difference Kalman filter (CDKF) [28], and the second order divided difference filter (DD2) [31], both use a second order polynomial approximation of the transforming function (excluding cross-terms). The covariance matrix is calculated using this model and is therefore always positive-semidefinite. The CDKF and the DD2 are essentially the same filters [49], and the mean is calculated using the same points and weights as the UT. Consequently, both methods have precision 3 in this aspect. In practice, a different set of weights is applied when calculating the covariance matrix. For example, the DD2 covariance matrix is given by:

$$\begin{aligned}\mathrm{Cov}(g(\mathbf{x})) \approx &\frac{1}{4h^2}\sum_{i=1}^{n}\left[g(\mathbf{x}^i)-g(\mathbf{x}^{i+n})\right]\left[g(\mathbf{x}^i)-g(\mathbf{x}^{i+n})\right]^T \\ &+\frac{h^2-1}{4h^4}\sum_{i=1}^{n}\left[g(\mathbf{x}^i)+g(\mathbf{x}^{i+n})-2g(\mathbf{x}^0)\right]\left[g(\mathbf{x}^i)+g(\mathbf{x}^{i+n})-2g(\mathbf{x}^0)\right]^T.\end{aligned} \tag{2.103}$$

using $h=\sqrt{\frac{n}{(1-w_0)}}$, the sigma-points are given by equations (2.99)–(2.100).

The MT, using the Hermite-polynomial base functions, also belongs to the family of polynomial interpolation filters. However, it differs from the above methods in that the interpolated polynomial can be modelled as a polynomial of arbitrary order.

2.4.3 Application examples

The sigma-point methods discussed above are exact for certain polynomial functions, but arguably many functions are not well approximated by low-order polynomials (recall that the precision of the more commonly used versions is 3). Furthermore, with the direct approach used by the UT and the cubature rule an expectation is also carried out over the squared function, which consequently also should be well approximated by a polynomial. Model-based methods, on the other hand, magnify errors present in the initial approximation when calculating the covariance matrix.

The purpose of this section is to illustrate how these methods perform in estimation tasks. First we study the prediction of a vehicle — an important task in active safety systems. Then we examine transformations for which the results can be less satisfactory and discuss the reason for this.

Predicting vehicle motion

An example relevant to automotive safety is predicting the path of a vehicle, which is done in threat assessment algorithms and in the tracking system. In Fig. 2.2, we illustrate a 200 ms long prediction of a vehicle moving according to the nonlinear motion model used in Papers II and III. The prior distribution is assumed Gaussian, but the predicted distribution is not — it is rather shaped like a banana. We apply the UT and the linearization technique used by the EKF to estimate the mean and covariance of the predicted distribution, and compare the result to a reference Monte-Carlo simulation. Their respective covariance contours are plotted under the assumption that the distributions are Gaussian. In terms of the Kalman filter algorithm, the operation corresponds to the prediction step (2.44)–(2.45). The estimates from the linearised predictor do not approximate the true distribution as well as the estimate from the UT, which is very accurate. This example indicates a significant advantage with sigma-point methods.

Challenging transformations

The estimation quality of sigma-point methods can be quite sensitive to the position of the sigma points. Fig. 2.3 illustrates the application of sigma-point methods to sinusoidal transformations, which can be particularly difficult due to the periodicity. A comparison of Fig. 2.3a to Fig. 2.3c, shows that a sigma-point method that performs well for one transformation can be a bad choice for another transformation. This is made very clear when calculating the variance for two seemingly similar transformations of $x \sim \mathcal{N}(0, 1)$:

$$y_1 = \cos(x), \text{ and } y_2 = \cos(x + \frac{\pi}{4}).$$

The cubature estimate of $\text{var}(y_1)$ is zero, whereas for y_2 the estimate is nearly exact. In Fig. 2.3c–2.3d the interesting region resembles a polynomial to a much higher extent, and all sigma-point methods performs well.

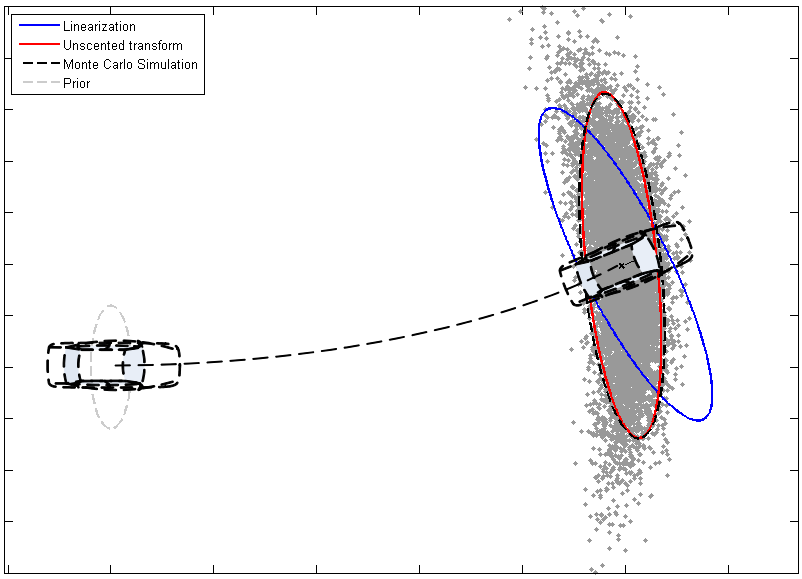


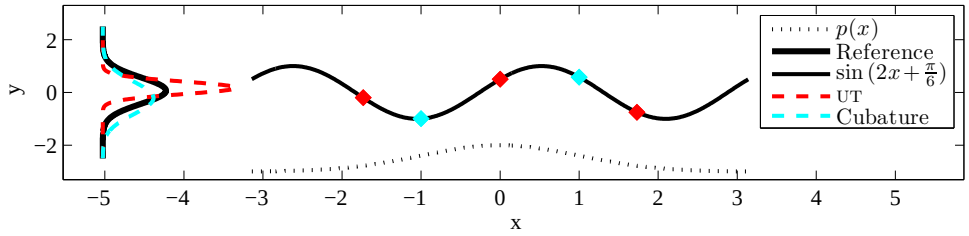
Figure 2.2: Predicting the motion of a vehicle. Results are given for a linearised model, the standard UT using $w_0 = 1 - n/3$, and a Monte Carlo simulation. The car, shown on a reduced scale in the figure, is assumed to be travelling at 90 km/h and performing a turn manoeuver. The samples from the Monte-Carlo simulation are included as grey dots to illustrate the shape of the true distribution.

In Fig. 2.4 we illustrate how the same methods perform for the inverse tangent function. Contrary to the sinusoids in the previous example, the function changes rapidly only near the distribution mean, which is particularly unfortunate for a linearised approximation. The result confirms our observation from the previous example; when the function is *not* polynomial, the sigma-point positions become more important.

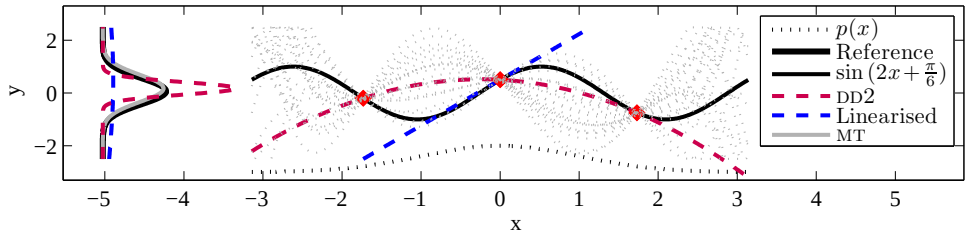
The MT is included in the evaluations to illustrate that the method can present good results for transformations which are typically difficult for other sigma-point methods to calculate correctly. Furthermore, it provides a tool for illustrating the underlying assumptions; the MT calculates a description of the functions that could have performed the transformation, and 50 random polynomials from this distribution are included in Fig. 2.3 and Fig. 2.4. Any of these functions (and infinitely many more) have the same mean as the estimates produced by the MT, UT and the DD2. See Paper I for more details, e.g., the meaning of the prior mentioned in the captions for figures 2.3–2.4.

Conclusion

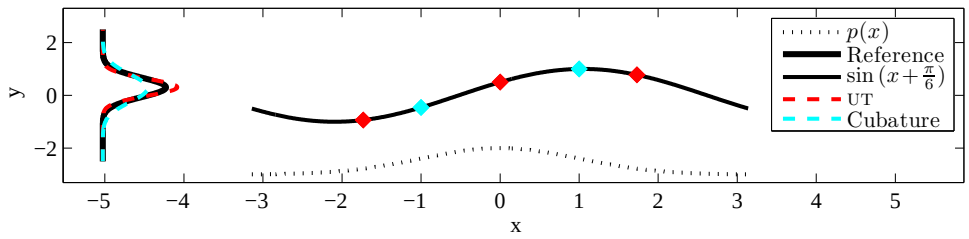
One of the arguments for the sigma-point approach has been that it is easier to approximate the probability distribution than the transforming function [30], [57], and our examples indicate that this is correct — even though the functions in



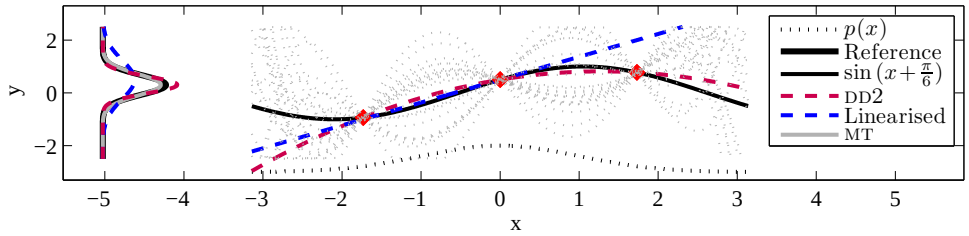
(a) Direct application of the UT and the cubature rule. The UT is overconfident whereas the cubature rule is fairly accurate, but with a small error in the estimate of the mean.



(b) Estimation using the linearised function, the MT, and the DD2 estimator, including 50 random polynomials generated from the MT posterior distribution. Contrary to the linearization, the second order approximation is overconfident.

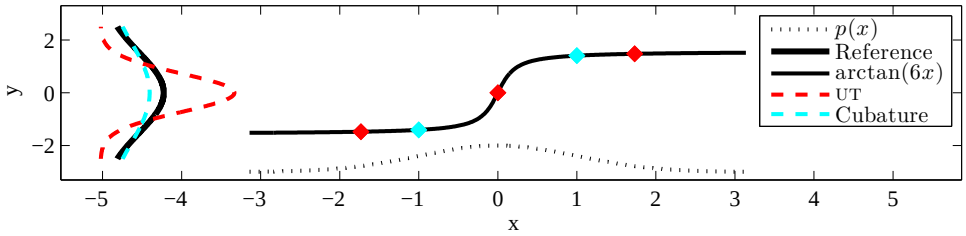


(c) Direct application of the UT and the cubature rule. The UT performs much better for this function than for the transformation illustrated in (a).

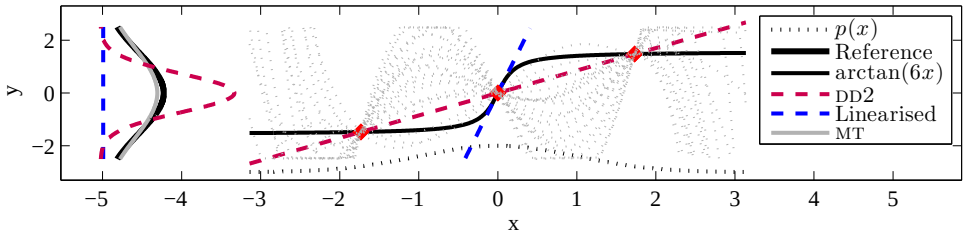


(d) Estimation using the linearised function, the MT, and the DD2 estimator, including 50 random polynomials generated from the MT posterior distribution.

Figure 2.3: Comparison of different approaches towards moment estimation. Gaussian approximations of $y = g(x)$, where $x \sim \mathcal{N}(0, 1)$, are shown along the y-axis. The UT, DD2 and the MT use the red sigma-points $(0, \pm\sqrt{3})$ and the cubature rule use the blue sigma-points (± 1). The MT diagonal prior is $[1 \frac{1}{100} \frac{1}{100} \frac{1}{100} \frac{1}{100}]$.



(a) Direct application of the UT and the cubature rule. The UT use $w_0 = 1 - n/3$, i.e., the three red sigma-points, whereas the cubature rule is using the two turquoise sigma-points.



(b) Model based estimation using the linearised function, the MT, and the dd2 estimator. The MT use the prior $(1\frac{1}{100}\frac{1}{100}\frac{1}{100}\frac{1}{100})$ and both sigma-point methods use the three red sigma-points, which for the dd2 corresponds to $h = \sqrt{3}$.

Figure 2.4: Comparison of different approaches towards moment estimation. Gaussian approximations of $y = \tan^{-1}(6x)$, where $x \sim \mathcal{N}(0, 1)$, are shown along the y-axis. The UT, DD2 and the MT use the red sigma-points $(0, \pm\sqrt{3})$ and the cubature rule use the blue sigma-points (± 1). The MT diagonal prior is $[1\ 1\frac{1}{10}\frac{1}{10}\frac{1}{10}]$.

Fig. 2.3a–2.3b and Fig. 2.4 are not well approximated by a low order polynomial, some methods approximate the mean and covariance very accurate. However, it is important to understand that even if the transforming function is not explicitly expressed by a sigma-point method, implicitly there exists a family of polynomials whose integrals correspond to the sigma-point approximation. In other words, approximating the transformation with any of these functions leads to the same result. Therefore, it is a very relevant question to ask whether or not this family is representative for the actual transformation.

Chapter 3

Tracking

In Chapter 2 we learned how to perform filtering on a dynamic system. When put to the task of following objects in the real world, using non-ideal sensors, several practical issues need to be taken care of before a filter can be applied. Examples of such tasks are *data association*, *track management*, *model evaluation*, *state parameterisation* and *timing issues*. In essence, a tracking system consists of a filter for updating the tracked objects, which are commonly denoted targets, and a set of methods responsible for solving these additional tasks. This chapter aims to clarify why the filter cannot be directly applied, and describes standard methods for solving the practical issues.

The system layout of the tracking system is presented in Section 3.1 and the challenges of target-to-measurement association are explained in Section 3.2, together with some useful association algorithms. A procedure for assessing the quality of the produced estimates is discussed in Section 3.3. In the multi-sensor setup often used by automotive safety systems, the tracking system performs *data fusion* to reliably estimate, e.g., positions and velocities of surrounding vehicles, which is discussed in Section 3.4. More details are available in Paper V and, e.g., textbooks [58] and [59].

3.1 The tracking system

The exact form of a tracking system depends on the filtering strategies and the methods used to solve the additional tasks discussed in the introduction, but in general most systems share a similar structure. The layout of a typical multi-sensor tracking system is illustrated in Figure 3.1. Given a new measurement, the posterior distribution from the previous iteration is predicted to the time for the received measurements and the data association procedure takes place (indicated by the dashed arrows). Then, tracks with a history of poor performance may be removed, whereas new tracks can be initiated to explain measurements that do not originate from known objects. Finally, tracks are updated with the new data and

are made available to other applications, in this case threat assessment algorithms. The prediction and the state update tasks are carried out by a filter as described in Chapter 2.

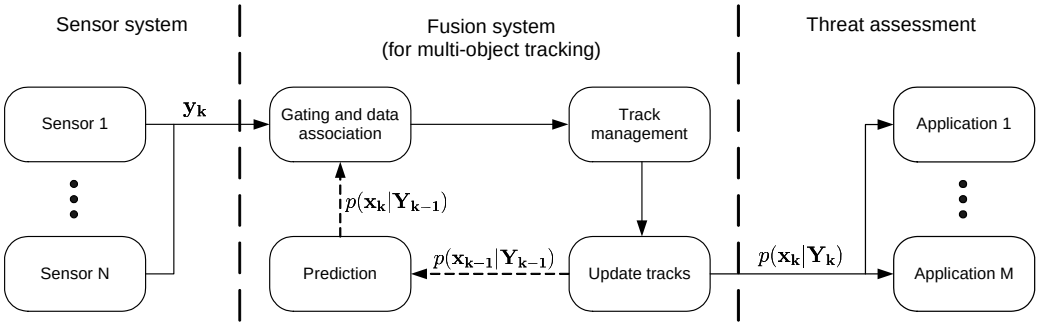


Figure 3.1: A general description of the tracking system. The solid arrows show how new data $\mathbf{y}_k$ is used to update the state vector $\mathbf{x}_{k-1}$.

3.1.1 System model

The system model provided in Section 2.1 describes a general state vector, $\mathbf{x}_k$, and measurement vector, $\mathbf{y}_k$. Assume that the tracking system receives observations on an unknown number of independently moving objects. The state vector can then be parameterised as

$$\mathbf{x}_k = \left[\mathbf{z}_k^{(1)}\mathbf{z}_k^{(2)}\dots\mathbf{z}_k^{(N_k)}\right], \tag{3.1}$$

where $\mathbf{z}_k^{(i)}$ is the state of the i^{th} track and N_k is the number of tracks believed to be present at time t_k. At every iteration, we receive M_k measurements

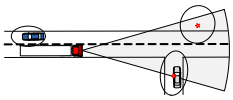
$$\mathbf{y}_k = \left[\boldsymbol{\gamma}_k^{(1)}, \boldsymbol{\gamma}_k^{(2)}, \dots, \boldsymbol{\gamma}_k^{(M_k)}\right], \tag{3.2}$$

which are used to update the state. If the sensor platform is moving, as in the case of an automotive safety application, the position of the platform is often included in the state vector as well — it can typically be tracked with high accuracy using onboard sensors such as gyros and speedometers.

3.1.2 Tracking procedure

The actions during a recursion of the tracking system are illustrated, at a high level, in the following example. Given the result of the previous iteration, $p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$, a general tracking algorithm performs the following steps:

1. Predicts the tracks to the time for the next measurement. In other words; calculate $p(\mathbf{z}_k^{(i)}|\mathbf{Y}_{k-1})$, for $i = 1 \dots N_{k-1}$.



2. Calculates the measurement distribution for each predicted track, i.e., calculate $p(\boldsymbol{\gamma}_k^{(j)}|\mathbf{Y}_{k-1})$ for $j = 1 \dots N_{k-1}$.

3. Associates measurements with tracks, meaning that we decide which $\boldsymbol{\gamma}_k^{(j)}$ should be used to update track $\mathbf{z}_{k|k-1}^{(i)}$. Gating, described in Section 3.2.1, is performed prior to association.

4. Starts track candidates for measurements not associated with existing tracks.

5. Deletes tracks, e.g., those which have not been observed for some time.

6. Updates each track which has an associated measurement to calculate the posterior distribution $p(\mathbf{z}_k^{(i)}|\mathbf{Y}_k)$.

Note that in general, $M_k \neq N_{k-1}$ and it is not until step 5 that N_k is known. These steps differ depending on the approaches towards the estimation task at hand. One example of a different strategy is the cardinalized probability hypothesis density filter [60], which does not attempt to track single objects individually; one of the consequences is that N_k is described by a probability distribution, which arguably is a more appropriate description.

3.2 Data association

In order to update a tracked object with new information, we must first decide which measurements to use. This task is called data association (DA) and is an important component in a tracking system [61]. When every track is associated with a measurement, any of the common methods described in Chapter 2 can be used. However, if a track is updated with a measurement that does not originate from the tracked object, we will obviously introduce errors. It is therefore important to employ a robust DA strategy. It should also be considered that a target does not necessarily give rise to a measurement, i.e., is detected, in every measurement cycle, and we denote the probability of detection by P_{D}.

The basic methods for measurement-to-track association presented in this section are based on the hypothesis that each target gives rise to at most one measurement, i.e., that at most one measurement should be associated to each target. We will discuss *nearest neighbour* algorithms and *probabilistic data association* algorithms, which are useful methods that illustrates the complexity of the problem. An alternative method, which performs very well, is the computationally complex *multiple hypothesis tracking* [62].

3.2.1 Gating

A simple way to reduce the complexity of the data association problem is to introduce a so-called gate for each track. This is used to exclude measurements that are highly unlikely to originate from the track. As shown in Chapter 2, an

inherent part of the tracking filter is to calculate the measurement distribution $p(\mathbf{y}_k|\mathbf{Y}_{k-1})$. The probability P_G that the target measurement will be inside a region $\mathcal{S}$ is given by

$$P_\mathrm{G} = \int_{\mathcal{S}} p(\mathbf{y}_k|\mathbf{Y}_{k-1}) d\mathbf{y}_k. \tag{3.3}$$

The region $\mathcal{S}$ constitutes our gate and we use only measurements $\mathbf{y}_k \in \mathcal{S}$.

Applying the gate can be very easy. If $p(\mathbf{y}_k|\mathbf{Y}_{k-1})$ is Gaussian, with mean $\mathbf{y}_{k|k-1}$ and covariance matrix $\mathbf{S}_{k|k-1}$, the square of the statistical distance

$$d^2(\mathbf{y}_k) = (\mathbf{y}_k - \mathbf{y}_{k|k-1})^T \mathbf{S}_{k|k-1}^{-1} (\mathbf{y}_k - \mathbf{y}_{k|k-1}) \tag{3.4}$$

is Chi-square distributed with m degrees of freedom, where m is the dimension of the measurement space: $d^2(\mathbf{y}_k) \sim \chi^2_m$. Gating then consists of a simple test:

$$d^2(\mathbf{y}_k) \gtrless \lambda. \tag{3.5}$$

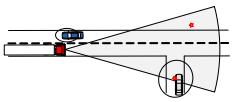
The gate size λ is calculated from (3.3) by solving $\int_0^\lambda p(\tau) d\tau = P_\mathrm{G}$, where $\tau \sim \chi^2_m$, with respect to λ for a desired probability P_G. Gating using (3.4)–(3.5) will result in ellipsoidal gates, as shown in Figure 3.2.

Clutter

It is possible for sensors to report measurements that do not originate from objects that should be tracked by the system. Such false returns are called *clutter* and could originate, e.g., from noise affecting the sensor detection algorithm or non-modelled objects visible to the sensor. Radar clutter in a traffic environment may originate from traffic signs, metal litter on the verge of the road, etc. We can attempt to calculate the expected number of false measurements inside a gate by modelling the clutter intensity. The gate size can then be determined based on the desired P_G, the actual P_D and the risk for including clutter. In other words, if the clutter intensity is non-zero we can determine how badly we want the correct measurement to fall inside the gate versus the risk of clutter being inside the gate as well.

Gating example

The data association task originating from the scenario in Figure 3.2 is made clear in Figure 3.3, where the global gating problem (3.3a) is shown from the perspective of track one (3.3b) and two (3.3c) respectively. Clearly, gating does not solve our problem completely because there are still multiple measurements inside the gates and one of the measurements appears in both. If the same measurement is used to update multiple tracks, the tracks may coalesce after some time unless this is taken into consideration.



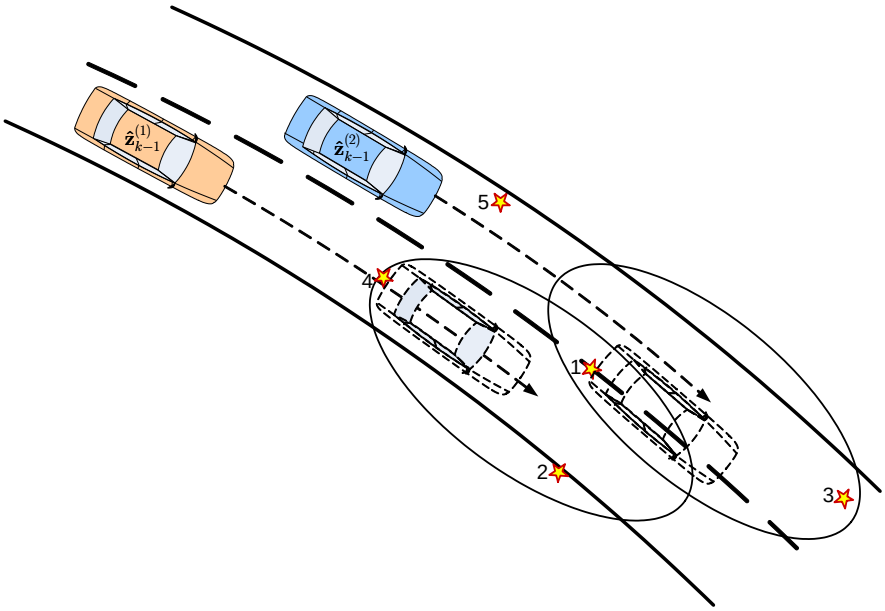


Figure 3.2: A prediction and data association scenario. The ellipsoidal gates are centred over the predicted positions of the two vehicles and the true positions are shown by the outlined cars, deviating from the predicted positions. The measurements are indicated by ($\star$), and the choice of DA method will significantly affect the result.

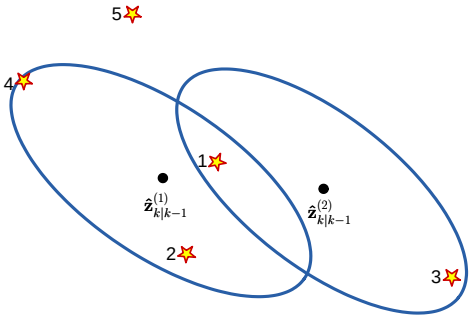
The scenario in Figure 3.2 is such that both vehicles deviate from the predicted position (the tip of respective arrows); the car in the right lane brakes to let the other car, which is also braking slightly, in to the lane. Given the true position of the cars, we conclude that measurement four probably originates from track one and measurement one probably originates from track two. In Sections 3.2.2–3.2.5 we discuss different methods to handle the association problem. These methods are applied to a DA task and the results are shown in Figure 3.4.

3.2.2 Nearest neighbour (NN)

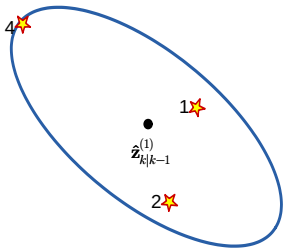
One simple and straightforward method for data association is to let each track use the best measurement. The best measurement is defined to be the one nearest to the predicted track position, motivating the name nearest neighbour, but there are several ways to measure distance. For example, we may choose the measurement that maximises the likelihood $p(\boldsymbol{\gamma}_k^{(j)}|\mathbf{z}_k^{(i)})$ for each track. In the case of Gaussian measurement noise, we pick the measurement $\boldsymbol{\gamma}_k^{(j)}$ for which

$$d_{i,j}^2 = [\boldsymbol{\gamma}_k^{(j)} - \hat{\boldsymbol{\gamma}}_{k|k-1}^{(i)}]^T {\mathbf{S}_{k|k-1}^{(i)}}^{-1} [\boldsymbol{\gamma}_k^{(j)} - \hat{\boldsymbol{\gamma}}_{k|k-1}^{(i)}] \quad \{\forall\, j : d_{i,j}^2 < \lambda\} \tag{3.6}$$

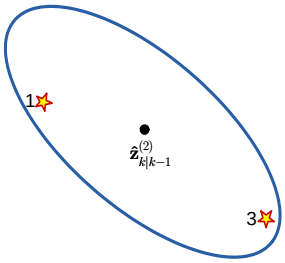
is minimised. Alternatives include minimising the Euclidian distance or choosing the measurement with highest signal strength, the latter also called the *strongest*



(a) Global association problem (gates are included for reasons of clarity).



(b) Measurements inside the gate for track one.



(c) Measurements inside the gate for track two.

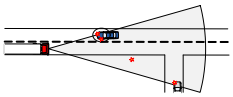
Figure 3.3: The global association problem (a), as seen from the local perspective of track one (b) and track two respectively (c). The predicted track positions are marked by •. Both tracks are competing for measurement one.

neighbour association scheme. The NN method has obvious flaws in that tracks may share measurements; two closely separated tracks are likely to merge and in high clutter environments tracks are likely to be associated with clutter instead of the true measurement. Nevertheless, if the circumstances are right, the NN method may be appropriate, motivated by its simplicity and the fact that each track can be updated individually. Recall from Section 2.2.1 that the residual is the difference between the measurement and the predicted measurement, $\mathbf{y}_k - \hat{\mathbf{y}}_k$. Figure 3.4a shows the measurement residuals $\tilde{\mathbf{y}}^{(1)}_k$ and $\tilde{\mathbf{y}}^{(2)}_k$ that would be used in, e.g., the Kalman filter, for the tracking situation presented in Figure 3.2.

3.2.3 Global nearest neighbour (GNN)

A major improvement to the NN approach is to constrain the association such that no tracks may share the same measurement, resulting in a global optimisation problem. Using the Gaussian noise example (3.6), the task is to minimise

$$\min_{\mathbf{j}} \sum_{i=1}^{N_{k-1}} d^2_{i,j} \quad \{\forall\, j : d^2_{i,j} < \lambda\}, \tag{3.7}$$

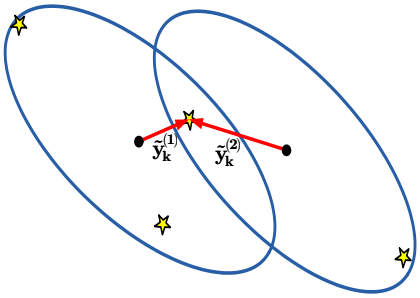


where $\mathbf{j} \in \mathbb{R}^{N_{k-1}}$ is the global association vector and each measurement may only be used once. The auction algorithm, e.g., the version presented in [63], is well suited to resolve the global assignment problem. In Figure 3.4b the resulting GNN measurement residuals $\tilde{\mathbf{y}}_k^{(1)}$ and $\tilde{\mathbf{y}}_k^{(2)}$ are shown for the tracking situation presented in Figure 3.2. This approach solves the problem of tracks sharing the same measurements, but will still perform poorly in cluttered environments.

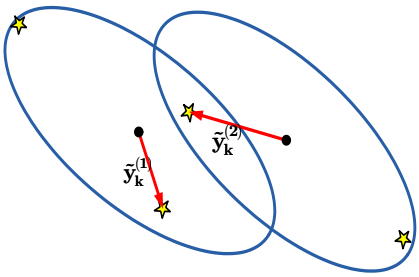
In order to prevent uncertain tracks from stealing measurements from good tracks, it is common to use the generalised statistical distance

$$\bar{d}_{i,j}^2 = d_{i,j}^2 + \ln(|\mathbf{S}_k^{(i)}|) \tag{3.8}$$

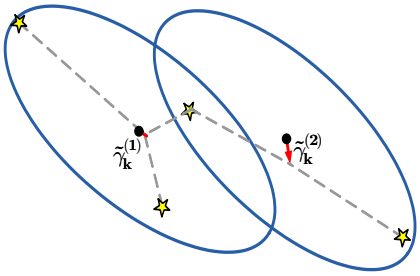
instead of $d_{i,j}^2$ in equation (3.7), i.e., maximise the likelihood $p(\mathbf{y}_k|\mathbf{Y}_{k-1})$.



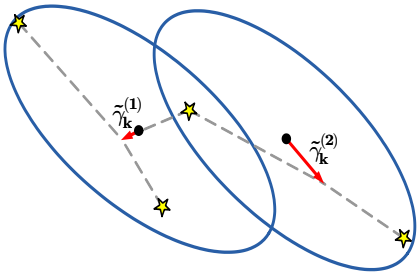
(a) NN association. Both tracks use the same measurement.



(b) GNN association. Track two gets the shared measurement.



(c) PDA association. The weighted residual for track one is almost zero.



(d) JPDA association. Compared to the PDA, weighted residuals are less influenced by the shared measurement.

Figure 3.4: A comparison of different data association methods. The measurement residual used to update each track is drawn with a black arrow. The following settings were used: $P_D = 0.9$, $P_G = 0.975$, and clutter intensity is approximated using $\beta = \frac{N_G}{V_G}$, where V_G is the gate volume.

3.2.4 Probabilistic data association (PDA)

The probabilistic data association approach was presented in [64] and acknowledges that the sources of the measurements are unknown. It is based on a Bayesian probability approach where each measurement is said to originate either from

the tracked object or from clutter. The latter is often modelled as a Poisson process whereas measurement noise is assumed to be Gaussian. If the number of measurements that fall inside the gate is N_G^i, a total of $N_G^i + 1$ association hypotheses can be formed for each target, including the event of a missed detection:

$$\mathcal{H}_0^i : \text{No measurement originates from object } i$$
$$\mathcal{H}_j^i : \text{Measurement } j \text{ originates from object } i$$

The probability for each hypothesis, $p_{ij} = \Pr(\mathcal{H}_j^i), j = 0 \dots N_G^i)$, derived in [58], is used to form a weighted sum of all residuals

$$\tilde{\boldsymbol{\gamma}}_k^i = \sum_{j=1}^{N_G^i} p_{ij} \tilde{\boldsymbol{\gamma}}_k^{(j)}. \tag{3.9}$$

The weighted measurement residual, $\tilde{\boldsymbol{\gamma}}_k^i$, can be used to update the i^{th} track using any of the Gaussian filters described in Chapter 2 (Section 2.2.1 and Section 2.3.1). Since clutter has been included in the state update, the posterior covariance matrix is adjusted to account for the increased uncertainty:

$$\boldsymbol{P}_k^i = p_{i0} \boldsymbol{P}_{k|k-1}^i + (1 - p_{0j}) \tilde{\boldsymbol{P}}_{k|k}^i + d\boldsymbol{P}_k^i, \tag{3.10}$$

where $\tilde{\boldsymbol{P}}_{k|k}^i$ is the posterior covariance given that there are no association uncertainties, $\boldsymbol{P}_{k|k-1}$ is the covariance of the predicted state, and

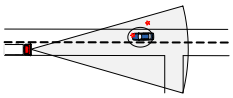
$$d\boldsymbol{P}_k^i \triangleq \mathbf{K}_k \left[\sum_{j=1}^{N_G^i} p_{ij} \tilde{\boldsymbol{\gamma}}_k^{(j)} (\tilde{\boldsymbol{\gamma}}_k^{(j)})^T - \tilde{\boldsymbol{\gamma}}_k^i \tilde{\boldsymbol{\gamma}}_k^{iT} \right] \mathbf{K}_k^T. \tag{3.11}$$

A PDA assignment results in the use of a measurement residual influenced by all gated measurements, shown in Figure 3.4c using the same example as earlier. Note that the PDA, similar to the NN, makes the simplifying assumption that other tracked objects do not generate measurements. In practice, the PDA will experience problems when the gating regions of two tracks overlap — the tracks tend to merge since they are updated using more or less the same observations.

3.2.5 Joint probabilistic data association (JPDA)

Contrary to the PDA approach, if the association probabilities are calculated jointly for all tracks and all gated observations we can consider the possibility that tracks yield measurements that fall into the gates of nearby tracks. The JPDA approach was introduced in [65], and the effect is that the influence on each track from measurements appearing in more than one gate, is reduced. This is illustrated in Fig. 3.4d, which should be compared to the PDA result, shown in Fig. 3.4c.

The association probabilities p_{ij} need to be recalculated to take the global effects into account. The expressions are available, e.g., in [58]. Calculating all hypotheses can be tough when several tracks share measurements and iterated versions of PDA have been suggested to approximate this solution, see, e.g., [59].



3.3 Track management

Previously, our goal has been to update the objects from the previous time instant with new data, but the number of objects has been assumed constant. We have also overlooked the question of track quality. We now direct our attention towards track initialisation and track removal, first treated in [61]. Extensive research on the topic is accounted for in [58], [59].

3.3.1 Sequential probability ratio test (SPRT)

Due to the DA uncertainties, it is possible for tracks to be updated with clutter, which obviously is not desirable. At least, if it happens, the receiver of the estimates should be made aware of it. Clearly it is not possible to know when this happens, so a statistical test is used instead. Two hypotheses that can be evaluated for each tracked object $i = 1 \dots M^k$ are introduced:

$$\mathcal{H}_1^i : \text{We are tracking a true object}$$
$$\mathcal{H}_0^i : \text{We are tracking clutter.}$$

The sequential probability ratio test, introduced in [66], can be used to create a test statistic, η^i, for comparing hypotheses such as these. The statistic is defined as the quotient between the respective hypothesis probabilities and, conditioned on the measurements $\mathbf{Y}_k$, we write

$$\eta_k = \frac{\Pr(\mathcal{H}_1^i|\mathbf{Y}_k)}{\Pr(\mathcal{H}_0^i|\mathbf{Y}_k)} = \frac{p(\mathbf{Y}_k|\mathcal{H}_1^i)\Pr(\mathcal{H}_1^i)}{p(\mathbf{Y}_k|\mathcal{H}_0^i)\Pr(\mathcal{H}_0^i)}. \tag{3.12}$$

A detector can then be designed to use this statistic for decision-making. Note also that the statistic can be recursively calculated in a tracking framework, as

$$\eta_k = \frac{p(\mathbf{y}_k|\mathbf{Y}_{k-1}, \mathcal{H}_1^i)}{p(\mathbf{y}_k|\mathbf{Y}_{k-1}, \mathcal{H}_0^i)}\eta_{k-1} \tag{3.13}$$

The recursive form is possible due to the Markov property of the process model, and that the measurement model is a function of the state vector. The logarithm of this statistic is often used in a tracking system as a quality measure of the track.

Track score

The so called track score $L^i(k)$ is defined as the logarithm of the hypotheses test statistic η_k^i, and we see from Eq. (3.13) that $L^i(k-1)$ can be recursively updated to include new data $\mathbf{y}_k$ by a simple addition:

$$L^i(k) = L^i(k-1) + \Delta L^i(k) \tag{3.14}$$

$$\Delta L^i(k) = \log \frac{p(\mathbf{y}_k|\mathbf{Y}_{k-1}, \mathcal{H}_1^i)}{p(\mathbf{y}_k|\mathbf{Y}_{k-1}, \mathcal{H}_0^i)} \tag{3.15}$$

The term $\Delta L^i(k)$, expressed in detail for various clutter models in [58], is related to how well each model explains the observed data. Thus, the score $L^i(k)$ can be used to determine the quality of the track, and as $\Delta L^i(k)$ can take both positive and negative values, the score will increase and decrease over time.

Track validation

Whenever measurements are not associated to an existing track a new object may have appeared, but it could also be clutter. A pragmatic approach is to initiate new candidate tracks for these measurements and use future data to determine the measurement origin. One commonly used approach is to use the track score $L(k)$ as an indicator; a new track is *validated* if the score exceeds a design threshold, or rejected if the score drops to low. The time spent in the so-called validation region, waiting for any of these thresholds to be crossed, varies depending on data.

A much simpler, yet sometimes appropriate, validation scheme is to use an m-out-of-n criterion: a track is validated when it has been associated with a measurement m times during the last n association rounds.

Track removal

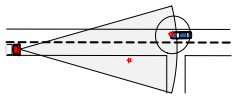
Naturally, just as new objects can enter the area observed by the sensors, they may leave. It makes little sense to keep track of an object that has left the field-of-view of the sensor, and one approach is to use the track score function also for this purpose. A common approach for preventing old tracks with a flawless record from becoming invincible, is to saturate $L(k)$ at a certain value. Other heuristic approaches include removing tracks that are too far away, tracks whose covariance matrices are to big, or tracks that have not been associated with data for some time, similar to the m-out-of-n criterion for track initialisation.

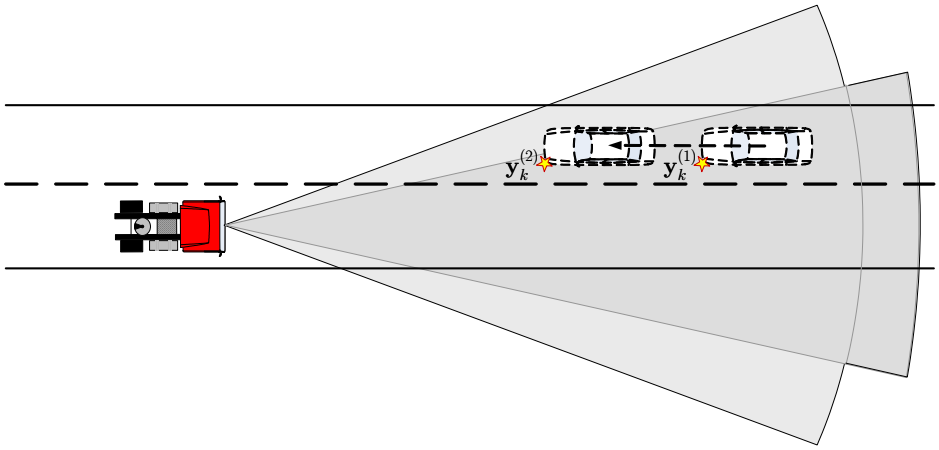
Model evaluation

Our estimation is based on models for dynamics and measurements, so the estimates are conditioned on the model choices; consider for example the Kalman filter — the posterior covariance matrix is not dependent on data, it is determined entirely by the models. The SPRT statistic can be used to decide which model is appropriate for a certain object after data has been observed, e.g., during the track validation period.

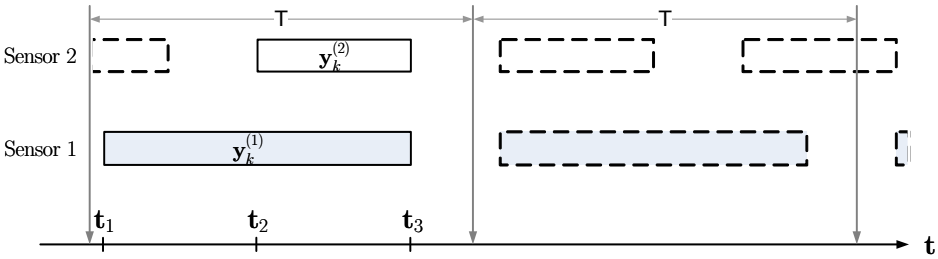
3.4 Data fusion

It is often beneficial to use a multi-sensor system in order to obtain multiple measurements from the tracked objects. Preferably, the sensors utilise different measurement principles in order to minimise technology-specific systematic errors and to benefit from the combination of their respective strengths. Multiple sensors





(a) Two measurements on the position of a *single* car are made at two different times. The car moves relatively far during the time between the first and the second measurement.



(b) Two sensors provide measurements to a tracking system. The length of a rectangle corresponds to time spent by a sensor of internal signal processing. Thus the left side of a rectangle corresponds to the time of the actual measurement (t_1, t_2), whereas the right (t_3) is when the tracking system can use the measurement.

Figure 3.5: Two sensors deliver data to the tracking system, but sensor one reports slightly older measurements than sensor two. The updated state description of the vehicle position will be poor if the delay is not accounted for.

can improve tracking results significantly yet also impose additional requirements on the system architecture, for example, on sample rate and time-stamping of measurements. Paper V presents a framework and a fusion approach in which several of the methods presented in this thesis have been included.

3.4.1 Measurement delays

Any potential benefits of a data fusion system can be ruined if we do not know, or cannot estimate, how old the data is when it reaches the tracking system. Of course that would be a problem in a single-sensor system as well, but if the delay is constant at least the estimates will be consistent. In a data fusion system where two sensors measure the same moving object at two different times, they will report two different positions. If they are used by the tracking system without knowledge

of the time difference, the effects will be similar to when gridlock errors[1] occur in distributed sensor systems. An example is given in Figure 3.5; two sensors collect measurements, but at different times, and due to delays they are available to the tracking system at the same time. If the tracking system cannot compensate for these delays, a fused track may perform worse than if only one sensor was used.

Example: *Two vehicles are driving at* 90 *km/h in opposite directions. A delay of* 50 *ms between the measurements results in an offset of* 2.5 *m.*

When an observation is delayed such that newer measurements have been used to perform a state update before the old measurement reaches the fusion system, it is called an *out-of-sequence* measurement. The recursive scheme cannot be directly applied to update the state using the old measurement, unless previous states and measurements are stored such that the filter can be re-run — this would be the optimal approach. In a multi-sensor multi-target tracking system, however, there is a limit to how much of the history can be saved and the optimal approach can only be used to a limited extent. Provided we are aware of the extent of the delay, an easy but unsatisfactory approach is to disregard the old data. This, of course, is a waste of information. A better approach is to do a "backwards prediction", i.e., a negative-time measurement update, such that the state can be updated at the time of the measurement, as shown in [67], [68], for linear systems.

3.5 Tracking example

An informal tracking example is included in this thesis, in the header area on pages 19–51. It can be viewed from here by putting a thumb on the 50^{th} page and letting one page go at the time, until the 18^{th} page has been reached; preferably in a pace of approximately 5 sheets per second. This should give an optical illusion of a radar-equipped truck approaching an intersection, while meeting a car in the oncoming lane. Another car drives out in front of the truck from the smaller road, such that the truck is forced to apply the brakes. The small dots are radar measurements and the ellipses are covariance contours of the state vector of the respective cars. Folding the title pages of Chapter 3 probably improves the experience.

[1] Gridlock errors occur when measurements originating from a sensor whose position is uncertain, are to be combined with other measurements.

Chapter 4

Modelling

A MODEL of a system is a useful tool; it enables reasoning regarding system properties and lays down the guiding principles for how the system should be treated. The model selection will be a critical design choice in any application. A sound strategy is to use a model that is as simple as possible, but still describes the behaviour that we are interested in with sufficient accuracy. Deciding what is "simple" and "sufficient" may of course be far from obvious. Numerous textbooks have been written on the topic, including [21], [69] to mention just a few, where the former discusses models for estimation and the latter teaches model design in general.

It is important to understand that model selection leads to assumptions not only regarding what properties we choose to include in our model, equally important is also what is *not* included.

Example: *A sensor measures the range, r, to an object and we wish to model the sensor output, $y = f(\cdot)$. A very simple model would be*

$$y = r. \tag{4.1}$$

In the example above we have stated that the sensor output depends only on the true range and that the sensor is totally accurate. However, we implicitly state that several other properties are also valid, e.g., there is no limit in how far the sensor can see, the object is a point in space, the object is always detected, the surrounding environment does not affect the measurements, etc. Most of these assumptions are usually not correct. In this section we will explain how to design models suitable in a tracking system, i.e., motion models and measurement models, and formally handle the effects of simplifications.

4.1 Stochastic model description

It is common to include a stochastic model parameter to explain inaccuracies when it is not possible to model a system with total accuracy [21], [58]. In other words,

we recognise that a model cannot describe the system in a deterministic sense. Instead we attempt to model the inaccuracies through an unknown parameter whose statistical properties are known, referred to as *noise*.

Example: *To extend the model from the previous example to explain inaccuracies in the range measurement, let*

$$y = r + e, \tag{4.2}$$

where e is a scalar stochastic variable with known statistic properties, for example $e \sim \mathcal{N}(\mu_e, \sigma_e^2)$. The statistical properties of the resulting sensor output are then described as,

$$y \sim \mathcal{N}(r + \mu_e, \sigma_e^2). \tag{4.3}$$

It is important to correctly model the statistical properties of noise parameters, as estimators are derived based on these assumptions. To obtain mathematically tractable models we often assume that noise enters the model as additive white Gaussian noise (WGN), unless there is strong evidence to the contrary [20]. This assumption is further justified by Theorem 1, which broadly speaking states that the sum of infinitely many random variables is normally distributed[1].

Theorem 1 (Central limit theorem)
Let $X_1, X_2, \ldots X_n$ be a sequence of independent identically distributed random variables with finite mean μ and non-zero variance, σ. Then, as $n \to \infty$

$$\tfrac{1}{\sqrt{N\sigma^2}} \sum_{i=1}^{n} (X_i - \mu) \xrightarrow{D} \mathcal{N}(0, 1),$$

meaning that the distribution function of the sum will converge to the distribution function of $\mathcal{N}(0, 1)$.

Proof of this theorem is given in [70].

Even though Theorem 1 indicates that noise is likely to be approximately WGN at some point, due to subsequent transformations, it is still possible for the observable variable to have a non-Gaussian distribution.

4.2 Models in the tracking framework

Chapter 2 discusses the formal solution towards filtering the system described in Section 2.1, and the filter recursion described by (2.5) contains the distributions

$$p(\mathbf{x}_k | \, \mathbf{x}_{k-1}) \tag{4.4}$$
$$p(\mathbf{y}_k | \, \mathbf{x}_k). \tag{4.5}$$

[1] It is in fact the distribution function that converges to that of a normal distribution.

In a tracking system the state vector $\mathbf{x}_k$ can be viewed as a snapshot of the tracked entities, e.g., the position of a vehicle, at time t_k. Similarly, when a measurement, $\mathbf{y}_k$, is obtained by a sensor, it is a snapshot of the tracked entities at time t_k, *as perceived by the sensor*. In terms of the distributions (4.4)–(4.5), we conclude that $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ describes how the state vector $\mathbf{x}_k$ changes as a function of time, and $p(\mathbf{y}_k|\mathbf{x}_k)$ describes how the sensor perceives the world for a given state vector $\mathbf{x}_k$. The two distributions are denoted *process model* and *measurement model*, also often referred to as the motion model and the sensor model. A commonly used formulation is on the form (2.1)–(2.2), i.e.,

$$\mathbf{x}_k = f_{k-1}(\mathbf{x}_{k-1}, \mathbf{v}_{k-1}) \tag{4.6}$$

$$\mathbf{y}_k = h_k(\mathbf{x}_k, \mathbf{w}_k), \tag{4.7}$$

where it is made clear that noise, $\mathbf{v}_{k-1}$ and $\mathbf{w}_k$ affects the motion. Developing models is the base of estimation and therefore also the foundation of a tracking framework. In this section we will look deeper into these models.

4.2.1 The motion model

The motion model, or process model, describes how the state vector changes over time. In tracking, the state vector typically contains physical entities such as the position of vehicles in the surrounding environment. A good survey on dynamic models for manoeuvering targets is [71]. The dynamics of a system can generally be written

$$\dot{\mathbf{x}}(t) = g(\mathbf{x}(t), \boldsymbol{\vartheta}(t)), \tag{4.8}$$

where the stochastic noise process $\boldsymbol{\vartheta}(t)$ is introduced to explain model inaccuracies and events that cannot be predicted. There is a balance between accuracy and model complexity; $\dot{\mathbf{x}}(t)$ is modelled as a function of the state and noise, so any parameter affecting the derivative should be included in one of these terms, which could grow to an arbitrary size. Typically, a parameter is included in the state vector if it is required for accuracy, and can be measured or accurately predicted.

Motion model example

We shall attempt to model the motion of a vehicle moving in a Cartesian coordinate system ξ and assume that $\ddot{\xi}(t)$ can be modelled as a stochastic noise process $\boldsymbol{\vartheta}(t)$. This motivates the state parameterisation

$$\mathbf{x}(t) = [\xi_x(t)\ \xi_y(t)\ \dot{\xi}_x(t)\ \dot{\xi}_y(t)]^T. \tag{4.9}$$

The relation (4.8) can be written as a set of linear differential equations

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\boldsymbol{\vartheta}(t), \tag{4.10}$$

where

$$\mathbf{A} = \begin{bmatrix} \mathbf{0}_{2x2} & \mathbf{I}_{2x2} \\ \mathbf{0}_{2x2} & \mathbf{0}_{2x2} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{0}_{2x2} \\ \mathbf{I}_{2x2} \end{bmatrix}, \quad \boldsymbol{\vartheta}(t) = \begin{bmatrix} \vartheta_x(t) \\ \vartheta_y(t) \end{bmatrix}. \tag{4.11}$$

If the process noise is white, i.e.

$$\mathrm{E}[\boldsymbol{\vartheta}(t)] = \mathbf{0} \tag{4.12}$$

$$\mathrm{E}[\boldsymbol{\vartheta}(t)\boldsymbol{\vartheta}(\tau)^T] = \mathbf{Q}_\vartheta \delta(t-\tau), \tag{4.13}$$

this model is called a continuous white noise acceleration (CWNA) model, commonly referred to as a *constant velocity* model.

Using the model for prediction

Let us consider the problem of estimating future values of the state vector $\mathbf{x}(t)$ given the current state $\mathbf{x}(t_0)$. The state will be affected by noise during the prediction interval, so the predicted state is a stochastic process. Using Eq. (4.8), the prediction can be written

$$\mathbf{x}(t) = \mathbf{x}(t_0) + \int_{t_0}^{t} g(\mathbf{x}(\tau), \boldsymbol{\vartheta}(\tau)) d\tau. \tag{4.14}$$

If we can form a linear time-invariant model such as (4.10), the solution takes the form [21]:

$$\mathbf{x}(t) = \mathbf{F}(t, t_0)\mathbf{x}(t_0) + \int_{t_0}^{t} \mathbf{F}(t, \tau)\mathbf{B}(\tau)\boldsymbol{\vartheta}(\tau) d\tau, \tag{4.15}$$

where

$$\mathbf{F}(t, t_0) = e^{\mathbf{A}(t-t_0)} \tag{4.16}$$

$$e^{\mathbf{A}t} = \sum_{k=0}^{\infty} \frac{1}{k!}(\mathbf{A}t)^k. \tag{4.17}$$

For time dependent systems, however, the state transition matrix $\mathbf{F}(t, t_0)$ may not have an explicit form.

Discretisation of a continuous motion model

We often need discrete-time models; practically all algorithms are implemented on a computer and measurements arrive only at discrete time instants. In other words, we wish to find an expression

$$\mathbf{x}(kT + T) = f(\mathbf{x}(kT), \tilde{\mathbf{v}}(kT)) \tag{4.18}$$

where T is the system sampling interval and $k \in \mathbb{N}$ a counter. Usually notation is simplified by writing $\mathbf{x}(kT)$ as $\mathbf{x}_k$. We immediately spot the resemblance with the prediction task, so the solution is given by equations (4.14)–(4.15). Solving the integral expression may be difficult, and simplifications and approximations may be necessary; it is common to assume constant noise in between samples. For the CWNA model example (4.10), where the state transition matrix takes the form $e^{\mathbf{A}(t-t_0)}$ we can derive the exact solution:

$$\mathbf{x}_{k+1} = e^{\mathbf{A}T}\mathbf{x}_k + \tilde{\mathbf{v}}_k. \tag{4.19}$$

The new discrete process noise parameter $\tilde{\mathbf{v}}(kT)$ has covariance matrix $\mathbf{Q}_{\tilde{v}}$, which relates to the intensity of the continuous noise process $\boldsymbol{\vartheta}(\mathrm{t})$ through the integral in (4.14):

$$\tilde{\mathbf{v}}_k = \int_0^T e^{\mathbf{A}(T-\tau)}\mathbf{B}\boldsymbol{\vartheta}(kT+\tau)d\tau, \tag{4.20}$$

where $\tilde{\mathbf{v}}(kT)$ is a zero-mean random variable with covariance matrix [21]:

$$\mathbf{Q}_{\tilde{v}} = \begin{bmatrix} \frac{1}{3}T^3 & 0 & \frac{1}{2}T^2 & 0 \\ 0 & \frac{1}{3}T^3 & 0 & \frac{1}{2}T^2 \\ \frac{1}{2}T^2 & 0 & T & 0 \\ 0 & \frac{1}{2}T^2 & 0 & T \end{bmatrix} \begin{bmatrix} \mathbf{Q}_\vartheta & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_\vartheta \end{bmatrix}, \tag{4.21}$$

i.e., a function of the power spectral density $\mathbf{Q}_\vartheta$ of the continuous time noise process.

Direct discretisation

To develop and discretise a continuous time motion model can be time consuming and it may be much more efficient to formulate a discrete motion model directly, shown here with a simple example — the discrete white noise acceleration (DWNA) model. It is very similar to the CWNA model shown in the examples above. Let the state vector be

$$\mathbf{x}_k = [\xi_{x_k}\ \xi_{y_k}\ \dot{\xi}_{x_k}\ \dot{\xi}_{y_k}]^T, \tag{4.22}$$

and assume $\dot{\xi}_{x_k}$, $\dot{\xi}_{y_k}$ to be white discrete-time random walk Gaussian processes driven by the noise $\mathbf{v}_k$. The dynamics are described by the difference equation

$$\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k + \boldsymbol{\Gamma}\mathbf{v}_k, \tag{4.23}$$

where

$$\mathbf{F} = \begin{bmatrix} \mathbf{I}_{2\mathrm{x}2} & \mathbf{I}_{2\mathrm{x}2}\cdot T \\ \mathbf{0}_{2\mathrm{x}2} & \mathbf{I}_{2\mathrm{x}2} \end{bmatrix}, \quad \boldsymbol{\Gamma} = \begin{bmatrix} \mathbf{I}_{2\mathrm{x}2}\cdot\frac{1}{2}T^2 \\ \mathbf{I}_{2\mathrm{x}2}\cdot T \end{bmatrix}, \quad \mathbf{v}_k = \begin{bmatrix} v_{x_k} \\ v_{y_k} \end{bmatrix}. \tag{4.24}$$

The unit of $\mathbf{v}_k$ is $\left[\frac{m}{s^2}\right]$ and the expression for Γ originates from the relation

$$\int_0^T v_{x_k} dt = T v_{x_k} \tag{4.25}$$

$$\int_0^T \int_0^T v_{x_k}\, dt\, d\tau = \frac{1}{2} T^2 v_{x_k}. \tag{4.26}$$

The difference between this representation and the discretised CWNA is the definition of the process noise, which now is assumed to be a sequence of random numbers. Note that the noise covariance matrix $\mathrm{E}[\mathbf{\Gamma} \mathbf{v}_k \mathbf{v}_k^T \mathbf{\Gamma}^T]$ does not have the same form as $\mathbf{Q}_{\tilde{v}}$ in (4.21):

$$\mathbf{\Gamma} \mathbf{Q}_\mathbf{v} \mathbf{\Gamma}^T = \begin{bmatrix} \frac{1}{2}T^2 & 0 \\ 0 & \frac{1}{2}T^2 \\ T & 0 \\ 0 & T \end{bmatrix} \begin{bmatrix} \mathbf{Q}_\mathbf{v} \end{bmatrix} \begin{bmatrix} \frac{1}{2}T^2 & 0 & T & 0 \\ 0 & \frac{1}{2}T^2 & 0 & T \end{bmatrix} \tag{4.27}$$

$$= \begin{bmatrix} \frac{1}{4}T^4 & 0 & \frac{1}{2}T^3 & 0 \\ 0 & \frac{1}{3}T^4 & 0 & \frac{1}{2}T^3 \\ \frac{1}{2}T^3 & 0 & T^2 & 0 \\ 0 & \frac{1}{2}T^3 & 0 & T^2 \end{bmatrix} \begin{bmatrix} \mathbf{Q}_\mathbf{v} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_\mathbf{v} \end{bmatrix}, \tag{4.28}$$

where $\mathrm{E}[\mathbf{v}_k \mathbf{v}_k^T = \mathbf{Q}_\mathbf{v}]$ and (4.28) is included so we can compare the form with (4.21). Similar to the CWNA these models are also often referred to as constant velocity models.

Modifications to the CWNA and the DWNA

The CWNA and DWNA models shown in the previous section can be altered into the so-called continuous wiener process acceleration (CWPA) model and discrete wiener process acceleration (DWPA) model by including the acceleration in the state vector. In the CWPA case, the resulting parameterisation is

$$\mathbf{x}(t) = [\xi_x(t)\ \xi_y(t)\ \dot{\xi}_x(t)\ \dot{\xi}_y(t)\ \ddot{\xi}_x(t)\ \ddot{\xi}_y(t)]^T \tag{4.29}$$

$$\begin{bmatrix} \dddot{\xi}_x(t) \\ \dddot{\xi}_y(t) \end{bmatrix} = \begin{bmatrix} \vartheta_x(t) \\ \vartheta_y(t) \end{bmatrix}, \tag{4.30}$$

where the unit of $\boldsymbol{\vartheta}(t)$ is $\left[\frac{m}{s^3}\right]$. In the case of the DWPA, noise is often modelled as an increment in acceleration, making it easier to choose the process noise covariance. These models are commonly denoted *constant acceleration* models and are widely used. Other popular models include the *Singer model* [72], the *coordinated turn* models, and *bicycle models* of various complexity, e.g., described in [4] and [73]. Textbooks [21], [58] and tutorial [71] present further details. A model assumption in these models, when applied to vehicle motion, is that driver input is modelled

as noise. One can think of several occasions when this is not a good assumption, especially when modelling the motion of vehicles. Generally a driver controls the vehicle in accordance with some plan or preferences, which would be beneficial to include in the model. Paper IV provides a framework for modelling such behaviour for improved predictions and situation assessment. The output is believed to be particularly useful in Paper II, where a framework for decision-making using a driver model is presented.

4.2.2 The measurement model

The purpose of the measurement model is to describe the measurement density $p(\mathbf{y}_k|\mathbf{x}_k)$. It describes a snapshot of the dynamic system, seen through the eyes of the sensor. We have concluded in Chapter 2 that it is a necessary component in calculating the posterior distribution, and it was shown in Chapter 3 that it can be used to determine the quality of the estimates.

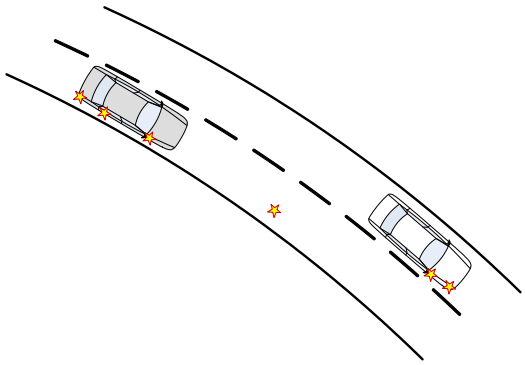


Figure 4.1: A sensor reports six detections marked in the figure by stars. Three measurements originate from the top left car, two from the bottom right car and one does not seem to originate from a vehicle at all. A sensor model should explain, in a statistical sense, where these measurements occur.

The sensor model has two main functions. It describes how the state space is perceived by the sensor, but also how the surrounding environment influences the measurements. We intuitively see how the former can enter the model as a mathematical relation, e.g. (4.31) if measurement noise $\mathbf{w}_k$ is additive. The example of polar-to-Cartesian coordinate transformation,

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} r\cos\varphi \\ r\sin\varphi \end{bmatrix} + \begin{bmatrix} w_x \\ w_y \end{bmatrix}, \tag{4.31}$$

is illustrated in Section 2.4.3. Examples of other descriptions provided by the sensor model are, for example, the operational range, sensor mounting position, expected number of detections per object, clutter sensitivity, etc. A typical scenario when tracking vehicles with a radar sensor is shown in Figure 4.1, illustrating

five measurements distributed near the two vehicles, and one measurement in the middle of the road. Some typical sensor model parameters and sensor model dependent statistics are the following:

P_{D}	:	Probability of detection ($P_{\mathrm{D}} = 0$ outside the detection region).
$1 - P_{\mathrm{D}}$	:	The probability that an object is not detected even though it is within the detection range.
λ^c	:	The clutter intensity
Δ_d	:	A resolution cell, in which two or more different targets cannot be separated.
$p(\mathbf{y}_k \vert \mathbf{x}_k, \lambda^c, P_{\mathrm{D}} = 1)$	:	The measurement distribution for a detected target.

Design of a sensor model depends heavily on the underlying sensing principle and hardware design, a more detailed model is generally associated with a specific sensor and will not be further discussed here. Examples of common sensor technologies are *radar*, *laser based ranging*, *infrared*, *acoustic*, and *image* sensors, whose characteristics from a tracking perspective are studied e.g. in [58]. A more detailed survey regarding radar systems modelling for tracking is given in [59] and one example of how to track distributed targets, such as the cars in Figure 4.1, using a more detailed radar sensor model is given in Paper III.

Chapter 5

Contributions and future work

While the previous chapters provide a background to the problems discussed in the contributions in Part II, this chapter presents a short summary of the contributions in the appended papers and elaborate on possible extensions and future work.

5.1 Contributions

The author's work has mainly focused on recursive estimation, or filtering. More specifically, modelling radar measurements, prediction models including the driver input, and methods for estimating the mean and covariance of random variables have been in focus — a trio which comes in handy when designing a filtering framework; much like the three musketeers in a sword-fight. The efforts have resulted in a derivative-free filter for recursive estimation, and a framework for motion modelling where the control input from the driver is included. Further, a sensor model that explains the presence of multiple detections from a single object has been developed and incorporated in a tracking framework. The goal has been to describe the traffic environment such that active safety applications can utilise the description in making intervention decisions. A probabilistic framework is well suited for such a description and allows for a natural extension of the uncertainty representation into the decision layer, see Fig. 1.3. The work has therefore come to include some aspects of decision-making and driver modelling too, albeit not covering actual threat assessment. Rather, the proposed framework for decision-making is intended for usage with any set of threat assessment algorithms. Additionally, when designing a complete tracking system, several complex design choices must be analysed and practical issues that arise must be coped with. The work in this particular field has been characterised by the desire to create a modular development platform that is suitable for continuous development.

The work has been supported by the Swedish intelligent vehicle safety systems (IVSS) programme and the strategic vehicle research and innovation programme

(FFI), both of which are funded by the Swedish governmental agency for innovation systems (VINNOVA).

Paper I: Moment Estimation Using a Marginalized Transform

So-called sigma-point methods are often applied to the task of estimating the mean and covariance of transformed random variables in Gaussian filtering frameworks, due to their simplicity and documented performance. Although the transforming function is typically known, the integral expressions for mean and covariance are not. In this paper, we propose to use evaluations of the transforming function to describe a family of functions which could have performed the observed evaluations. The family is selected such that expressions for mean and covariance are known; estimates of the mean and covariance are then derived by marginalising the function from the analytical expressions. The information provided by the evaluations is used in a Bayesian framework, and the resulting estimation algorithm performs well, especially in estimating the covariance matrix. Contrary to the unscented transform, the resulting approximation of the covariance matrix is guaranteed to be positive-semidefinite.

The author has contributed to a majority of this work.

Paper II: A Probabilistic Framework for Decision-Making in Collision Avoidance Systems

When designing intervention rules intended to avoid or mitigate the consequences of an accident, our understanding of the measurement and prediction uncertainties should be taken into consideration. This paper describes how the posterior description of the traffic situation, provided by a tracking system, can be used together with prediction models to calculate a probabilistic description of possible threats. We propose a decision-making framework which not only formally treats these uncertainties, but also includes a model of the driver acceptance of interventions. The reason is that, when aiming at collision avoidance, it is hard to define objective performance measurements — how does one express the condition for when an intervention is too early? Such decisions are subjective; in some situations it might be appropriate with high margins, whereas in other situations this is not accepted by the drivers at all. In this paper we include a driver model in the decision-framework in order to assess whether an intervention would be considered justified or not by the driver. It is shown that this approach can lead to earlier interventions in certain scenarios, rendering a higher safety benefit while suppressing interventions that the driver might not have appreciated. The example implementation uses a driver model expressed in terms of the driver conception of the world, and a previously known threat assessment algorithm.

This paper is a refinement of the ideas initially presented in [2], where the author and Mattias Brännström jointly explored the no man's land between their respective expertise (filtering and threat assessment). The author's contribution

has therefore mainly concerned using statistical methods to describe the output from threat assessment algorithms, the formal treatment of the associated uncertainties in the decision-making process, and the inclusion and evaluation of a driver model therein.

Paper III: Extended object tracking using a radar resolution model

Vehicles can be so-called distributed targets, meaning that they may yield multiple radar detections in a single scan. The result is a difficult data association problem that requires a detailed sensor model. The benefit of such measurements is that information such as object size and orientation can be extracted. In this paper we show how vehicles can be tracked using radar data containing multiple detections from each vehicle, given a reflector-based sensor model and an improved data association scheme.

This work has been carried out in close co-operation between the four authors of the paper, where all contributed to a similar extent. The main contributions of the author include designing a filter capable of using the reflector model, analysis and acquisition of radar data, and validation and design of the reference filter.

Paper IV: A new vehicle motion model for improved predictions and situation assessment

In this paper we present a framework for designing motion models which take the expected control input from the driver into consideration. A methodology for formal treatment of uncertainties regarding driver preferences and driving style is presented, which makes such models suitable for use in a tracking system. The ideas were originally introduced in [74], and were further developed in [6].

This work has been carried out in close co-operation between the four authors of the paper, where all contributed to a similar extent. The author's work has particularly concerned the modelling of driver-specific parameters and the marginalisation of these parameters from the state estimate.

Paper V: A design architecture for sensor data fusion systems with application to automotive safety

When designing a tracking system, there are many requirements to consider; not only must the output meet the requirements of the decision layer, the system must also be adapted for real-time implementation in a vehicle. Furthermore, the design should facilitate continuous development of the platform, such that future systems benefit from old and new experiences. This document describes how a tracking system for real-time usage in a vehicle can be designed and implemented to meet these needs, and contains descriptions and advice regarding methods and components of a tracking system. A tracking system using this design has been used in the evaluations of Paper III and Paper II.

This work has been carried out in close co-operation, and both authors have been involved in all of the design choices.

5.2 Future work

Several interesting ideas for future research were born during this research; some are necessary in order to make use of the presented methods in a real-time system in a vehicle, whereas other ideas intend to improve the benefit of active safety systems as a whole. Perhaps the most interesting ideas are the following:

Extending the marginalised transform framework

The MT framework shows promising results and there are several directions in which the method could be developed. An interesting thought is to use a different set of base functions than the Hermite polynomials, which are used in Paper I. This could be useful when models are not well approximated as polynomials, but it also invites to density filtering. This is related to a filtering approach that has appeared recently in the literature, where the aim is to calculate the probability density functions directly, rather than using the Gaussian filter framework.

Another interesting aspect is to use an adaptive prior for the MT, such that the description of the function evolves over time and over the state space, when applied in the recursive filtering framework. This could improve accuracy and make it possible to use fewer sigma points, thereby improving real-time performance.

Modelling the sigma-points as noisy observations opens up the possibility to include previously propagated sigma-points into the estimation, without constraining the approximation to pass through these points. The approach can be useful in situations where it is computationally expensive to evaluate the sigma-points, such as solving an optimisation problem.

The Gaussian filter in its basic form does not use the most recent measurement in the calculation of the covariance matrix estimate. Some methods, such as the iterated EKF, perform the state update iteratively to improve results using the latest measurement. To use the marginalised Kalman filter in a similar approach would be interesting, as it may reduce the influence of a poorly placed sigma-point on the final estimate.

Including new sensor-families in the fusion framework

The inclusion of driver-state sensors, map databases and vehicle-to-vehicle communication systems into the tracking system is an intuitive step, considering the usefulness of such information in threat assessment. However, this would require a different state-space representation than what we have used so far, and requires new sensor models. It is not clear if this approach is beneficial compared with other methods for including such information. Developing a sensor model for a driver-state sensor, for example, could be an intriguing challenge to start with.

Validation of driver models for decision-making

The evaluation of the decision framework proposed in Paper II makes use of a driver model developed based on assumptions regarding drivers' conception of the traffic situation. Although the model is supported by studies on how the human perception works, a model designed for use in a production-ready system needs to be adjusted and verified using actual drivers.

Extending the real-time system

The system presented in Paper V needs to be continuously developed in order to meet the demands of future in-vehicle safety systems. New sensors require not only models of their measurements; if the sensor can observe objects which previously could not be detected, new target models are also required. Track management methods and self-assessment procedures need to be developed jointly with threat assessment algorithms in order to develop relevant confidence measures. Robust strategies for fusion of data on different levels of refinement are required, as some sensors deliver filtered tracks, whereas others provide less processed measurements. Furthermore, a carefully prepared development platform facilitates the transfer of recent advances into vehicle programmes — this is a design challenge in itself.

Bibliography

[1] F. Sandblom and L. Svensson, "Marginalized sigma-point filtering," in *Proc. 14th Int. Conference on Information Fusion*, Chicago, USA, July 2011, pp. 1–8.

[2] F. Sandblom and M. Brännström, "Probabilistic threat assessment and driver modeling in collision avoidance systems," in *Intelligent Vehicles Symposium, 2011 IEEE*, Baden-Baden, Germany, June 2011, pp. 914 –919.

[3] M. Ahrholdt, F. Sandblom, L. Danielsson, and C. Lundquist, "SEFS results on sensor data fusion system development," in *ITS World Congress*, Stockholm, Sweden, September 2009.

[4] F. Bengtsson and L. Danielsson, "Designing a real time sensor data fusion system with application to automotive safety," Signals and Systems, Chalmers University of Technology, Göteborg, Tech. Rep. R007/2008, April 2008.

[5] J. Gunnarsson, L. Svensson, L. Danielsson, and F. Bengtsson, "Tracking vehicles using radar detections," in *Intelligent Vehicles Symposium, 2007 IEEE*, Istanbul, Turkey, June 2007, pp. 296 –302.

[6] J. Gunnarsson, L. Svensson, F. Bengtsson, and L. Danielsson, "Joint driver intention classification and tracking of vehicles," in *Nonlinear Statistical Signal Processing Workshop 2006*, Cambridge, UK, September 2006.

[7] C. Mathers and D. Loncar, "Updated projections of global mortality and burden of disease, 2002-2030: data sources, methods and results," *Geneva, World Health Organization*, 2005.

[8] M. Peden *et al.*, "World report on road traffic injury prevention: summary," 2004, ISBN: 92-4-159131-5.

[9] G. Carlsson, "Fatalities in Swedish road traffic - an overview," September 2002.

[10] Gothenburg & Lyon ARTs, "Volvo 3P European Accident Research Safety Report," Volvo 3P, Tech. Rep. ER-624264, 2011.

[11] "Regulation (EC) No 661/2009 of the European Parliament and of the Council of 13 July 2009 concerning type-approval requirements for the general safety of motor vehicles, their trailers and systems, components and separate technical units intended therefor," *Official Journal of the European Union*, pp. 1–24, 2009. [Online]. Available: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32009R0661:EN:NOT

[12] A. Kullgren, "Dose-response models and EDR data for assessment of injury risk and effectiveness of safety systems," in *Proc of Int. IRCOBI Conf.*, Bern, Switzerland, February 2008, pp. 3–14.

[13] M. Richter, H.-C. Pape, D. Otte, and C. Krettek, "Improvements in passive car safety led to decreased injury severity-a comparison between the 1970s and 1990s." *Injury*, vol. 36, no. 4, pp. 484–8, 2005.

[14] M. Distner, M. Bengtsson, T. Broberg, and L. Jakobsson, "City safety–a system addressing rear-end collisions at low speeds," in *Proc. 21th Enhanced Safety Vehicles Conf.*, paper 09-0371, 2009.

[15] A. Polychronopoulos and A. Amditis, "Revisiting JDL model for automotive safety applications: the PF2 functional model," in *Proc. 9th Int. Conference on Information Fusion*, July 2006, pp. 1 –7.

[16] M. Brännström, E. Coelingh, and J. Sjöberg, "Model-based threat assessment for avoiding arbitrary vehicle collisions," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 658–669, Sep. 2010.

[17] R. Munroe, "Conditional risk," (do not miss the tooltip). [Online]. Available: http://xkcd.com/795/

[18] A. Anund, G. Kecklund, and B. Peters, "Min trötta resa," VTI, Tech. Rep., 2004.

[19] V. L. Neale, T. A. Dingus, S. G. Klauer, and M. Goodman, "An overview of the 100-car naturalistic study and findings," *Traffic Safety*, pp. 1–10, 2005.

[20] S. M. Kay, *Fundamentals of statistical signal processing - estimation theory.* Prentice-Hall, Inc., 1993.

[21] Y. Bar-Shalom, X. Rong Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation.* John Wiley & Sons, Inc., 2001.

[22] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman filter: particle filters for tracking applications.* Artech House, 2004.

[23] X.-R. Li, "A survey of maneuvering target tracking: approximation techniques for nonlinear filtering," *Proceedings of SPIE*, vol. 5428, no. April, pp. 537–550, 2004.

[24] H. Lütkepohl, *Handbook of matrices*. Wiley, 1996.

[25] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. of the ASME – Journal of Basic Eng.*, vol. 82, no. Series D, pp. 35–45, 1960.

[26] P. Tichavský, C. Muravchik, and A. Nehorai, "Posterior Cramér-Rao bounds for discrete-time nonlinear filtering," *IEEE Trans. Signal Processing*, vol. 46, no. 5, pp. 1386 –1396, may 1998.

[27] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis, Second Edition*, 2nd ed. Chapman and Hall/CRC, Jul. 2003, ch. 2.9.

[28] K. Ito and K. Xiong, "Gaussian filters for non linear filtering problems," *IEEE Trans. Automatic control*, vol. 45, pp. 910–927, 2000.

[29] S. J. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte, "A new approach for filtering nonlinear systems," in *Proc. American Control Conference*, vol. 3, 1995, pp. 1628–1632 vol.3.

[30] S. Julier and J. Uhlmann, "Unscented filtering and nonlinear estimation," *Proc. IEEE*, vol. 92, no. 3, pp. 401 – 22, 2004.

[31] M. Norgaard, N. K. Poulsen, and O. Ravn, "New developments in state estimation for nonlinear systems," *Automatica*, vol. 36, no. 11, pp. 1627–38, 2000.

[32] I. Arasaratnam, S. Haykin, and R. J. Elliott, "Discrete-time nonlinear filtering algorithms using Gauss-Hermite quadrature," *Proc. IEEE*, vol. 95, no. 5, pp. 953–977, 2007.

[33] I. Arasaratnam and S. Haykin, "Cubature Kalman filters," *IEEE Trans, Autom. Control*, vol. 54, no. 6, pp. 1254–1269, 2009.

[34] Y. Wu, D. Hu, M. Wu, and X. Hu, "A numerical-integration perspective on Gaussian filters," *IEEE Trans. Signal Processing*, vol. 54, no. 8, pp. 2910–21, 2006.

[35] W. F. Denham and S. Pines, "Sequential estimation when measurement function nonlinearity is comparable to measurement error," *AIAA Journal*, vol. 4, no. 6, pp. 1071–1076, 1966.

[36] M. Athans, R. Wishner, and A. Bertolini, "Suboptimal state estimation for continuous-time nonlinear systems from discrete noisy measurements," *IEEE Trans. Autom. Control*, vol. 13, no. 5, pp. 504–514, Oct 1968.

[37] S. Holmes, G. Klein, and D. Murray, "An $O(N^2)$ square root unscented Kalman filter for visual simultaneous localization and mapping," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 7, pp. 1251 –1263, July 2009.

[38] S. Julier, "The scaled unscented transformation," *Proc. American Control Conference*, vol. 6, no. 2, pp. 4555–4559, 2002.

[39] R. Van der Merwe and E. Wan, "The square-root unscented Kalman filter for state and parameter-estimation," in *Proc. 2001 Conf. on Acoustics, Speech, and Signal Processing*, vol. 6, 2001, pp. 3461–3464.

[40] I. Arasaratnam and S. Haykin, "Square-root quadrature Kalman filtering," *IEEE Trans. Signal Processing*, vol. 56, no. 6, pp. 2589–2593, June 2008.

[41] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. pp. 97–109, 1970.

[42] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. New York: Springer Verlag, 2001.

[43] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statist. Comput.*, vol. 10, no. 3, pp. 197–208, 2000.

[44] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 174 – 88, 2002.

[45] S. Saha, Y. Boers, H. Driessen, P. Mandal, and A. Bagchi, "Particle based map state estimation: A comparison," in *Proc. 12th Int. Conference on Information Fusion*, July 2009, pp. 278 –283.

[46] N. Gordon, D. Salmond, and A. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proceedings F (Radar and Signal Processing)*, vol. 140, no. 2, pp. 107 – 13, 1993.

[47] A. Kong, J. Liu, and W. Wong, "Sequential imputations and Bayesian missing data problems," *Journal of the American Statistical Association*, vol. 89, no. 425, pp. 278–288, 1994.

[48] T. Schon, F. Gustafsson, and P.-J. Nordlund, "Marginalized particle filters for mixed linear/nonlinear state-space models," *IEEE Trans. Signal Processing*, vol. 53, no. 7, pp. 2279–2289, July 2005.

[49] R. V. D. Merwe, "Sigma-point Kalman filters for probabilistic inference in dynamic state-space models," Ph.D. dissertation, Oregon Health & Science University, 2004.

[50] G. H. Golub, "Some modified matrix eigenvalue problems," *SIAM Rev.*, vol. 15, no. 2, p. 318, 1973.

[51] J. Uhlmann, "Dynamic map building and localization: new theoretical foundations," Ph.D. dissertation, Univ. Oxford, Oxford, U.K., 1995.

[52] U. Lerner, "Hybrid Bayesian Networks for Reasoning About Complex Systems," Ph.D. dissertation, Stanford University, 2002.

[53] J. McNamee and F. Stenger, "Construction of fully symmetric numerical integration formulas," *Numerische Mathematik*, vol. 10, no. 4, pp. 327–344, 1967.

[54] L. de Menezes, A. Soares, F. Silva, M. Terada, and D. Correia, "A new procedure for assessing the sensitivity of antennas using the unscented transform," *IEEE Trans. Antennas Propag.*, vol. 58, no. 3, pp. 988 –993, March 2010.

[55] C.-L. Su, "Probabilistic load-flow computation using point estimate method," *IEEE Trans. Power Syst.*, vol. 20, no. 4, pp. 1843 – 1851, Nov. 2005.

[56] G. Steiner, H. Zangl, and D. Watzenig, "Generic statistical circuit design based on the unscented transformation and its application to capacitive sensor instrumentation," in *IEEE Int. Conf. on Ind. Technology*, Dec. 2005, pp. 108 –113.

[57] J. Uhlmann, "Simultaneous map building and localization for real time applications," Transfer thesis, Univ. Oxford, Oxford, U.K., 1994.

[58] S. Blackman and R. Popoli, *Design and analysis of modern tracking systems.* Artech House, 1999.

[59] Y. Bar-Shalom and W. Blair, *Multitarget-Multisensor Tracking Volume III: Applications and Advances.* Artech House, 2000.

[60] R. Mahler, "PHD filters of higher order in target number," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 43, no. 4, pp. 1523–1543, Oct. 2007.

[61] R. Sittler, "An optimal data association problem in surveillance theory," *IEEE Transactions on Military Electronics*, vol. MIL-8, no. 2, pp. 125–139, 1964.

[62] S. Blackman, "Multiple hypothesis tracking for multiple target tracking," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 19, no. 1, pp. 5–18, Jan 2004.

[63] D. Bertsekas and D. Castanon, "A forward/reverse auction algorithm for asymmetric assignment problems," *Computational Optimization and Applications*, vol. 1, no. 3, pp. 277 – 97, 1992.

[64] Y. Bar-Shalom and E. Tse, "Tracking in a cluttered environment with probabilistic data association," *Automatica*, vol. 11, September 1975.

[65] T. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Sonar tracking of multiple targets using joint probabilistic data association," *IEEE Journal of Oceanic Engineering*, vol. 8, July 1983.

[66] A. Wald, *Sequential Analysis*, 1st ed. John Wiley and Sons, 1947.

[67] Y. Bar-Shalom, "Update with out-of-sequence measurements in tracking: exact solution," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 38, no. 3, pp. 769–777, Jul 2002.

[68] Y. Bar-Shalom, H. Chen, and M. Mallick, "One-step solution for the multistep out-of-sequence-measurement problem in tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 40, no. 1, pp. 27–37, Jan 2004.

[69] L. Ljung and T. Glad, *Modellbygge och simulering.* Studentlitteratur, 1991.

[70] G. Grimmett and D. Stirzaker, *Probability and Random Processes.* Oxford University press, 2001.

[71] X. R. Li and V. Jilkov, "Survey of maneuvering target tracking. Part I: Dynamic models," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 4, pp. 1333 –1364, 2003.

[72] R. Singer, "Estimating optimal tracking filter performance for manned maneuvering targets," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-6, no. 4, pp. 473 –483, July 1970.

[73] M. Bühren and B. Yang, "A global motion model for target tracking in automotive applications," in *Proc. 2007 Conf. on Acoustics, Speech, and Signal Processing*, vol. 2, april 2007, pp. II–313–316.

[74] L. Svensson and J. Gunnarsson, "A new motion model for tracking of vehicles," *Proc. 14th IFAC Symposium on System Identification*, 2006.

Part II
Publications

Paper I

Moment estimation using a marginalized transform

F. Sandblom and L. Svensson

Submitted to *IEEE Transactions on Signal Processing*

Moment Estimation Using a Marginalized Transform

Fredrik Sandblom, Lennart Svensson

Abstract

We present a method for estimating mean and covariance of a transformed Gaussian random variable. The method is based on evaluations of the transforming function and resembles the unscented transform and Gauss-Hermite integration in that respect. The information provided by the evaluations is used in a Bayesian framework to form a posterior description of the parameters in a model of the transforming function. Estimates are then derived by marginalizing these parameters from the analytical expression of the mean and covariance. An estimation algorithm, based on the assumption that the transforming function can be described using Hermite polynomials, is presented and applied to the non-linear filtering problem. The resulting marginalized transform (MT) estimator is compared to the cubature rule, the unscented transform and the divided difference estimator. The evaluations show that the presented method performs better than these methods, more specifically in estimating the covariance matrix. Contrary to the unscented transform, the resulting approximation of the covariance matrix is guaranteed to be positive-semidefinite.

1 Introduction

Calculating the mean and covariance of stochastic variables is central to many estimation tasks, including, e.g., sensitivity analysis, which can be applied to a variety of systems including antenna characterization [1], power system analysis [2] and circuit design [3]. It is also frequently an essential component in recursive state estimation where the posterior mean and covariance often are used to characterize the distribution [4,5]. The importance of this task, with applications ranging from surveillance to medicine, have motivated a large part of recent research within the area of moment estimation [5], [6], [7], [8], [9], [10].

The general Bayesian solution to the state estimation problem involves integration of probability density functions —integrals which are rarely mathematically tractable. The family of *Gaussian filters* solves the recursive estimation problem under the assumption that the concerned distributions are approximately Gaussian. The equations used to compute the posterior mean and covariance under this assumption are those of the linear minimum mean square error (LMMSE) estimator, which coincides with the well known Kalman filter for linear systems [4].

A variety of Gaussian filters have been proposed to cope with non-linear models [5], and the derivative-free filters [11], [6], [7], [8], [9] are particularly useful; with

little or no adjustment, they can be applied to a wide range of problems. These filters use a transformed set of deterministically chosen points, often referred to as *sigma-points*, to approximate the mean and covariance. Arguably, the most well-known sigma-point method is the unscented transform (UT) [7], [11], that has been shown [12] to realize the fully symmetric integration formula presented in [13], which is exact for integration over certain polynomial functions. The (second order) divided difference filter (DD2) [6] calculates the mean and covariance matrix jointly, and both estimates are exact for a certain family of second order polynomials. An extensive analysis of the numerical integration perspective on Gaussian filters is given in [14].

Although easy to apply, derivative-free filters are not problem-free. The UT covariance matrix estimate is sometimes calculated such that it is not necessarily positive-semidefinite. This behavior was overcome with the recent introduction of the cubature integration rule [9], a special case of the UT, whose covariance matrix estimates are guaranteed to be non-negative definite. It performs well compared to methods of similar complexity [9], [15], [16], but unfortunately, the robustness comes at the expense of using a less accurate integration rule. Furthermore, similar to the UT, the mean and covariance are computed independently, which implies two different assumptions on the underlying mapping within the same method.

In this paper the transforming function is approximated with a linear combination of Hermite polynomials, for which closed-form expressions for the mean and covariance are well known. The polynomial coefficients are given a hierarchical prior, and the posterior distribution of these coefficients is computed conditioned on the transformed sigma-points. The desired mean and covariance can then be calculated by marginalizing the influence of the coefficients from the analytical expressions. The approximation of the function as a linear combination of Hermite polynomials, with unknown parameters, is the only approximate step in these calculations. Similar approaches have been suggested in [17] and [18], albeit using a non-parametric Gaussian process as a model of the transformation. A Bayesian approach towards learning such a process through evaluations was presented already in [19].

There are several reasons to derive sigma-point algorithms using Bayesian techniques. First, the mean and the covariance matrix estimates are calculated jointly, based on analytical expressions rather than a numerical approach. Hence, it is possible to guarantee a positive-semidefinite covariance matrix. Second, the model assumptions become clearly visible through the prior distribution, making it easier to understand the algorithm. Third, Bayesian methods are generally well performing in the sense that they are admissible under relatively loose assumptions [20] and that they are optimal when the performance is averaged over the prior. Finally, we know that the key to improve performance is the choice of the prior. Although the design of a prior can be difficult, we believe the choice is better made explicitly than implicitly. To illustrate this, we present a family of priors that result in the cubature, UT, and DD2 estimators, for certain choices of the prior. It is shown that the presented algorithm can provide very good estimates of the

mean and covariance, and that the estimation error of the recursive filter is more accurately described using the proposed method. More specifically, we appear to provide more robust covariance estimates, when the underlying polynomials are not completely linear.

The paper is organized as follows. Section 2 describes the estimation task at hand and a summary of the sigma-point approach. The proposed marginalization technique is introduced in Section 3, and is applied to Hermite polynomials in Section 4. Closed-form expressions for mean and covariance are derived in Section 5 together with a summary of the algorithm. Analytical results and a clarification of the relationship to other sigma-point methods are discussed in Section 6. Usage of the technique in a Kalman filter framework is demonstrated in Section 7, and estimation and tracking performance is evaluated in Section 8. Our conclusions are listed in Section 9. Finally, Appendices A–C provide results regarding the positive-definiteness of the UT covariance matrix, properties of Hermite polynomials, and an interpretation of the sigma-point selection scheme.

2 Problem formulation

Consider a transformation $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$ and a stochastic variable $\mathbf{x} \in \mathbb{R}^n$ with probability density function

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}),$$

where g, $\boldsymbol{\mu}_\mathbf{x}$ and $\boldsymbol{P}_\mathbf{x}$ are all known. We wish to calculate the mean and covariance of the transformed variable $\mathbf{y} \in \mathbb{R}^m$:

$$\mathbf{y} = g(\mathbf{x}).$$

These moments are given by the integral expressions

$$\mathbb{E}[\mathbf{y}] = \int_{\mathbb{R}^n} \mathcal{N}(\mathbf{x}\,; \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) g(\mathbf{x}) d\mathbf{x} \tag{1}$$

$$\begin{aligned}\mathrm{Cov}(\mathbf{y}) &= \int_{\mathbb{R}^n} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) \big[g(\mathbf{x}) - \mathbb{E}[\mathbf{y}]\big] \big[g(\mathbf{x}) - \mathbb{E}[\mathbf{y}]\big]^T d\mathbf{x} \\ &= \int_{\mathbb{R}^n} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) g(\mathbf{x}) g(\mathbf{x})^T d\mathbf{x} - \mathbb{E}[\mathbf{y}]\mathbb{E}[\mathbf{y}]^T. \end{aligned} \tag{2}$$

Expressing the solutions to these integrals on a closed form is often impossible for transformations encountered in practice. Sigma-point methods provide approximate solutions to these integrals, and have demonstrated nice properties with respect to performance and simplicity. The question at hand is therefore how to use the sigma-points as efficiently as possible.

2.1 Summary of the sigma-point approach to statistical moment calculations

The family of sigma-point filters approximate integrals (1)–(2) using a weighted sum:

$$\int_{\mathbb{R}^n} \mathcal{N}(\mathbf{x}\,;\boldsymbol{\mu}_{\mathbf{x}}, \boldsymbol{P}_{\mathbf{x}}) g(\mathbf{x}) d\mathbf{x} \approx \sum_{i=0}^{2n} w_i g(\mathbf{x}^i). \tag{3}$$

The so-called sigma-points, $\{\mathbf{x}^0, \ldots, \mathbf{x}^{2n}\}$, and the associated weights, w_i, are chosen according to a deterministic scheme. For the unscented transform, they are:

$$\mathbf{x}^0 = \quad \mathbb{E}[\mathbf{x}] \tag{4}$$

$$\mathbf{x}^i = \begin{cases} \mathbb{E}[\mathbf{x}] + \left(\sqrt{\frac{n}{(1-w_0)}\boldsymbol{P}_{\mathbf{x}}}\right)_i, & 1 \leq i \leq n \\ \mathbb{E}[\mathbf{x}] - \left(\sqrt{\frac{n}{(1-w_0)}\boldsymbol{P}_{\mathbf{x}}}\right)_{i-2n/2}, & n < i \leq 2n \end{cases} \tag{5}$$

$$w_i = \frac{1-w_0}{2n}, \tag{6}$$

where $i = 1, \ldots, 2n$ and $\left(\sqrt{\boldsymbol{P}_{\mathbf{x}}}\right)_i$ is the i^{th} column of a matrix square root such that $\sqrt{\boldsymbol{P}_{\mathbf{x}}}\sqrt{\boldsymbol{P}_{\mathbf{x}}}^T = \boldsymbol{P}_{\mathbf{x}}$. When $\mathbf{x}$ is Gaussian, the suggested setting for the UT [7] is to use $w_0 = 1 - n/3$, whereas the cubature rule is obtained by setting $w_0 = 0$, effectively removing $\mathbf{x}^0$ from the set of sigma-points. This integral approximation strategy, applied to equation (1), yields the estimator

$$\mathbb{E}[g(\mathbf{x})] \approx \sum_{i=0}^{2n} w_i g(\mathbf{x}^i) \triangleq \bar{\mathbf{y}}. \tag{7}$$

The covariance matrix estimate, $\hat{\boldsymbol{P}}_{\mathbf{y}}$, is usually expressed in terms of the weighted sum of squares, but we prefer to view it on the form (2) to make the dual use of the integral approximation clear:

$$\begin{aligned} \text{Cov}(\mathbf{y}) &\approx \sum_{i=0}^{2n} w_i [g(\mathbf{x}^i) - \bar{\mathbf{y}}][g(\mathbf{x}^i) - \bar{\mathbf{y}}]^T \\ &= \sum_{i=0}^{2n} w_i g(\mathbf{x}^i) g(\mathbf{x}^i)^T - \sum_{i=0}^{2n} w_i g(\mathbf{x}^i) \bar{\mathbf{y}}^T - \bar{\mathbf{y}} \sum_{i=0}^{2n} w_i g(\mathbf{x}^i)^T + \sum_{i=0}^{2n} w_i \bar{\mathbf{y}} \bar{\mathbf{y}}^T \\ &= \sum_{i=0}^{2n} w_i g(\mathbf{x}^i) g(\mathbf{x}^i)^T - \bar{\mathbf{y}} \bar{\mathbf{y}}^T \triangleq \hat{\boldsymbol{P}}_{\mathbf{y}}. \end{aligned} \tag{8}$$

If the mean (7) is correctly calculated using a *minimum* number of points, the covariance matrix estimate (8) will in general not be exact. In fact, with negative weights it may not even be positive-semidefinite; see proof in Appendix A.

The DD2 estimator uses the same sigma-point selection scheme (5), but is parameterized using a scalar $h^2 = \frac{n}{1-w_0}$. The sigma points and the weights used to calculate the mean are identical to those of the UT. The covariance matrix approximation, however, employs a different set of weights which are positive regardless of the dimensionality.

3 Proposed idea

Even though the transforming function g is known, we model it as a stochastic process with a prior distribution $\pi(g)$. Apart from the prior, the only available information is the evaluated points, $\chi = \left[\mathbf{x}^0, \ldots, \mathbf{x}^{2n}\right]$, and the function values at these points, $\mathbf{z} = \left[g(\mathbf{x}^0), \ldots, g(\mathbf{x}^{2n})\right]$. Using estimation terminology: χ and $\mathbf{z}$ are our measurements, the function g is a nuisance parameter with posterior distribution $p(g|\mathbf{z}, \chi)$ and our objective is to estimate the mean, $\bar{\mathbf{y}}_\pi$, and covariance, $\boldsymbol{P}_{\mathbf{y},\pi}$, of $\mathbf{y}$.

The mean, expressed as a function of the transformation, g, is denoted by

$$\bar{\mathbf{y}}(g) \triangleq \int \mathcal{N}(\mathbf{x}\,; \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) g(\mathbf{x}) d\mathbf{x}, \tag{9}$$

and the corresponding covariance matrix by

$$P_\mathbf{y}(g) \triangleq \int \mathcal{N}(\mathbf{x}\,; \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x}) [g(\mathbf{x}) - \bar{\mathbf{y}}(g)][g(\mathbf{x}) - \bar{\mathbf{y}}(g)]^T d\mathbf{x}. \tag{10}$$

The expressions for the desired mean and covariance of $\mathbf{y}$, given $\mathbf{z}$ and χ, are given by marginalization over g:

$$\bar{\mathbf{y}}_\pi = \mathbb{E}[\mathbf{y}|\mathbf{z}, \chi] = \int \bar{\mathbf{y}}(g) p(g|\mathbf{z}, \chi) dg \tag{11}$$

$$\boldsymbol{P}_{\mathbf{y},\pi} = \mathbb{E}\left[\boldsymbol{P}_\mathbf{y}(g) \,|\mathbf{z}, \chi\right] = \int \boldsymbol{P}_\mathbf{y}(g) p(g|\mathbf{z}, \chi) dg. \tag{12}$$

The idea is to use a prior $\pi(g)$ for which the integrals in (11) and (12) have closed-form solutions. Although it is possible to find solutions for infinite-dimensional integrals, it is more practical to consider a finite parameterization of g. In this paper we focus on one such prior, presented in Section 4, where g is assumed to belong to the family of Hermite polynomials. An interpretation of using this prior is that the mean (11) and covariance (12) are averaged over polynomials that pass through the points $(\mathbf{x}^i, g(\mathbf{x}^i))$, for all integers $i \in [0, 2n]$, as illustrated in Fig. 1.

4 Using a Hermite polynomial to model the transforming function

In order to motivate the use of Hermite polynomials, and to illustrate some fundamental properties, we study a scalar transformation. Any function g, for which

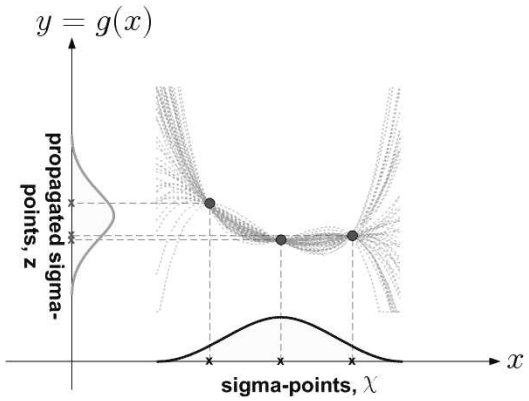


Figure 1: An example of polynomial functions that may have performed the transformation of the sigma-points. The mean and covariance of $y = g(x)$ is calculated by a weighted average of all such functions.

$\mathbb{E}[g(x)^2] < \infty$, can be expressed in terms of a series of weighted Hermite polynomials [10]:

$$g(x) = \sum_{k=0}^{\infty} \frac{1}{k!} \mathbb{E}[g(x) H_k(x)] H_k(x), \tag{13}$$

for $x \sim \mathcal{N}(0, 1)$. To have a tractable solution, we assume that the transforming function can be approximated using a finite series, fully described by a weight vector $\boldsymbol{\theta} = [\theta_0, \ldots, \theta_p]^T$:

$$g(x) \approx \sum_{k=0}^{p} \theta_k H_k(x). \tag{14}$$

The k^{th} order hermite polynomial, H_k, is given by (85) in Appendix B, which contains a summary of useful properties of Hermite polynomials. For instance, the usage of Hermite polynomials leads to very simple expressions for the mean, $\bar{y}(g)$, and covariance, $P_y(g)$, or as they now can be expressed, $\bar{y}(\boldsymbol{\theta})$ and $\sigma_y^2(\boldsymbol{\theta})$:

$$\begin{aligned} \bar{y}(\boldsymbol{\theta}) &= \theta_0 \\ \sigma_y^2(\boldsymbol{\theta}) &= \sum_{k=1}^{p} \theta_k^2 k! \,. \end{aligned}$$

For example, if $y = x + x^2$ and $x \sim \mathcal{N}(0, 1)$, then $y = H_0 + H_1 + H_2$, (i.e., $\boldsymbol{\theta} = [1\ 1\ 1]^T$). Consequently, the expected value is $\theta_0 = 1$ and the variance is $\theta_1^2 + \theta_2^2 2! = 3$.

Additionally, using Hermite polynomials facilitates comparisons with other sigma-point methods, which typically calculate the integral (7) exactly for certain polynomials.

4.1 Multidimensional transformation

A transformation $g : \mathbb{R}^n \to \mathbb{R}^m$, performed by a linear combination of base functions can be written as

$$g(\mathbf{x};\boldsymbol{\theta}) = \boldsymbol{\theta}^T \mathbf{h}(\mathbf{x}), \tag{15}$$

where the base functions enter the equation through

$$\mathbf{h}(\mathbf{x}) = [H_0,\ H_1(x_1), \ldots, H_p(x_1),\ H_1(x_2), \ldots, H_p(x_2), \ldots,\ H_1(x_n), \ldots, H_p(x_n)]^T. \tag{16}$$

In the following sections we assume $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{n\times n})$, a simplification justified in Section 4.4. We construct the weight matrix from the p-dimensional vectors $\boldsymbol{\theta}^{i,j}$, each describing the transformation from x_i to y_j, and the scalars θ_0^j, for $i = 1 \ldots n$ and $j = 1 \ldots m$:

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0^1 & \ldots & \theta_0^j & \ldots & \theta_0^m \\ \boldsymbol{\theta}^{1,1} & \ldots & \boldsymbol{\theta}^{1,j} & \ldots & \boldsymbol{\theta}^{1,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \boldsymbol{\theta}^{n,1} & \ldots & \boldsymbol{\theta}^{n,j} & \ldots & \boldsymbol{\theta}^{n,m} \end{bmatrix}. \tag{17}$$

Consequently, $\boldsymbol{\theta}^j$, the j^{th} column of $\boldsymbol{\theta}$, defines the mapping from $\mathbf{x} \in \mathbb{R}^n$ to y_j over the base functions in $\mathbf{h}(\mathbf{x})$:

$$y_j = (\boldsymbol{\theta}^j)^T \mathbf{h}(\mathbf{x}). \tag{18}$$

The function g is completely described by $\boldsymbol{\theta}$ through equation (15), and we turn our attention to the expressions for $\bar{\mathbf{y}}(\boldsymbol{\theta})$ and $\boldsymbol{P}_{\mathbf{y}}(\boldsymbol{\theta})$. For a given polynomial, i.e., one realization of $\boldsymbol{\theta}$, $\mathbf{y}$ has the mean

$$\begin{aligned} \bar{\mathbf{y}}(\boldsymbol{\theta}) &= \mathbb{E}[\boldsymbol{\theta}^T \mathbf{h}(\mathbf{x}) | \boldsymbol{\theta}] \\ &= \left[\theta_0^1, \ldots, \theta_0^m\right]^T, \end{aligned} \tag{19}$$

where $\mathbb{E}[\mathbf{h}(\mathbf{x})]$ is given by equation (83). To simplify notation, we introduce the vector

$$\mathbf{w} \triangleq \mathbb{E}[\mathbf{h}(\mathbf{x})] = [1,\ 0, \ldots, 0]^T, \tag{20}$$

and write the covariance matrix for $\mathbf{y}$:

$$\begin{aligned} \boldsymbol{P}_{\mathbf{y}}(\boldsymbol{\theta}) &= \mathbb{E}\left[[\boldsymbol{\theta}^T \mathbf{h}(\mathbf{x}) - \boldsymbol{\theta}^T \mathbf{w}][\boldsymbol{\theta}^T \mathbf{h}(\mathbf{x}) - \boldsymbol{\theta}^T \mathbf{w}]^T | \boldsymbol{\theta}\right] \\ &= \boldsymbol{\theta}^T \mathbb{E}\left[[\mathbf{h}(\mathbf{x}) - \mathbf{w}][\mathbf{h}(\mathbf{x}) - \mathbf{w}]^T\right] \boldsymbol{\theta} \\ &= \boldsymbol{\theta}^T \mathbf{C} \boldsymbol{\theta}. \end{aligned} \tag{21}$$

All off-diagonal elements of $\mathbf{C} \triangleq \mathbb{E}[[\mathbf{h}(\mathbf{x}) - \mathbf{w}][\mathbf{h}(\mathbf{x}) - \mathbf{w}]^T]$ are zero, and the $pn + 1$ diagonal elements are:

$$\text{diag}(\mathbf{C}) = [0,\ 1!,\ 2!, \ldots,\ p!, \ldots,\ 1!,\ 2!, \ldots,\ p!]^T, \tag{22}$$

see equations (83) and (84) in Appendix B. The relation between the mean (19) and covariance (21) of $\mathbf{y}$, and the parameter vector $\boldsymbol{\theta}$, is now clear. Before we attempt to marginalize $\boldsymbol{\theta}$ from these expressions, we attend to the prior.

4.2 Designing the prior distribution

Using Hermitian polynomials, designing the prior $\pi(g)$ is now equivalent to designing $\pi(\boldsymbol{\theta})$, and there is an intuitive interpretation: the number of elements in $\boldsymbol{\theta}$ determines the maximum order of the transforming polynomial. Similarly, the variance determines which coefficients are updated with the information provided in the propagated sigma points.

The proposed prior assumes the vectors $\boldsymbol{\theta}^{i,j}$ to be independently generated from a hierarchical model:

$$\boldsymbol{\theta}^{i,j} \sim \mathcal{N}(0, \alpha_j \boldsymbol{P}_\theta^{i,j}). \tag{23}$$

It is shown in Section 6.2 that the sigma-points can be selected such that the prior on θ_0 does not affect the posterior distribution, $p(\boldsymbol{\theta}|\mathbf{z}, \chi)$, but for completeness let it be assumed that all scalars θ_0^j are independently drawn from $\mathcal{N}(0, \sigma_{\theta_0}^2)$. The covariance matrix $\text{Cov}(\boldsymbol{\theta}^j) = \alpha_j \boldsymbol{P}_\theta^j$ is therefore block-diagonal, with:

$$\boldsymbol{P}_\theta^j = \begin{bmatrix} \sigma_{\theta_0}^2/\alpha_j & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{P}_\theta^{1,j} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \ldots & \mathbf{0} & \boldsymbol{P}_\theta^{n,j} \end{bmatrix}. \tag{24}$$

The hyperparameter, α_j, will be discussed in Section 5.2.

4.3 Estimates of mean and covariance

Expressions (19) and (21) are derived for a given weight matrix, $\boldsymbol{\theta}$. However, since $\boldsymbol{\theta}$ is modeled as a stochastic variable, the marginalization in (11) and (12) gives the final estimators:

$$\begin{aligned} \bar{\mathbf{y}}_\pi &= \mathbb{E}[\boldsymbol{\theta}^T|\mathbf{z}, \chi]\mathbf{w} \\ &= \left\{ \boldsymbol{\mu}_{\theta|\mathbf{z}}^T \triangleq \mathbb{E}[\boldsymbol{\theta}^T|\mathbf{z}, \chi] \right\} = \boldsymbol{\mu}_{\theta|\mathbf{z}}^T \mathbf{w} \\ &= \mathbb{E}\left[[\theta_0^1, \ldots, \theta_0^m]^T|\mathbf{z}, \chi\right] \end{aligned} \tag{25}$$

$$\begin{aligned}\boldsymbol{P}_{\mathbf{y},\pi} &= \mathbb{E}[\boldsymbol{\theta}^T \mathbf{C}\, \boldsymbol{\theta}|\mathbf{z},\chi] \\ &= \boldsymbol{\mu}_{\theta|\mathbf{z}}^T \mathbf{C}\, \boldsymbol{\mu}_{\theta|\mathbf{z}} + \mathbb{E}\left[[\boldsymbol{\theta}-\boldsymbol{\mu}_{\theta|\mathbf{z}}]^T \mathbf{C}[\boldsymbol{\theta}-\boldsymbol{\mu}_{\theta|\mathbf{z}}]|\mathbf{z},\chi\right] \\ &= \boldsymbol{\mu}_{\theta|\mathbf{z}}^T \mathbf{C}\, \boldsymbol{\mu}_{\theta|\mathbf{z}} + \begin{bmatrix} \alpha_1 \mathrm{Tr}\left\{\boldsymbol{P}_{\theta|\mathbf{z}}^1 \mathbf{C}\right\} & & 0 \\ & \ddots & \\ 0 & & \alpha_m \mathrm{Tr}\left\{\boldsymbol{P}_{\theta|\mathbf{z}}^m \mathbf{C}\right\} \end{bmatrix}. \end{aligned} \tag{26}$$

Expressions for the conditional mean, $\boldsymbol{\mu}_{\theta|\mathbf{z}}$, and posterior covariance matrices, $\boldsymbol{P}_{\theta|\mathbf{z}}^j$ $(j = 1 \dots m)$, given observations $\mathbf{z}, \chi$, are derived in Section 5.

4.4 Stochastic decoupling

The simple forms for $\mathbf{w}$ in (20) and $\mathbf{C}$ in (22) are expressed for vector arguments, $\mathbf{x}$, whose elements are uncorrelated with unit variance. Rather than expressing $\mathbf{w}$ and $\mathbf{C}$ for any mean and covariance of $\mathbf{x}$, a stochastic decoupling procedure similar to the approach in [6] is proposed, such that $\mathbf{w}$ and $\mathbf{C}$ are constant. Instead of studying

$$\mathbf{y} = g(\mathbf{x}), \quad \mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}}, \boldsymbol{P}_{\mathbf{x}}), \tag{27}$$

we introduce $\tilde{\mathbf{x}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{n\times n})$, where $\mathbf{I}_{n\times n}$ is the $n \times n$ identity matrix, and set

$$\mathbf{y} = \tilde{g}(\tilde{\mathbf{x}}) \triangleq g(\boldsymbol{\mu}_{\mathbf{x}} + \sqrt{\boldsymbol{P}_{\mathbf{x}}}\, \tilde{\mathbf{x}}), \tag{28}$$

which has the same distribution as the original $\mathbf{y}$ in (27). *Therefore, rather than recalculating* $\mathbf{w}$ *and* $\mathbf{C}$*, we assume the transformation is performed by* $\tilde{g}$ *in* (28)*. This adaptation is built in to the algorithm described in Section 5.3.*

5 Calculating the posterior distribution

Our objective is now to calculate the posterior distribution $p(\boldsymbol{\theta}|\mathbf{z},\chi)$ and its first two moments, which are needed in the expressions for the mean and covariance of $\mathbf{y}$, given by equations (25)–(26). An exact expression of the distribution is obtained by marginalizing the hyperparameter, α, from the hierarchical model:

$$p(\boldsymbol{\theta}^j|\mathbf{z},\chi) = \int p(\boldsymbol{\theta}^j|\alpha_j,\mathbf{z},\chi) p(\alpha_j|\mathbf{z},\chi) d\alpha_j. \tag{29}$$

Finding a closed-form solution to (29) is usually difficult. A simple yet useful substitute is to use a point estimate of α_j. In other words, we set

$$p(\boldsymbol{\theta}^j|\mathbf{z},\chi) \approx p(\boldsymbol{\theta}^j|\hat{\alpha}_j,\mathbf{z},\chi). \tag{30}$$

In the following section, the first two moments of $p(\boldsymbol{\theta}^j|\hat{\alpha}_j,\mathbf{z},\chi)$ are calculated for a given estimate, $\hat{\alpha}_j$, which is then derived in Section 5.2.

5.1 Mean and covariance of $\boldsymbol{\theta}$

The linear relation between observations $\mathbf{z}$ and parameter vector $\boldsymbol{\theta}$ was established in equation (15):

$$\mathbf{z} = \boldsymbol{\theta}^T \mathbf{H}^T(\chi), \tag{31}$$

where the observation matrix is given by:

$$\mathbf{H}(\chi) = \begin{bmatrix} \mathbf{h}^T(\mathbf{x}^0) \\ \vdots \\ \mathbf{h}^T(\mathbf{x}^{2n}) \end{bmatrix}. \tag{32}$$

For notational convenience, we omit the reference to χ from now on. Given a zero-mean Gaussian prior distribution on $\boldsymbol{\theta}^j$, with $\mathrm{Cov}(\boldsymbol{\theta}^j) = \alpha_j \boldsymbol{P}_\theta^j$, the posterior distribution is also Gaussian with mean and covariance [21]:

$$\boldsymbol{\mu}_{\theta|\mathbf{z}}^j = \boldsymbol{P}_\theta^j \mathbf{H}^T \left[\mathbf{H} \boldsymbol{P}_\theta^j \mathbf{H}^T \right]^{-1} \mathbf{z}^j \tag{33}$$

$$\alpha_j \boldsymbol{P}_{\theta|\mathbf{z}}^j = \left(\mathbf{I} - \boldsymbol{P}_\theta^j \mathbf{H}^T \left[\mathbf{H} \boldsymbol{P}_\theta^j \mathbf{H}^T \right]^{-1} \mathbf{H} \right) \alpha_j \boldsymbol{P}_\theta^j, \tag{34}$$

where $\mathbf{z}^j$ is the j^{th} column in $\mathbf{z}^T$. The conditional mean of $\boldsymbol{\theta}$ is $\boldsymbol{\mu}_{\theta|\mathbf{z}} = [\boldsymbol{\mu}_{\theta|\mathbf{z}}^1, \boldsymbol{\mu}_{\theta|\mathbf{z}}^2, \ldots, \boldsymbol{\mu}_{\theta|\mathbf{z}}^m]$. Estimates $\bar{\mathbf{y}}_\pi$ and $\boldsymbol{P}_{\mathbf{y},\pi}$ in (25) and (26) can thus readily be calculated.

If all transformations are treated the same way a priori, i.e., if the covariance matrices $\boldsymbol{P}_\theta^{i,j}$ in (23) do not depend on j, the elements $\mathrm{Tr}\{\boldsymbol{P}_\theta^j \mathbf{C}\}$ are also independent of j. Hence, the superscript j can be dropped and the expression for $\boldsymbol{P}_{\mathbf{y},\pi}$ can be simplified to

$$\boldsymbol{P}_{\mathbf{y},\pi} = \boldsymbol{\mu}_{\theta|\mathbf{z}}^T \mathbf{C} \boldsymbol{\mu}_{\theta|\mathbf{z}} + \begin{bmatrix} \alpha_1 & & 0 \\ & \ddots & \\ 0 & & \alpha_m \end{bmatrix} \mathrm{Tr}\big\{ \boldsymbol{P}_{\theta|\mathbf{z}} \mathbf{C} \big\}. \tag{35}$$

To simplify notation in the remaining part of the paper, it is assumed that $\boldsymbol{P}_\theta$ and $\boldsymbol{P}_\theta^j$ can be used interchangeably. Furthermore, according to equation (34), $\mathrm{Tr}\{\boldsymbol{P}_{\theta|\mathbf{z}} \mathbf{C}\}$ does not depend on $\mathbf{z}$ and can therefore be calculated in advance.

5.2 The hyperparameter α

Estimates of α_j, which were assumed known in the previous section, are preferably derived from the posterior distribution conditioned on the propagated sigma-points $\mathbf{z}$:

$$p(\alpha_j|\mathbf{z}) \propto p(\mathbf{z}|\alpha_j) p(\alpha_j). \tag{36}$$

The posterior, on the other hand, relies on expressions for the likelihood $p(\mathbf{z}\big|\alpha_j)$ and the prior $p(\alpha_j)$.

The likelihood function

In our setting, $\boldsymbol{\theta}^j$ is a zero-mean Gaussian random variable, conditioned on α_j, and so is the linearly dependent observations $\mathbf{z}^j$. However, from the results in Appendix C it follows that the mean is known for the cases we study and, consequently, is independent of the hyperparameter prior. The observation vector of interest, $\tilde{\mathbf{z}}^j$, is therefore the j^{th} column in $\left[g(\mathbf{x}^0) - \boldsymbol{\theta}_0, \ldots, g(\mathbf{x}^{2n}) - \boldsymbol{\theta}_0\right]^T$, and the likelihood function takes the following simple form:

$$p(\tilde{\mathbf{z}}^j|\alpha_j) = \frac{1}{(2\pi)^{\frac{\rho}{2}}(\alpha_j)^{\frac{\rho}{2}}\sqrt{|\mathbf{H}\boldsymbol{P}_\theta^j\mathbf{H}^T|}}\ e^{-\frac{1}{2\alpha_j}\tilde{\mathbf{z}}^{jT}(\mathbf{H}\boldsymbol{P}_\theta^j\mathbf{H}^T)^{-1}\tilde{\mathbf{z}}^j}, \tag{37}$$

in which ρ is the number of observations, in this case $2n+1$.

The prior

In the absence of prior knowledge of α_j, we want the prior to be noninformative to ensure a weak influence on the posterior distribution. It is argued in [22] that

$$p(\alpha_j) \propto 1/\alpha_j, \tag{38}$$

is a sensibly vague prior with respect to the likelihood (37).

The posterior distribution

The expression for the posterior distribution, using the likelihood (37) and prior (38), is:

$$p(\tilde{\mathbf{z}}^j|\alpha_j)\, p(\alpha_j) \propto \frac{1}{{\alpha_j}^{\frac{\rho}{2}+1}}\ e^{-\frac{1}{2\alpha_j}d^2}, \tag{39}$$

where $d^2 = \tilde{\mathbf{z}}^{j\,T}(\mathbf{H}\boldsymbol{P}_\theta^j\mathbf{H}^T)^{-1}\tilde{\mathbf{z}}^j$. The above expression is proportional to the scaled inverse chi-square distribution, so

$$\alpha_{j|\mathbf{z}} \sim \text{inv-}\chi^2(\nu, s^2), \tag{40}$$

with parameters $\nu = \rho$ and $s^2 = d^2/\rho$. The mean and mode of the scaled inverse chi-square distribution are:

$$\mathbb{E}(\alpha_j) = \frac{\nu}{\nu-2}s^2, \quad \text{mode}(\alpha_j) = \frac{\nu}{\nu+2}s^2, \tag{41}$$

and can be used as point estimates of α_j in the posterior covariance matrix expression (35). Note that the conditional mean (33) is unaffected by the hyperparameter. The algorithm presented in Section 5.3 employs the mode (41) as a point estimate of α_j.

5.3 The marginalized transform (MT) estimator

We have now reached the point where the MT estimation algorithm can be summarized, and somewhat simplified, in a few easy steps. There are two design decisions that can be made independently: the order of the transforming polynomial, p, and the sigma-point selection scheme. Using $2 \leq p \leq 3$ for the cubature points and $2 \leq p \leq 5$ for the UT points assures a fully known mean (further explained in Section 6.2).

For $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{y} = g(\mathbf{x}) \in \mathbb{R}^m$, $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_\mathbf{x}, \boldsymbol{P}_\mathbf{x})$, do:

1. Select a prior covariance matrix $\boldsymbol{\Sigma}$, a diagonal $p \times p$ matrix, $p \leq 5$, with at least two nonzero elements. See, e.g., the priors used in Section 8.

2. Generate sigma-points using $w_0 = 1 - \frac{n}{3}$:

$$\begin{aligned}
\mathbf{x}^0 &= \quad \mathbf{0}_{n\times 1} \\
\mathbf{x}^k &= \begin{cases} +\left(\sqrt{\frac{n}{(1-w_0)}\mathbf{I}_{n\times n}}\right)_k & ,\ 1 < k \leq n \\ -\left(\sqrt{\frac{n}{(1-w_0)}\mathbf{I}_{n\times n}}\right)_{k-n} & ,\ n < k \leq 2n \end{cases} \\
\chi &= \left[\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^{2n}\right].
\end{aligned}$$

   (although $w_0 = 0$ *can* be used if $p \leq 3$, and for $p = 2$, *any* $2n+1$ points can be used).

3. Set $\boldsymbol{P}_\theta^{i,j} = \boldsymbol{\Sigma}$ in equation (24) to form $\boldsymbol{P}_\theta$. The value for $\sigma_{\theta_0}^2/\alpha_j$ will not matter. Calculate $\mathbf{w}, \mathbf{C}, \mathbf{H}(\chi)$ and $\boldsymbol{P}_{\theta|\mathbf{z}}$ using equations (20), (22), (32) and (34) respectively.

4. Propagate the sigma-points:

$$\mathbf{z} = \left[g(\boldsymbol{\mu}_\mathbf{x} + \sqrt{\boldsymbol{P}_\mathbf{x}}\mathbf{x}^0), \ldots, g(\boldsymbol{\mu}_\mathbf{x} + \sqrt{\boldsymbol{P}_\mathbf{x}}\mathbf{x}^{2n})\right].$$

5. Compute the mean, $\bar{\mathbf{y}}_\pi$, using equation (25) and (33):

$$\begin{aligned}
\boldsymbol{\mu}_{\theta|\mathbf{z}} &= \boldsymbol{P}_\theta \mathbf{H}^T \left[\mathbf{H}\boldsymbol{P}_\theta\mathbf{H}^T\right]^{-1} \mathbf{z}^T \\
\bar{\mathbf{y}}_\pi &= \boldsymbol{\mu}_{\theta|\mathbf{z}}\mathbf{w}
\end{aligned}$$

6. Estimate the modes of the hyperparameters:

$$\hat{\alpha}_j = \frac{1}{(2n+1)+2}\tilde{\mathbf{z}}^{j\,T}\left[\mathbf{H}\boldsymbol{P}_\theta\mathbf{H}^T\right]^{-1}\tilde{\mathbf{z}}^{j\,T},$$

where $\tilde{\mathbf{z}}^{j\,T}$ is the j^{th} row in the observation matrix with subtracted mean, $\big[g(\mathbf{x}^0) - \bar{\mathbf{y}}_\pi, \ldots, g(\mathbf{x}^{2n}) - \bar{\mathbf{y}}_\pi\big]$.

7. Calculate the covariance matrix, $\boldsymbol{P}_{\mathbf{y},\pi}$, using equation (35).

Steps $1-3$ can be done in advance, as well as computing $\boldsymbol{P}_\theta \mathbf{H}^T \big[\mathbf{H}\boldsymbol{P}_\theta \mathbf{H}^T\big]^{-1}$, $\big[\mathbf{H}\boldsymbol{P}_\theta \mathbf{H}^T\big]^{-1}$ and $\mathrm{Tr}\big\{\boldsymbol{P}_{\theta|\mathbf{z}}\mathbf{C}\big\}$, in that way simplifying the algorithm significantly. For example, the calculation of the mean can be identical to the UT, cubature rule or to the DD2, for which also the covariance matrix estimator can be the same — all depending on the design of the prior, see the discussion in Section 6.

5.4 Calculating the posterior cross-covariance matrix

It is sometimes required to know the cross-covariance between the state, $\mathbf{x}$, and the transformed state, $\mathbf{y} = g(\mathbf{x})$. In the filtering algorithm that will be presented in Section 7.3, it is a necessity, and is in fact already known from estimating $\boldsymbol{\mu}_{\theta|\mathbf{z}}$. The cross-covariance matrix is:

$$\begin{aligned}
\boldsymbol{P}_{\mathbf{xy}}(\boldsymbol{\theta}) &= \int_{R^n} \mathcal{N}(\mathbf{x}\,; \mathbf{0}, \mathbf{I}_n)\big[\mathbf{x} - \mathbb{E}[\mathbf{x}]\big]\big[g(\mathbf{x};\boldsymbol{\theta}) - \bar{\mathbf{y}}(\boldsymbol{\theta})\big]^T d\mathbf{x} \\
&= \mathbb{E}[\mathbf{x}[\boldsymbol{\theta}^T\mathbf{h}(\tilde{\mathbf{x}}) - \boldsymbol{\theta}^T\mathbf{w}]^T] \\
&= \mathbb{E}[\mathbf{x}[\mathbf{h}(\mathbf{x}) - \mathbf{w}]^T]\boldsymbol{\theta} \\
&= \mathbf{D}\boldsymbol{\theta}. \qquad (42)
\end{aligned}$$

The sparse matrix $\mathbf{D} \triangleq \mathbb{E}[\tilde{\mathbf{x}}[\mathbf{h}(\tilde{\mathbf{x}}) - \mathbf{w}]^T]$ is constant and can be written:

$$\mathbf{D} = \begin{bmatrix} 0 & [1,0,\ldots,0] & \mathbf{0}^T & \cdots \\ 0 & \mathbf{0}^T & \ddots & \ddots \\ \vdots & \vdots & \ddots & [1,0,\ldots,0] \end{bmatrix}, \qquad (43)$$

which follows from the orthogonality property (82) of Hermite polynomials described in Appendix B (recall that $x = H_1(x)$). In other words, $\boldsymbol{P}_{\mathbf{xy}}(\boldsymbol{\theta})$ is the $n \times m$ matrix of all first order weights:

$$\boldsymbol{P}_{\mathbf{xy}}(\boldsymbol{\theta}) = \begin{bmatrix} \theta^{1,1}(1) & \ldots & \theta^{1,m}(1) \\ \theta^{2,1}(1) & \ldots & \theta^{2,m}(1) \\ \vdots & & \vdots \\ \theta^{n,1}(1) & \ldots & \theta^{n,m}(1) \end{bmatrix}. \qquad (44)$$

The above cross-covariance matrix describes the relation to $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{n\times n})$, whereas the relation to a correlated state is established by multiplication with

$\sqrt{\boldsymbol{P}_{\mathbf{x}}}$. Including the square-root matrix and carrying out the marginalization of $\boldsymbol{\theta}$ in (42) yields

$$\begin{aligned}\boldsymbol{P}_{\mathbf{xy},\pi} =& \sqrt{\boldsymbol{P}_{\mathbf{x}}}\ \mathbf{D}\mathbb{E}\left[\boldsymbol{\theta}\middle|\mathbf{z},\chi\right] \\ =& \sqrt{\boldsymbol{P}_{\mathbf{x}}}\ \mathbf{D}\boldsymbol{\mu}_{\theta|\mathbf{z}},\end{aligned} \tag{45}$$

which is the estimate of the cross covariance matrix.

6 Analysis and comparison

In this section, we further explain the behavior of the proposed estimator, and clarify the relationship with other sigma-point estimators.

6.1 Posterior uncertainties in mean and covariance

First, we analyze our estimates in terms of their distributions. Conditioned on α, the mean, $\bar{\mathbf{y}}(\boldsymbol{\theta})$, is a Gaussian random variable with covariance

$$\mathbb{E}\left[[\bar{\mathbf{y}}(\boldsymbol{\theta}) - \bar{\mathbf{y}}_\pi][\bar{\mathbf{y}}(\boldsymbol{\theta}) - \bar{\mathbf{y}}_\pi]^T|\alpha\right] = \mathbf{I}_{m\times m}\mathbf{w}\boldsymbol{P}_{\theta|\mathbf{z}}\mathbf{w}^T. \tag{46}$$

The distribution of the elements in the covariance matrix, $\boldsymbol{P}_{\mathbf{y}}(\boldsymbol{\theta})$, is less trivial; diagonal elements are weighted sums of chi-square distributed variables, whereas the off-diagonal elements are created from products between independent Gaussian random variables. This could be looked upon as a weighted sum of Wishart distributed matrices created from the rows, $\boldsymbol{\theta}_i$, of $\boldsymbol{\theta}$:

$$\boldsymbol{P}_{\mathbf{y}}(\boldsymbol{\theta}) = \sum_{k=0}^{pn} \boldsymbol{\theta}_k^T\boldsymbol{\theta}_k c_{k+1}, \tag{47}$$

where c_k is the k^{th} diagonal element in $\mathbf{C}$, defined in equation (22).

Equation (46) illustrates how uncertainties in $\boldsymbol{\theta}$ affect $\bar{\mathbf{y}}(\boldsymbol{\theta})$, and it is desirable to design an estimator such that this variance equals zero. Inserting the expression for $\boldsymbol{P}_{\theta|\mathbf{z}}$, from equation (34), into (46), we see that the covariance of $\bar{\mathbf{y}}(\boldsymbol{\theta})$ is

$$\mathrm{Cov}(\bar{\mathbf{y}}(\boldsymbol{\theta})) = \mathbf{w}\left(\mathbf{I} - \boldsymbol{P}_\theta\mathbf{H}^T\left[\mathbf{H}\boldsymbol{P}_\theta\mathbf{H}^T\right]^{-1}\mathbf{H}\right)\boldsymbol{P}_\theta\mathbf{w}^T. \tag{48}$$

One of the arguments for sigma-point approaches has been that it is easier to approximate the probability distribution than the transforming function [7], [23]. However, it is not required for $\boldsymbol{\theta}$ to be fully known ($\boldsymbol{P}_{\theta|\mathbf{z}} = 0$) in order for the estimate to be exact; we see from equation (48) that it is enough to project the uncertainties in $\boldsymbol{\theta}$ onto the plane orthogonal to the vector $\mathbf{w}$. In Appendix C it is shown that the selection scheme (4)–(5) attains this projection, which means that $\bar{\mathbf{y}}_\pi = \bar{\mathbf{y}}(\boldsymbol{\theta})$ with probability one. In other words, $\bar{\mathbf{y}}(\boldsymbol{\theta})$ is identical for all polynomials passing through the sigma-points.

The result follows from using an integration rule, well-known from the literature, [12], [14], which integrates these functions correctly. However, the new derivation provided here is conceptually different and may be more intuitive to some readers. Furthermore, the type of uncertainty analysis performed in this paper can provide an important tool for designing new sigma-point selection schemes in the future.

6.2 Comparison with the UT and the cubature rule

Contrary to the UT and the cubature rule, the presented method suggests to calculate the covariance matrix using a model of the transformation, and the estimates are therefore conceptually different. The estimates of the mean, however, are easier to compare; the UT and the cubature rule employ known integration rules, and the proposed method can yield these rules under certain conditions. To show the similarities, we write the MT estimator of the mean (25) on the same form as the UT estimator (7):

$$\bar{\mathbf{y}}_\pi = \mathbf{z}\left[\boldsymbol{P}_\theta \mathbf{H}^T \left[\mathbf{H}\boldsymbol{P}_\theta \mathbf{H}^T\right]^{-1}\right]^T \mathbf{w}. \tag{49}$$

This is clearly a weighted sum, $\bar{\mathbf{y}}_\pi = \mathbf{z}\boldsymbol{\lambda}$, of the evaluated sigma-points, with a column weight vector

$$\boldsymbol{\lambda} = \left[\mathbf{H}\boldsymbol{P}_\theta \mathbf{H}^{\mathbf{T}}\right]^{-1} \mathbf{H}\boldsymbol{P}_\theta \mathbf{w}. \tag{50}$$

The MT and UT estimators are the same when the elements of $\boldsymbol{\lambda}$ are identical to the UT weights.

The definition of the precision of an integration rule is [14]:
'A rule is said to have precision p if it integrates monomials up to degree p exactly, that is, monomials $\Pi_{i=1}^d x_i^{k_i}$ *with* $k_i \geq 0$ *and* $\sum_{i=1}^d k_i \leq p$*, but not exactly for some monomials of degree* $\sum_{i=1}^d k_i = p+1$*'.*

For the presented method, this definition is equivalent to having no uncertainties in $\bar{\mathbf{y}}(\boldsymbol{\theta})$, when the prior includes all monomials up to degree p. It is shown in Appendix C that the sigma-point selection scheme (4) – (5) satisfies exactly this — the MT and UT estimators for the mean are then identical. The explicit model assumptions in the proposed method coincide with the implicit assumptions in the sigma-point filter, and the actual values in the prior covariance matrix, $\boldsymbol{P}_\theta$, no longer affect the result.

The integration rule used by the UT and the cubature rule have precision 3, which can be quite limiting. A simple example serves as illustration:

$$y = x_1 x_2, \quad x \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{2\times 2}). \tag{51}$$

The variance of y is $\mathbb{E}[x_1^2 x_2^2] = 1$, but the sigma-point methods discussed in this paper all fail to calculate the variance correctly. However, the prior used in the presented method explicitly excludes cross-terms in the model, so the result should

come as no surprise. Moreover, the solution is straightforward: modify the model to include also cross-terms and add sigma-points to observe them. It should be mentioned here that the MT and the UT, with $w_0 = 1 - n/3$, would have precision 5 if it weren't for these cross-terms, i.e., single-element monomials, x_i^p, are correctly integrated up to $p = 5$.

Contrary to the UT, the MT can be tuned without moving the sigma-points. The cubature rule, on the other hand, cannot be tuned at all, and the position of the sigma-points varies in a predetermined manner with the dimensionality, n. For instance, in a tracking system where targets are tracked using a joint state vector, the performance of the cubature estimator depends on the number of targets, even if the targets are well separated with independent measurements (with respect to other targets).

6.3 Comparison with the divided difference filter

The DD2 is based on a second-order polynomial approximation of the transforming function, with cross-terms excluded. The MT assumes that the underlying distribution is Gaussian, which corresponds to setting the DD2 design parameter $h = \sqrt{3}$. It is possible to design an MT-prior to correspond to this estimator. More specifically, assuming a second order polynomial and using the UT sigma-points yields equally many unknowns as observations. The second order polynomial is therefore fully known, i.e. there are no posterior uncertainties in the parameter vector $\boldsymbol{\theta}$, and the estimators are, for this particular prior, identical.

6.4 Selecting a set of sigma-points

The effects of employing a particular set of sigma points with the MT can be evaluated in terms of the posterior uncertainties of the estimates. However, our focus here is to evaluate the MT performance when using the $2n + 1$ UT points, and the $2n$ cubature points, where the main difference is that the cubature rule does not employ a weight in the distribution mean.

It is foreseeable that there will be functions for which the integral of a polynomial passing through the evaluated sigma-points, may constitute a worse approximation of the actual integral, than the integral over a lower order polynomial passing through fewer points. For instance, in [9], it was shown that the cubature rule performed better than the DD2 in estimating the mean of the function

$$g(\mathbf{x}) = \frac{1}{(\sqrt{1 + \mathbf{x}^T\mathbf{x}})^q}, \tag{52}$$

when the integer q and the dimensionality of $\mathbf{x}$ was increased. Under these circumstances, the function (52) does not resemble a polynomial, and including a sigma-point in the mean $\mathbb{E}[\mathbf{x}]$ degrades performance. It cannot, however, be argued that it is generally sound to exclude that particular sigma-point — it has to be judged depending on the function. Including the point provides information

of the function, which obviously sometimes is helpful, especially when calculating the covariance matrix. For example, the covariance matrix for functions symmetric over the covariance contour will be zero when calculated using the cubature rule, e.g.:

$$y = x^2, \quad x \sim \mathcal{N}(0, 1). \tag{53}$$

If all propagated points have the same value this will also be the estimate of the mean, i.e., $g(\mathbf{x}^i) = \bar{\mathbf{y}}$ for all sigma-points. The variance estimate is then:

$$\sum_{i=0}^{2n} w_i [g(\mathbf{x}^i) - \bar{\mathbf{y}}][g(\mathbf{x}^i) - \bar{\mathbf{y}}]^T = \mathbf{0}.$$

This would be the case also for (52), if $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. In real situations this is rarely the case, but nevertheless illustrates an undesired behavior.

7 Application example: Recursive filtering

Robust recursive filters, e.g., for tracking a continuous process measured at discrete time instances, are arguably very valuable. A famous solution is the Kalman filter (KF) [4], although the KF is applicable only when models are linear. Several filters intended for usage with non-linear models share a similar structure, differing only in how they estimate moments, e.g., the UKF, CKF, and EKF. By applying the marginalization technique presented in this paper in a similar fashion, the marginalized Kalman filter is created — the MKF.

7.1 System model

A discrete-time non-linear system, described by the state vector, $\mathbf{x}_k$, is assumed to evolve according to the model:

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{w}_{k-1}). \tag{54}$$

Observations, $\mathbf{y}_k$, are provided at discrete time instances:

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{v}_k). \tag{55}$$

The noise terms $\mathbf{w}_k, \mathbf{v}_k$ are modeled as zero mean independent white Gaussian noise. The goal is to calculate the posterior distribution $p(\mathbf{x}_k|\mathbf{Y}_k)$, where $\mathbf{Y}_k$ is the collection of all available measurements, $[\mathbf{y}_1, \ldots, \mathbf{y}_k]$. Estimates of the state vector are often denoted $\hat{\mathbf{x}}_{k|k}$, where the first subscript refers to the time index of the state and the latter to the time index of the last measurement used to update the state.

7.2 The one-step linear estimation algorithm

An accustomed approach for calculating the posterior distribution, used for example by the EKF, UKF, CKF and DD2 filters, is to apply the LMMSE estimator for each new observation. The filter performs two operations:

Prediction

Given $p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$, calculate the first two moments of the state distribution at the time of the next unused measurement:

$$\begin{aligned}\hat{\mathbf{x}}_{k|k-1} &= \mathbb{E}[\mathbf{x}_k|\mathbf{Y}_{k-1}] \\ &= \mathbb{E}[f(\mathbf{x}_{k-1},\mathbf{w}_{k-1})|\mathbf{Y}_{k-1}] \qquad (56)\\ \boldsymbol{P}_{k|k-1} &= \mathrm{Cov}(\mathbf{x}_k|\mathbf{Y}_{k-1}) \\ &= \mathbb{E}[f(\mathbf{x}_{k-1},\mathbf{w}_{k-1})f(\mathbf{x}_{k-1},\mathbf{w}_{k-1})^T|\mathbf{Y}_{k-1}] - \hat{\mathbf{x}}_{k|k-1}\hat{\mathbf{x}}_{k|k-1}^T \qquad (57)\end{aligned}$$

Update

Correct the prediction, $\hat{\mathbf{x}}_{k|k-1}$, using the measurement, $\mathbf{y}_k$. The best update that is linear in $\mathbf{y}_k$, is given by the LMMSE estimator [24]:

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \boldsymbol{P}_{\mathbf{xy}}\mathbf{S}_{k|k-1}^{-1}\left(\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1}\right). \qquad (58)$$

The estimator (58) requires knowledge of the mean, $\hat{\mathbf{y}}_{k|k-1}$, and covariance, $\mathbf{S}_{k|k-1}$, of the measurement distribution, as well as the cross-covariance matrix $\boldsymbol{P}_{\mathbf{xy}}$:

$$\begin{aligned}\hat{\mathbf{y}}_{k|k-1} &= \mathbb{E}[\mathbf{y}_k|\mathbf{Y}_{k-1}] \\ &= \mathbb{E}[h(\mathbf{x}_k,\mathbf{v}_k)|\mathbf{Y}_{k-1}] \qquad (59)\end{aligned}$$

$$\begin{aligned}\mathbf{S}_{k|k-1} &= \mathrm{Cov}(\mathbf{y}_k|\mathbf{Y}_{k-1}) \\ &= \mathbb{E}[h(\mathbf{x}_k,\mathbf{v}_k)h(\mathbf{x}_k,\mathbf{v}_k)^T|\mathbf{Y}_{k-1}] - \hat{\mathbf{y}}_{k|k-1}\hat{\mathbf{y}}_{k|k-1}^T. \qquad (60)\\ \boldsymbol{P}_{\mathbf{xy}} &= \mathrm{Cov}(\mathbf{x}_k,\mathbf{y}_k|\mathbf{Y}_{k-1}) \\ &= \mathbb{E}[\mathbf{x}_k h(\mathbf{x}_k,\mathbf{v}_k)^T|\mathbf{Y}_{k-1}] - \hat{\mathbf{x}}_{k|k-1}\hat{\mathbf{y}}_{k|k-1}^T \qquad (61)\end{aligned}$$

The matrix mean squared error (MSE) of the estimate (58) is used as an approximation of the posterior covariance matrix, $\boldsymbol{P}_{k|k}$. The matrix MSE is:

$$\mathbb{E}\left[[\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}][\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}]^T|\mathbf{y}_k\right] = \boldsymbol{P}_{k|k-1} - \boldsymbol{P}_{\mathbf{xy}}\mathbf{S}_{k|k-1}^{-1}\boldsymbol{P}_{\mathbf{yx}}, \qquad (62)$$

and is a reasonable approximation to a posterior covariance matrix which does not depend on the observation $\mathbf{y}_k$. Expressed in terms of the so called gain matrix, $\mathbf{K}_k = \boldsymbol{P}_{\mathbf{xy}}\mathbf{S}_{k|k-1}^{-1}$, the expressions for the state update are:

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1}\right) \qquad (63)$$

$$\boldsymbol{P}_{k|k} = \boldsymbol{P}_{k|k-1} - \mathbf{K}_k\mathbf{S}_{k|k-1}\mathbf{K}_k^T. \qquad (64)$$

To sum up, the filter approximates the first two moments of the posterior distribution, $p(\mathbf{x}_k|\mathbf{Y}_k)$, with the estimate of the mean (63) and the matrix MSE (64), concluding the recursion.

7.3 The marginalized Kalman filter (MKF)

The MKF is the recursive filter following the application of the MT to steps 1–2 in the previous section. The state vector can be augmented to include noise terms, described, e.g., in [7].

MKF prediction

Assume the state vector is Gaussian, i.e.,

$$p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}) = \mathcal{N}(\mathbf{x}_{k-1}; \hat{\mathbf{x}}_{k-1|k-1}, \boldsymbol{P}_{k-1|k-1}).$$

Use the algorithm in Section 5.3 to calculate the mean (56) and covariance (57) of the predictive distribution,

$$p(\mathbf{x}_k|\mathbf{Y}_{k-1}) \approx \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \boldsymbol{P}_{k|k-1}).$$

MKF update

Apply the algorithm a second time to calculate the mean (59) and covariance (60) of the measurement distribution. The cross-covariance matrix (61) is given by equation (45). Calculate the gain matrix, $\mathbf{K}_k = \boldsymbol{P}_{\mathbf{xy}}\mathbf{S}_{k|k-1}^{-1}$, and approximate the posterior distribution

$$p(\mathbf{x}_k|\mathbf{Y}_k) \approx \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k}, \boldsymbol{P}_{k|k}),$$

using the LMMSE estimate (63) and the matrix MSE (64).

8 Simulation examples

The cubature rule is a special case of the unscented transform with the benefit that the estimated covariance matrix is always positive-definite — a property shared also by the proposed method. Further, the results in [9] indicate that the cubature rule performs better than the divided difference filter. Therefore, our main goal is to show how the presented method performs compared to the cubature transform. Two examples are examined: the transformation from polar to Cartesian coordinates, which is also commonly used to illustrate the performance of the unscented transform, and the bearings-only tracking problem [25].

In the first evaluation we use the Kullback-Leibler (KL) discrimination[1] to measure how much a distribution $q(\mathbf{y})$ differs from a reference distribution $p(\mathbf{y})$

[1] Usually referred to as the Kullback-Leibler divergence, although when introduced in [26], the authors used the term "divergence" for the symmetric measure $d_{\mathrm{KL}}(p,q) + d_{\mathrm{KL}}(q,p)$.

[27]:

$$d_{\mathrm{KL}}(p,q) = \int p(\mathbf{y}) \log \frac{p(\mathbf{y})}{q(\mathbf{y})} d\mathbf{y}. \tag{65}$$

This measure was also used in [9] to evaluate the cubature rule, which further motivates using the same approach here. The distributions p and q are approximated as Gaussians, for which $d_{KL}(p,q)$ can be calculated analytically. The first two moments of the reference distribution, p, are estimated using Monte Carlo integration:

$$\int p(x)g(x)dx \approx \sum_{n=1}^{N} g(x_n). \tag{66}$$

Two slightly different versions, the MT^5 and the MT^3, of the presented method are evaluated. The MT^5 is implemented according to the algorithm in Section 5.3, with $p = 5$, using the $2n + 1$ UT sigma-points. However, in order to compare the method fairly to the cubature rule, the MT^3 is introduced, using $p = 3$ and the $2n$ cubature sigma-points. This is not the same as setting $w_0 = 0$ in the second step of the algorithm, which in practice would exclude the point $\mathbf{x}^0$ in the calculation of the mean but not in the calculation of the covariance matrix.

8.1 Polar to Cartesian transformation

In this section the MT^3, using two slightly different priors, is compared to the cubature rule. Let $\mathbf{y} = g(\mathbf{x})$ be the transformation from a polar coordinate system defined in terms of range, r, and azimuth, ψ, to a Cartesian coordinate system:

$$\mathbf{x} = \begin{bmatrix} r \\ \psi \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} x_1 \cos x_2 \\ x_1 \sin x_2 \end{bmatrix}. \tag{67}$$

By modifying the prior, the presented method can be optimized to yield excellent results for a narrow family of transformations. However, this is not a fair comparison and often not a realistic approach. Instead we use the same prior for the 11 positions in Fig. 2, and for each position we evaluate 8 different azimuth measurement noise variances, σ_ψ^2:

$$\sigma_\psi^2 = [5^2, 10^2, 15^2, 20^2, 25^2, 30^2, 35^2, 40^2](\frac{\pi}{180})^2 \quad [\mathrm{rad}^2]. \tag{68}$$

The range measurement noise variance is constant throughout all evaluations, $\sigma_r^2 = 0.5$ [m^2].

To illustrate the influence of the prior, we present results for two different priors, both assuming a zero-mean Gaussian distribution of $\boldsymbol{\theta}$. The first one is created using the simple assumption that the function is a 2$^{\mathrm{nd}}$ order polynomial

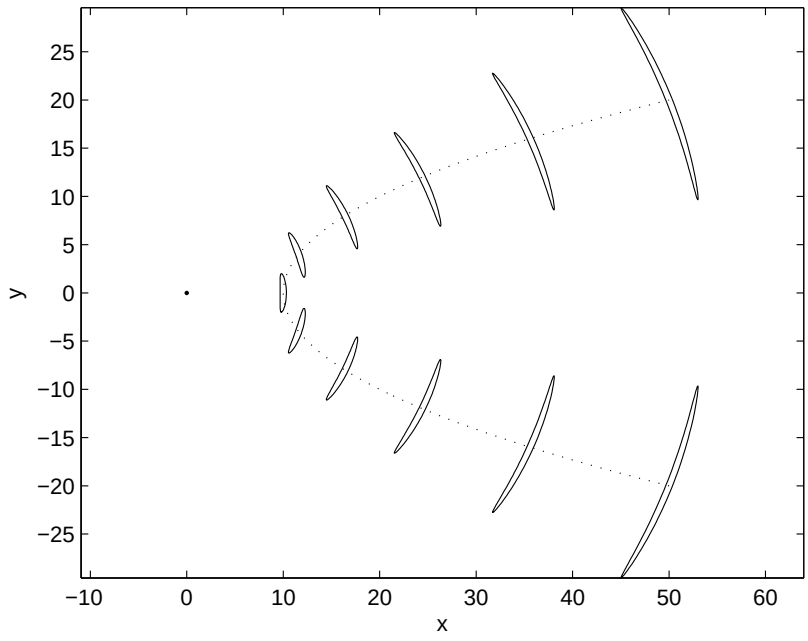


Figure 2: A sensor, situated in the origin, with uncertainties in range and angle measurements observes a target at eleven positions. The "banana-shaped" contours are measurement space covariance contours, transformed to the Cartesian coordinate system.

where the higher order term is relatively small, whereas the second one has been numerically derived to perform well in this scenario:

$$\mathbf{\Sigma}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{100} & 0 \\ 0 & 0 & 0 \end{bmatrix}, \; \mathbf{\Sigma}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.036 & 0 \\ 0 & 0 & 0.0007 \end{bmatrix}. \tag{69}$$

The cubature evaluation points are used by all three methods and, as argued in Section 6.2, the prior variance for the mean, θ_0, does not influence the estimate.

The average Kullback-Leibler discrimination is presented in Table 1 and the mean for each position and noise variance is displayed in Fig. 3. The reference density was calculated using 10^5 samples. The results show that, although all methods perform very well in absolute numbers, the marginalized sigma-point estimator outperforms the Cubature rule using the same points χ. It can also be seen that Σ_1 is the better description for some noise models, and for position 6, but that Σ_2 performs better on average.

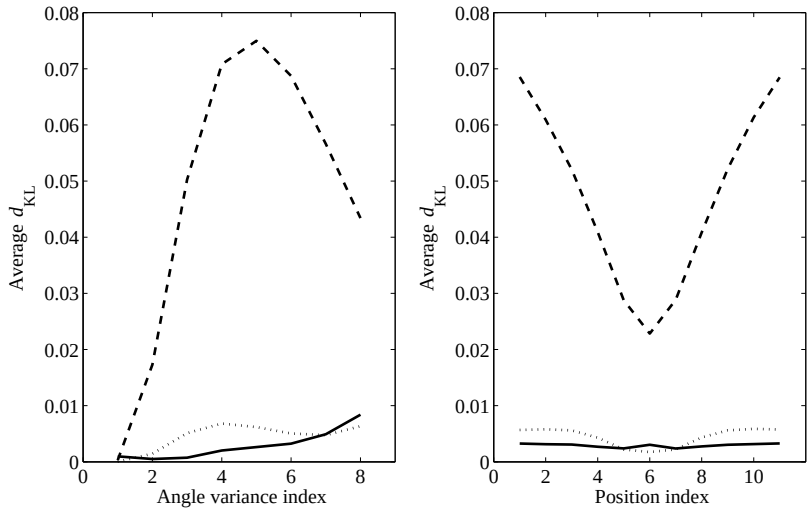


Figure 3: The left figure shows the average Kullback-Leibler discrimination for the different azimuth noise variances, whereas the right figure shows the average Kullback-Leibler discrimination for the positions. The dashed line illustrates the Cubature rule, the dotted line represents the use of $\mathbf{\Sigma}_1$, and the solid line the use of $\mathbf{\Sigma}_2$.

8.2 Bearings only tracking

The bearings only tracking problem is well-studied and arises in passive sensor applications such as sonar tracking. Several filters have been designed for this particular task, such as the range-parameterized EKF [25], but since we are interested in comparing sigma-point filters, those filters are not included in the comparison. Two MKF versions, based on the MT^5 and the MT^3, are compared to the CKF, the UKF and the DD2-filter.

The scenario we consider here, tracking of a non-maneuvering submarine, is illustrated in Fig. 4. Most parameter values are taken from [25]. The state vector contains the Cartesian position and velocity, $\mathbf{x} = [x\ y\ \dot{x}\ \dot{y}]^T$, and bearing

Table 1: Average Kullback-Leibler discrimination

	Average KL-discrimination [$\times 10^{-4}$]
Cubature rule	478
MT^3, $\mathbf{\Sigma}_1$	45
MT^3, $\mathbf{\Sigma}_2$	29

observations are non-linear transformations of $\mathbf{x}$, with additive Gaussian noise:

$$\theta = \tan^{-1}\left(\frac{y}{x}\right) + w. \tag{70}$$

The variance of the measurement noise, $w_k \sim \mathcal{N}(0, \sigma_w^2)$, is known to the tracking algorithms, which are also given perfect knowledge of the prior distribution; for each simulation, the initial position of the target is generated from the prior. The process model is linear:

$$\mathbf{x}_k = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{k-1} + \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix} \mathbf{v}_k, \tag{71}$$

with process noise, $\mathbf{v}_k \sim \mathcal{N}(0, \sigma_{\mathbf{v}}^2 \mathbf{I}_{2\times 2})$. The state distribution is assumed Gaussian, and the predicted distribution, which is consequently also Gaussian, is correctly calculated by all five filters. Hence, the methods differ only in the calculation of the measurement distribution and the cross-covariance matrix. The parameter values are:

$$\sigma_{\mathbf{v}} = \sqrt{10^{-5}}\,\frac{m}{s^2}, \quad \sigma_w = 1.5\,^\circ, \quad T = 60\,s, \quad N = 30,$$

where N is the length of a trajectory. The filter is initiated using the scheme in [25], at

$$\mathbf{x}_0 = \begin{bmatrix} 3000 \\ 4000 \\ -0.6 \\ -0.8 \end{bmatrix}, \quad \boldsymbol{P}_0 \approx \begin{bmatrix} 592^2 & 682^2 & 0 & 0 \\ 682^2 & 816^2 & 0 & 0 \\ 0 & 0 & 0.57 & -0.35 \\ 0 & 0 & -0.35 & 0.34 \end{bmatrix},$$

which corresponds to a target at a range of 5 km, traveling towards the sensor at a speed of 1 m/s with uncertainties in range ($\sigma_r = 1000$ m), speed ($\sigma_s = 0.3$ m/s) and course ($\sigma_c = \frac{\pi}{\sqrt{12}}$ rad).

Two performance measures are averaged over 10^4 simulations: The MSE, ξ, and the average normalized estimation error squared (NEES), ζ:

$$\xi = \frac{1}{N}\sum_{k=1}^{N} [\hat{\mathbf{x}}_k^p - \mathbf{x}_k^p]^T [\hat{\mathbf{x}}_k^p - \mathbf{x}_k^p] \tag{72}$$

$$\zeta = \frac{1}{N}\sum_{k=1}^{N} [\hat{\mathbf{x}}_k^p - \mathbf{x}_k^p]^T \left(\hat{\boldsymbol{P}}_{\mathbf{k}}^{\mathbf{p}}\right)^{-1} [\hat{\mathbf{x}}_k^p - \mathbf{x}_k^p] \tag{73}$$

Both are calculated for the position states, $\mathbf{x}^p = [x\, y]^T$, and its covariance matrix, $\boldsymbol{P}^p$. The results are summarized in Table 2. When the posterior covariance matrix correctly describes the estimation error, the NEES is equal to the number of dimensions of the evaluated state vector.

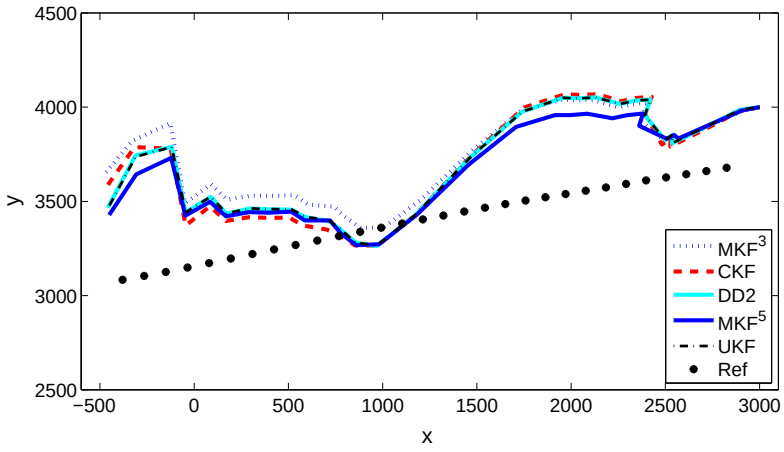


Figure 4: Five different filters are applied to the tracking problem where a bearings-only sensor, situated in the origin, makes 30 observations of a moving target. In this particular example the target process noise is near-zero.

Table 2: RMSE and NEES, averaged over 10^4 simulations

	RMSE, $\sqrt{\bar{\xi}}$	NEES, $\bar{\zeta}$	Number of sigma-points
MKF^3	1074	1.97	$2n$
CKF	1083	2.46	$2n$
DD2	1077	2.40	$2n+1$
MKF^5	1064	2.01	$2n+1$
UKF	1076	2.40	$2n+1$

In this evaluation, the UT mean weight, w_0, is $1-\frac{n}{3}$, the DD2 parameter, h, is $\sqrt{3}$, and the MKF^3 and MKF^5 are based on the MT^3 and MT^5, respectively[2], with priors:

$$\mathbf{\Sigma}_{\text{MT}^3} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{10} & 0 \\ 0 & 0 & \frac{5}{100} \end{bmatrix}, \quad \mathbf{\Sigma}_{\text{MT}^5} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{10} & 0 & 0 & 0 \\ 0 & 0 & \frac{5}{100} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{1000} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{1000} \end{bmatrix}.$$

From Table 2 we conclude that the choice of filters does not, on average, affect the MSE in particular and that the CKF, UKF and DD2 underestimate the size of the

[2]In other words, the MKF^3 and the MKF^5 estimates of the mean are calculated using the same rules as the CKF and the UKF/DD2, respectively.

error (recall that, nominally, $\zeta = 2$). The MKF, however, performs very well in this sense. This can be explained in terms of the posterior uncertainties regarding $\boldsymbol{\theta}$, which contribute to the covariance matrix estimate through the additive diagonal matrix in equation (35).

9 Conclusions

We have presented a derivative-free method, the marginalized transform (MT), for estimating the mean and covariance of a transformed Gaussian-distributed random variable, which has several beneficial properties. In summary, the method:

- performs better than well-known sigma-point methods, such as the UT, DD2, or cubature rule, in the evaluated estimation task and the bearings-only tracking scenario.
- is easy to apply, as the simplicity of derivative-free filters is maintained.
- has tuning-parameters that can be intuitively understood in terms of the model of the transforming function.

In a more general sense, we present a method for designing sigma-point estimators, based on explicit model assumptions. For example, it has been shown which assumptions lead to the integration rules of the DD2, UT, and the cubature rule.

Sigma-point filters have previously been analyzed in terms of the precision of the applied integral approximation. Still, as the non-linear functions encountered in most applications are not polynomial, we argue that it is relevant to ask what the estimates represent when they are *not* exact. A description of the latter is precisely what the MT gives; the family of functions contributing to the estimates.

Appendix A: UT covariance matrix estimates

The UT covariance matrix estimate (8) is on the form

$$\hat{\boldsymbol{P}}_y = \sum_{i=0}^{2n} w_i \mathbf{d}_i \mathbf{d}_i^T, \text{ with } \mathbf{d} = [g(\mathbf{x}^i) - \bar{\mathbf{y}}]. \tag{74}$$

Lemma 1: The covariance matrix estimate calculated by the UT is guaranteed to be positive-semidefinite when all weights are positive.

Proof: $\hat{\boldsymbol{P}}_y$ is positive-semidefinite if $\mathbf{x}^T \hat{\boldsymbol{P}}_y \mathbf{x} \geq 0$, and

$$\mathbf{x}^T \hat{\boldsymbol{P}}_y \mathbf{x} = \sum_{i=0}^{2n} w_i (\mathbf{x}^T \mathbf{d}_i)^2 \geq 0, \text{ if } w_i \geq 0 \, \forall i \tag{75}$$

Lemma 2: When $w_0 \notin [0, 1]$, there are functions for which $\hat{\boldsymbol{P}}_y$ is not positive-semidefinite.

Proof: For example, there exists a function $g : \mathbb{R}^n \to \mathbb{R}^1$, symmetric such that

$$a = g(\mathbf{x}^i)\,, i \in \{1, \dots 2n\} \tag{76}$$

$$b = g(\mathbf{x}^0). \tag{77}$$

The UT weights sum to one and $\bar{\mathbf{y}}$ is assumed zero,

$$w_0 b + 2n w_i a = 0, \tag{78}$$

leading to the following two relations:

$$w_i = \frac{1 - w_0}{2n} \triangleq w, \text{ and } a = b\frac{-w_0}{1 - w_0}. \tag{79}$$

The variance is negative if,

$$\begin{aligned} \sigma_y^2 = w_0 b^2 + 2nwa^2 < 0 \\ \Leftrightarrow w_0 b^2 + (1 - w_0) b^2 \frac{(-w_0)^2}{(1 - w_0)^2} < 0 \\ \Leftrightarrow w_0 (1 - w_0)^2 + (1 - w_0) w_0^2 < 0. \end{aligned} \tag{80}$$

The left hand side on the last row is a second order polynomial with roots $w_0 = 0$ and $w_0 = 1$, and a maximum in $w_0 = 1/2$. In other words:

$$w_0 \notin [0, 1] \Rightarrow \sigma_y^2 < 0. \tag{81}$$

Each diagonal element in the $m \times m$ covariance matrix, corresponding to $g : \mathbb{R}^n \to \mathbb{R}^m$, is calculated analogous to σ_y^2. The proof is therefore valid for any dimensionality.

Appendix B: Properties of Hermite polynomials

The univariate Hermite polynomials are orthogonal under integration under the Gaussian pdf, i.e., for $x \sim \mathcal{N}(0, 1)$,

$$\mathbb{E}[H_i(x) H_j(x)] = \int p(x) H_i(x) H_j(x) dx = \begin{cases} 0 & i \neq j \\ i! & i = j \end{cases}. \tag{82}$$

It follows that the expected value is zero for all but the 0^{th} polynomial:

$$\mathbb{E}[H_i(x)] = \int p(x) H_i(x) H_0(x) dx = \begin{cases} 0 & , i \neq j \\ 1 & , i = 0 \end{cases}. \tag{83}$$

Further, we conclude that, for $[x_1, \dots, x_n]^T \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{n \times n})$,

$$\mathbb{E}[H_i(x_k) H_j(x_l)] = \int p(\mathbf{x}) H_i(x_k) H_j(x_l) d\mathbf{x} = \begin{cases} 0 & , i \neq j \cup k \neq l \\ 1 & , i = j = 0 \, \forall k, l \\ i! & , i = j \cap k = l \end{cases}, \tag{84}$$

which follows from (82), (83). A simple formula expressing the Hermite polynomials in terms of a random variable $\nu \sim \mathcal{N}(0,1)$ was given in [28]:

$$H_n(x) = \mathbb{E}\left[(x + \nu\sqrt{-1})^n | x\right]. \tag{85}$$

The first six Hermite polynomials are

$$\begin{aligned} H_0(x) &= 1, & H_2(x) &= x^2 - 1, & H_4(x) &= x^4 - 6x^2 + 3 \\ H_1(x) &= x, & H_3(x) &= x^3 - 3x, & H_5(x) &= x^5 - 10x^3 + 15x. \end{aligned}$$

Scaling the Hermite polynomials to achieve orthogonality when $\sigma_x \neq 1$ is achieved by dividing the argument with the standard deviation: $H_i(x/\sigma_x)$. Expressions for multivariate Hermitian polynomials are described in [28], offering the possibility to extend the framework to model also terms not represented by the univariate Hermite polynomials, i.e., products on the form $y = \prod_{i=1}^{n} x_i^{\kappa_i}$, for $\kappa_i \in \{0, 1, 2, \dots\}$.

Appendix C: The sigma-point selection scheme

The uncertainties in the estimate of the mean are described by equation (48). It is zero if $\mathbf{H}\boldsymbol{P}_\theta\mathbf{H}^T$ is invertible and there exists a vector $\boldsymbol{\lambda}$ such that

$$\mathbf{H}^T(\chi)\boldsymbol{\lambda} = \mathbf{w}, \tag{86}$$

with $\mathbf{w} = [1,\ 0, \dots,\ 0]^T$. As we shall see, the sigma-point selection scheme (4) - (5) always attains the relation (86).

For $x \sim \mathcal{N}(0,1)$ the sigma-points are $\chi = [0, \sqrt{3}, -\sqrt{3}]$ and the observation matrix for Hermite polynomials up to order 5 is:

$$\mathbf{H}^T(\chi) = [\mathbf{h}(0),\ \mathbf{h}(\sqrt{3}),\ \mathbf{h}(-\sqrt{3})] = \begin{bmatrix} 1 & 1 & 1 \\ 0 & \sqrt{3} & -\sqrt{3} \\ -1 & 2 & 2 \\ 0 & 0 & 0 \\ 3 & -6 & -6 \\ 0 & -6\sqrt{3} & 6\sqrt{3} \end{bmatrix}. \tag{87}$$

For $\boldsymbol{\lambda} = [\lambda_0, \lambda_1, \dots]^T$ to solve equation (86) we see that:

$$\begin{aligned} 1: &\quad \textstyle\sum_{i=0}^{2n} \lambda_i = 1 && \text{(from row one)} \\ 2: &\quad \lambda_i = \lambda_j,\ \forall i, j \neq 0 && \text{(from row two and six)} \\ 3: &\quad \lambda_0 = 4\lambda_i,\ i > 0 && \text{(from row three and five)} \end{aligned} \tag{88}$$

When the dimensionality of $\mathbf{x}$ increases, no unique elements are added to $\mathbf{H}^T$. When $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{n\times n})$:

$$\mathbf{H}^T(\chi) = \begin{bmatrix} \mathbf{h}(0) & \mathbf{h}(\sqrt{3}) & \mathbf{h}(-\sqrt{3}) & \mathbf{h}(0) & \mathbf{h}(0) & \dots \\ \mathbf{h}(0) & \mathbf{h}(0) & \mathbf{h}(0) & \mathbf{h}(\sqrt{3}) & \mathbf{h}(-\sqrt{3}) & \ddots \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots \end{bmatrix}.$$

The third requirement is therefore adjusted to suit the multidimensional case: $\lambda_0 = (6 - 2n)\lambda_i$. Substituting λ_i with w_i, these are exactly the criterions (4) - (5), with $w_0 = 1 - n/3$. The observation matrix associated with the cubature sigma-point selection scheme enjoys the same properties (for $p \leq 3$).

References

[1] L. de Menezes, A. Soares, F. Silva, M. Terada, and D. Correia, "A new procedure for assessing the sensitivity of antennas using the unscented transform," *IEEE Trans. Antennas Propag.*, vol. 58, no. 3, pp. 988 –993, March 2010.

[2] C.-L. Su, "Probabilistic load-flow computation using point estimate method," *IEEE Trans. Power Syst.*, vol. 20, no. 4, pp. 1843 – 1851, Nov. 2005.

[3] G. Steiner, H. Zangl, and D. Watzenig, "Generic statistical circuit design based on the unscented transformation and its application to capacitive sensor instrumentation," in *IEEE Int. Conf. on Ind. Technology*, Dec. 2005, pp. 108 –113.

[4] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. of the ASME – Journal of Basic Eng.*, vol. 82, no. Series D, pp. 35–45, 1960.

[5] K. Ito and K. Xiong, "Gaussian filters for non linear filtering problems," *IEEE Trans. Automatic control*, vol. 45, pp. 910–927, 2000.

[6] M. Norgaard, N. K. Poulsen, and O. Ravn, "New developments in state estimation for nonlinear systems," *Automatica*, vol. 36, no. 11, pp. 1627–38, 2000.

[7] S. Julier and J. Uhlmann, "Unscented filtering and nonlinear estimation," *Proc. IEEE*, vol. 92, no. 3, pp. 401 – 22, 2004.

[8] I. Arasaratnam, S. Haykin, and R. J. Elliott, "Discrete-time nonlinear filtering algorithms using Gauss-Hermite quadrature," *Proc. IEEE*, vol. 95, no. 5, pp. 953–977, 2007.

[9] I. Arasaratnam and S. Haykin, "Cubature Kalman filters," *IEEE Trans, Autom. Control*, vol. 54, no. 6, pp. 1254–1269, 2009.

[10] J. Sarmavuori and S. Särkkä, "Fourier-Hermite Kalman Filter," to appear in IEEE Trans. Autom. Control. [Online]. Available: http://www.lce.hut.fi/ ssarkka/pub/fhkf-ieee.pdf

[11] S. J. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte, "A new approach for filtering nonlinear systems," in *Proc. American Control Conference*, vol. 3, 1995, pp. 1628–1632 vol.3.

[12] U. Lerner, "Hybrid Bayesian Networks for Reasoning About Complex Systems," Ph.D. dissertation, Stanford University, 2002.

[13] J. McNamee and F. Stenger, "Construction of fully symmetric numerical integration formulas," *Numerische Mathematik*, vol. 10, no. 4, pp. 327–344, 1967.

[14] Y. Wu, D. Hu, M. Wu, and X. Hu, "A numerical-integration perspective on Gaussian filters," *IEEE Trans. Signal Processing*, vol. 54, no. 8, pp. 2910–21, 2006.

[15] C. Fernández-Prades and J. Vilà-Valls, "Bayesian Nonlinear Filtering Using Quadrature and Cubature Rules Applied to Sensor Data Fusion for Positioning," in *IEEE Int. Conf. on Communications*, 2010, pp. 2–6.

[16] K. Pakki, B. Chandra, G. Da-Wei, and I. Postlethwaite, "Cubature Kalman Filter based Localization and Mapping," in *Preprints of the 18th IFAC World Congress, Milano, Italy*, 2011, pp. 2121–2125.

[17] M. P. Deisenroth, M. F. Huber, and U. D. Hanebeck, "Analytic moment-based Gaussian process filtering," in *Proc. of the 26th Annual Int. Conf. on Machine Learning*, ser. ICML '09. New York, NY, USA: ACM, 2009, pp. 225–232.

[18] J. Ko and D. Fox, "GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models," *Autonomous Robots*, vol. 27, pp. 75–90, 2009.

[19] A. O'Hagan, "Bayes-Hermite quadrature," *Journal of Statistical Planning and Inference*, vol. 29, no. 3, pp. 245–260, Nov. 1991.

[20] C. P. Robert, *The Bayesian Choice: From Decision-Theoretic Foundations to Computational Implementation*. Springer, 2007.

[21] S. M. Kay, *Fundamentals of statistical signal processing - estimation theory*. Prentice-Hall, Inc., 1993.

[22] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis, Second Edition*, 2nd ed. Chapman and Hall/CRC, Jul. 2003, ch. 2.9.

[23] J. Uhlmann, "Simultaneous map building and localization for real time applications," Transfer thesis, Univ. Oxford, Oxford, U.K., 1994.

[24] Y. Bar-Shalom, X. Rong Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., 2001.

[25] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman filter: particle filters for tracking applications*. Artech House, 2004, ch. 6.

[26] S. Kullback and R. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, pp. 79–86, 1951.

[27] A. R. Runnalls, "Kullback-Leibler Approach to Gaussian Mixture Reduction," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 43, no. 3, pp. 989–999, Jul. 2007.

[28] C. Withers, "A simple expression for the multivariate Hermite polynomials," *Statistics & Probability Letters*, vol. 47, no. 2, pp. 165–169, Apr. 2000.

Paper II

A probabilistic framework for decision-making in collision avoidance systems

M. Brannström, F. Sandblom and L. Hammarstrand

Submitted to *IEEE Transactions on Intelligent Transportation Systems*.

A Probabilistic Framework for Decision-Making in Collision Avoidance Systems

Mattias Brännström, Fredrik Sandblom, and Lars Hammarstrand

Abstract

This paper is concerned with the problem of decision-making in systems aiming at assisting drivers in avoiding collisions with other road users and objects. An important aspect of these systems is not only to assist the driver when needed, but also not to disturb the driver with unnecessary interventions. A probabilistic framework is presented for jointly evaluating the driver acceptance of an intervention and the necessity thereof to automatically avoid a collision. The driver's acceptance is modeled by assuming that drivers make predictions of the future trajectory of road users and assesses these trajectories to judge if a traffic situation is critical, or if a collision safely can be avoided with reasonable effort. The framework is evaluated on a few different types of collision scenarios, using both simulated and authentic sensor measurements. The results show that it enables earlier interventions, and hence increased system benefit, especially in situations where it is difficult for the driver to predict the future trajectory of an object.

1 Introduction

Road traffic accidents are one of the world's largest public health problems. In the EU alone, traffic accidents cause approximately 1.8 million injuries and 43,000 fatalities each year [1]. To reduce these numbers, vehicle manufactures are developing systems that can detect hazardous traffic situations and actively assist road users in avoiding or mitigating accidents. Systems that assist drivers in avoiding collisions are becoming increasingly more common and are even being introduced as standard equipment in some passenger cars [2].

Collision avoidance (CA) systems can generally be divided into three layers, as illustrated in Fig. 1. Measurements from on-board sensors, such as accelerometers, cameras and radars, are processed in the first layer and then interpreted in the second layer that makes decisions on when and how to assist the driver. The third layer executes the decision, e.g. by automatically applying the brakes of the vehicle.

The measurements are associated with uncertainties and, consequently, as are the vehicle and object state estimates that are obtained in the sensor fusion layer [3]. A threat assessment algorithm utilizes these estimates to make predictions of road user trajectories. Based on these predictions, an assessment is performed to estimate if and how a potential accident can be prevented [4, 5]. Both the state estimates and the predictions are

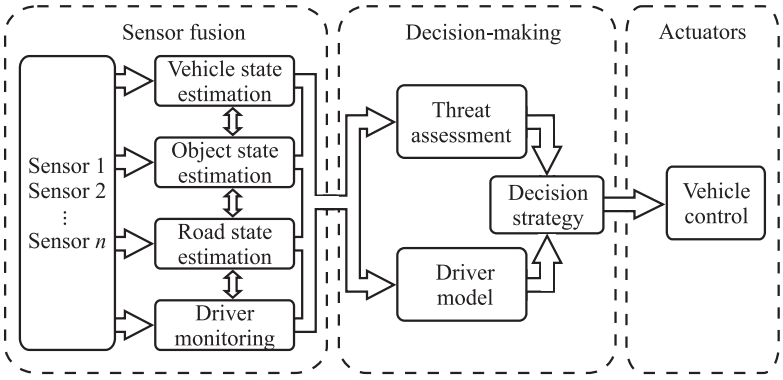


Fig. 1: Collision avoidance systems can be divided into three layers. This paper focuses on decision-making, threat assessment and modeling of the driver's perception of the traffic situation.

associated with uncertainties which need to be treated properly [6]. Moreover, to obtain a high customer acceptance it is important that the system also accounts for the preferences of driver, such that the driver is not disturbed by the system during normal driving conditions [7, 8].

The aim of CA systems is to assist drivers in avoiding collisions with other road users and objects, without triggering interventions that the driver may consider as unnecessary. Clearly, in order to autonomously avoid a collision, an intervention must be triggered while an accident is still avoidable. However, most collisions that can be avoided by CA systems *possibly* can also be avoided by a skilled driver. This implies that there is no way of telling whether the driver would have been able to handle the situation without the system intervening. That is, there is no objective measure available for judging whether an intervention was useful or whether the driver was disturbed by the system. It is only the driver of the vehicle that can decide whether an intervention was motivated or not. Hence, when making decisions on when the system shall take action, there is always a trade-off between making a successful intervention and the risk of disturbing the driver. This highlights the need of incorporating a driver model in the CA system, such that vehicle safety can be further improved by enabling earlier interventions in traffic situations that the driver judges as critical.

This paper is concerned with decision-making in CA systems in the presence of uncertainties. Several probabilistic decision-making algorithms have previously been proposed, e.g., [4, 6, 9–11]. In these approaches, the risk of a false alarm is balanced with the confidence that an autonomous intervention is actually needed to avoid a collision. The term "false alarm" is, in these cases, typically defined as an intervention performed even though there would not have been a collision. From the driver's point of view, however, a more relevant definition would be an intervention performed that the driver regards as unnecessary. In the present paper, we propose to simultaneously evaluate the driver acceptance of an intervention and the necessity thereof, as is illustrated in the decision-making layer in

Fig. 1. Decisions on when to assist the driver are made by taking a Bayesian approach to estimate:

1. how a collision can be avoided by an intervention, and
2. the probability that the driver of the vehicle will consider the autonomous intervention as motivated.

Interventions are initiated at an earlier stage when it is estimated that the driver acceptance for interventions is high. In this way, the benefit of CA systems can be increased without disturbing the driver. Moreover, to reduce the risk of the driver getting used to having interventions, and thus may start relying too much on the system, interventions are not initiated until a significant action is needed to avoid a collision.

As an example, a previously presented threat assessment algorithm is used as a base to assess how a collision with a single road user can be avoided [12]. Similar to [6], the proposed framework enables this assessment to be performed in a probabilistic way, such that uncertainties in object state estimates and predictions are accounted for. In order to take the driver's preferences into account, a novel driver model is formulated under a few basic assumptions regarding driver behavior, although it should be noted that the proposed framework also supports other driver models and a wide variety of threat assessment algorithms. Specifically, the proposed driver model is built on an assumption that some of the driver's safety margins originate from the driver's perceived uncertainty in the current and future state of road users. To put it differently, the driver's desired safety margins are modeled in a similar way to the safety margins that CA systems or autonomous vehicles may use in order to safely avoid accidents. Including a model of the driver's safety margins into a CA system enables the system to perform earlier interventions in situations where the future trajectories of road users are difficult for the driver to predict, e.g., when a wild animal enters the road.

The paper is organized as follows. Section 2 provides a motivation for this work and shows that a small change in the decision timing can have a significant impact on the benefit of CA systems. Section 3 outlines the problem formulation, whereas Section 4 describes the decision-making framework. Section 5 presents the modeling choices made in the specific implementation used to evaluate the framework in this paper. Section 6 describes, based on these modeling choices, how the decision-making framework can be realized. Results are presented in Section 7, where the realization is evaluated on a few different types of collision scenarios, using both simulated and authentic sensor measurements. It is shown that intervention timing can be improved by using a driver model, e.g., by taking a human centric approach to probabilistic decision-making. Conclusions are presented in Section 8.

2 Motivation

This section describes how the present paper is related to previous research on driver models in CA systems and shows, through an example, that system benefit is strongly affected by the timing of interventions.

2.1 Related literature

Goodrich and Boer [7] propose that CA systems should account for not only the capabilities of the vehicle and the sensor system, but also the autonomy and preferences of the driver. Benefit and cost functions are introduced to make decisions based on a trade-off between the potential benefit of an intervention and the cost of disturbing the driver with an unnecessary intervention. Although the concept of using cost functions is appealing at first glance, this concept has some potential drawbacks that are pointed out, e.g., in [4]. For example, the cost of an unnecessary intervention may be difficult to define and relate to the benefit of avoiding a potential collision. Hence, the behaviour of the CA system may be hard to predict and tuning of the system could become problematic.

McCall and Trivedi [13] propose that the probability that the driver intends to apply the brakes shall be estimated and that interventions shall be inhibited if the driver intends to brake. The intent to brake is predicted by using a camera that monitors the driver's pedal usage and a camera that monitors the driver's face. Although a foot camera may be used to predict whether the driver intends to apply the brakes, it is probably difficult to predict whether the driver intends to steer. The driver may also be drowsy or cognitively distracted, in which case driver intent could be difficult to predict, even if the driver has placed a foot on the brake pedal. Moreover, in traffic situations where the driver intends to brake to avoid a severe collision, it is reasonable to assume that the driver may consider a brake intervention as motivated and thus not disturbing.

Rather than solving the difficult problem of estimating the drivers's intent or the cost of disturbing the driver, the driver model presented in this paper aims only to estimate whether the driver will consider an intervention as motivated, as mentioned in Section 1. In this way, driver autonomy can be maintained by only allowing the system to act when it is estimated that the driver has a high acceptability for interventions.

2.2 Impact

As mentioned in Section 1, one of the aims of the present paper is to formulate a decision-making framework that can be used to improve the intervention timing of CA systems. What additional benefit is expected if the intervention timing is improved by a fraction of a second? This question is investigated through an example.

Assume that a vehicle equipped with a CA system is driving at an initial speed v_0, and that automatic emergency braking is initiated with a timing such that the speed is reduced to v_c before colliding with a stationary object. How much earlier does the braking have to be initiated in order to fully avoid a collision? In all CA systems the deceleration will eventually reach a constant value a and a collision is avoided if

$$s = \frac{v_\mathrm{c}^2}{2\,a} \tag{1}$$

extra meters extra meters are available available to bring the host vehicle to a complete stop. The time to travel this distance, before braking is initiated, is

$$\Delta t = \frac{s}{v_0} = \frac{v_\mathrm{c}^2}{2\,a\,v_0}. \tag{2}$$

Consequently, by applying the brakes Δt seconds earlier, the required extra distance is gained and the collision is avoided. The required timing change Δt as a function of the initial velocity v_0 is illustrated in Fig. 2.

For example, given the initial speed $v_0 = 15\,\text{m/s}$, the reduced impact speed $v_\text{c} = 4\,\text{m/s}$ (i.e., 54 km/h and 14 km/h) and that the vehicle is driven on dry asphalt ($a = 10$ m/s^2), it is enough to brake $\Delta t \approx 0.05\,\text{s}$ earlier in order to fully avoid an accident. To put this time into perspective, we will compare this time with the total intervention timing required for avoiding a collision. Assume that the autonomous brake system has an initial delay t_d before braking is applied, and that it takes an additional time t_r before full braking capacity a is reached. Furthermore, let t_TTC be the time-to-collision given that the vehicle speed is constant and braking is not applied. Then a collision is fully avoided if braking is initiated at

$$t_\text{TTC}^\text{avoid} = t_\text{d} + \frac{t_\text{r}}{2} + \frac{v_0}{2\,a} \tag{3}$$

whereas the vehicle collides with an impact speed

$$v_\text{c} = \sqrt{2\,a\,v_0\,\Delta t} \tag{4}$$

if the braking is delayed Δt s. For a typical CA system $t_\text{d} \approx 0.05$ s and $t_\text{r} \approx 0.5$ s, which in the example above gives that a collision is fully avoided if braking is initiated at $t_\text{TTC}^\text{avoid} = 1.05$ s whereas the vehicle collides at $v_c = 4$ m/s if braking is applied at $t_\text{TTC}^\text{mitigate} = t_\text{TTC}^\text{avoid} - \Delta t \approx 1.00$ s.

Apparently, the required difference in timing to attain avoidance rather than mitigation can be very small. The reason for this is that the required distance s to bring the host vehicle to a complete stop from the reduced velocity v_c is relatively short once a constant deceleration a has been reached, see (1). Consequently, the time Δt it takes to travel the short distance s before braking is initiated is small and decreases with increasing initial speed v_0, see (2). If the intervention timing can be improved just by a fraction of a second there is a high impact on the benefit of the system.

3 Problem formulation

Let $\mathbf{x}_k^\text{h}$ be a state vector representing the state, i.e., position, velocity, etc., at discrete time-instance t_k of a vehicle hosting a CA system. Similarly, let $\mathbf{x}_k^\text{t}$ be a state vector describing the state of all other objects of interest, e.g., other vehicles or pedestrians in the vicinity of the host vehicle. Given, noisy observations on these states collected up to the current time t_k, denoted $\mathbf{Y}_{1:k}$, an intervention decision rule for avoiding accidents without disturbing the driver is desired. This decision rule should take into consideration that the state vectors themselves are are not known to the CA system, but rather only their posterior probability density function (pdf) $p(\mathbf{x}_k^\text{h}, \mathbf{x}_k^\text{t} | \mathbf{Y}_{1:k})$ calculated by the sensor fusion system.

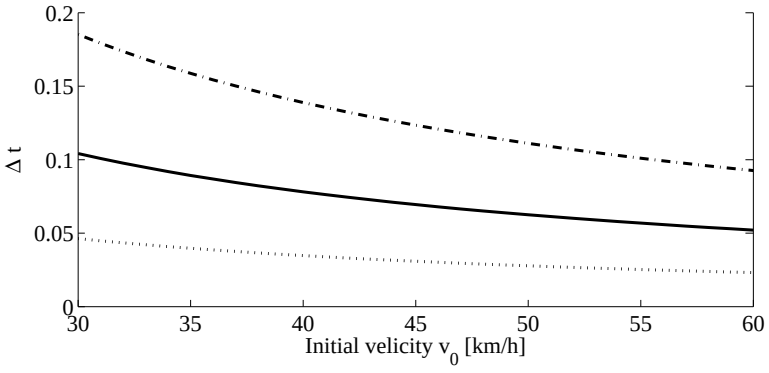


Fig. 2: The graph shows how much earlier the intervention of a CA system must be triggered in order to fully avoid a collision instead of colliding at an impact velocity v_c. The three curves show the required timing improvement Δt for three different crash velocities $v_c = 20,\ 15$, and 10 km/h, top to bottom. Braking Δt seconds earlier fully avoids a collision with a stationary target by the smallest possible margin. The result is valid for braking on dry asphalt and is invariant to system time delays t_d and brake system ramp-up times t_r within the range of most CA systems.

4 Probabilistic decision-making

This section describes a probabilistic decision-making framework, and aims at at answering the following questions:

1. How can the risk of disturbing the driver be modeled?

2. What decision strategy is suitable when trajectory predictions of road users are associated with uncertainties?

A robust decision rule in a CA system must treat several sources of uncertainties. Firstly, there is an inherent uncertainty in the states $\mathbf{x}_k^\mathrm{h}$ and $\mathbf{x}_k^\mathrm{t}$ as they are observed using noisy and imperfect measurements. Secondly, to assess the danger in a situation, the system needs to make predictions regarding the unknown future. Both of these uncertainty sources, illustrated in Fig. 3, can be addressed by using a probabilistic approach to decision-making. As previously described in Section 1, we propose to simultaneously evaluate the driver acceptance of an intervention and the necessity thereof. Furthermore, we propose to do so in a probabilistic decision framework considering two sets of hypotheses. The first, capturing the driver acceptance (*driver hypotheses*). The second, describing the necessity for the CA system to initiate an autonomous intervention to avoid an accident (*criticality hypotheses*). This latter hypothesis set is used to manage the risk that the driver gets used to having interventions and hence may start to rely on that the

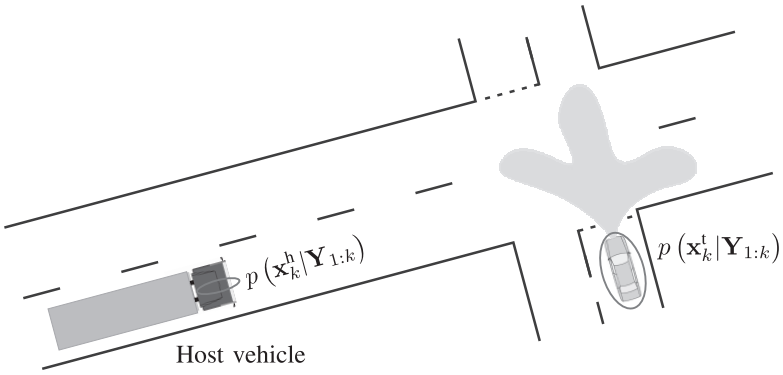


Fig. 3: When making intervention decisions, the proposed framework treats uncertainties both in state estimates and in obstacle trajectories over a prediction horizon, as indicated by the light grey area in front of the car. The vehicle hosting the CA system can be represented by any type of automotive vehicle, e.g. a car, a motorcycle, or a truck, as illustrated in this scenario.

system will avoid all accidents. The two hypothesis sets are denoted

$$\begin{cases} \mathcal{H}_0^{\mathrm{d}} & : \text{Driver does not accept an intervention} \\ \mathcal{H}_1^{\mathrm{d}} & : \text{Driver accepts an intervention} \end{cases} \tag{5}$$

$$\begin{cases} \mathcal{H}_0^{\mathrm{c}} & : \text{An immediate CA intervention is not needed} \\ \mathcal{H}_1^{\mathrm{c}} & : \text{An immediate CA intervention is needed} \end{cases} \tag{6}$$

where the CA system only takes action if both $\mathcal{H}_1^{\mathrm{d}}$ is selected over $\mathcal{H}_0^{\mathrm{d}}$ and $\mathcal{H}_1^{\mathrm{c}}$ is selected over $\mathcal{H}_0^{\mathrm{c}}$. If the probability of these hypotheses can be calculated based on the information available, $\Pr\{\mathcal{H}_i^{\mathrm{d}}|\mathbf{Y}_{1:k}\}$ and $\Pr\{\mathcal{H}_i^{\mathrm{c}}|\mathbf{Y}_{1:k}\}$, the decision rule detailed in Appendix A can be used to choose one hypothesis over the other, see (44). In the following sections we derive the framework needed to make these decisions.

4.1 Threat assessment

A vital component in a collision avoidance system is the ability to detect and assess the danger in the current traffic situation, i.e., *threat assessment*. In the literature there are many different approaches to do this and good examples can be found in, e.g., [4, 10, 12, 14–16].

We propose to assess the danger in a situation by calculating a set of n_c objective physical measures, denoted $\{\alpha_1, \ldots, \alpha_{n_c}\}$, describing how difficult it is for a CA system to avoid a collision with other objects. This could, e.g., be time-to-collision (TTC) [6] or the needed use of available tire-to-road friction in order to avoid an accident [12]. For the derivation of the framework we assume that there exists n_{c} such measures, denoted α_{1-n_c}, which are described by the functions

$$\alpha_i \triangleq g_i^{\mathrm{c}}(\mathbf{x}_k^{\mathrm{h}}, \mathbf{X}_{k:k+N}^{\mathrm{t}}) \qquad i = 1 \ldots n_{\mathrm{c}} \tag{7}$$

where $\mathbf{X}^{\mathrm{t}}_{k:k+N} = \{\mathbf{x}^{\mathrm{t}}_k, \mathbf{x}^{\mathrm{t}}_{k+1}, \ldots, \mathbf{x}^{\mathrm{t}}_{k+N}\}$ is the future trajectory of all other objects between time k and $k+N$. Given an initial state $\mathbf{x}^{\mathrm{t}}_k$, we assume that $\mathbf{X}^{\mathrm{t}}_{k+1:k+N}$ can be described using a known statistical model

$$\mathbf{X}^{\mathrm{t}}_{k+1:k+N} = f_{\mathrm{c}}(\mathbf{x}^{\mathrm{t}}_k, \mathbf{w}^{\mathrm{c}}_k) \tag{8}$$

where $\mathbf{w}^{\mathrm{c}}_k$ is a noise process describing the uncertainty in the motion of the objects.

Using the expressions in (7), a threat is detected if

$$\alpha_i > \alpha_i^{\mathrm{lim}} \qquad \forall\, i = 1 \ldots n_{\mathrm{c}} \tag{9}$$

lim where $\{\alpha_i^{\mathrm{lim}}\}_{i=1}^{n_{\mathrm{c}}}$ are feature specific design parameters typically specifying some critical limit of the measure. The clear advantage of this approach is that the design parameters can be related to physical properties, such as desired speed reduction for a collision mitigation system when evaluating TTC. Additionally, different features can set different thresholds depending on timing and intrusiveness of the intervention, e.g., a warning feature can have a lower threshold than a autonomous braking feature.

Using (9), the criticality hypotheses introduced in (6) can be defined as

$$\begin{cases} \mathcal{H}^{\mathrm{c}}_0 \ : \bigcup_{i=1}^{n_{\mathrm{c}}} \alpha_i \leq \alpha_i^{\mathrm{lim}} \\ \mathcal{H}^{\mathrm{c}}_1 \ : \bigcap_{i=1}^{n_{\mathrm{c}}} \alpha_i > \alpha_i^{\mathrm{lim}} \end{cases} \tag{10}$$

where $\mathcal{H}^{\mathrm{c}}_1$ is true if all physical measures are above there respective thresholds. To get a robust and conservative decision rule we propose to make a separate decision on each measure and require all to be true for there to be a critical situation. As such, the fact that one measure assess the situation as very critical has limited effect on the total assessment of the situation. For example, if the CA system needs to use all available friction to avoid a collision by steering to the left, the situation is still not assessed as critical if the system can easily avoid an accident by autonomously braking and/or steering to the right.

4.2 Driver acceptance modeling

In order to evaluate driver acceptance of an intervention we propose to model the probabilities for $\mathcal{H}^{\mathrm{d}}_0$ and $\mathcal{H}^{\mathrm{d}}_1$ as defined in (5) and use the decision rule (44) to determine whether an intervention is accepted or not. We argue that:

> The driver has a high acceptance for interventions if *the driver perceives* that the traffic situation indeed was critical when an intervention was initiated.

Similarly as for the general threat assessment discussed in Section 4.1, we assume that the driver's sense of criticality can be assessed by evaluating a set of physical measures.

The driver uses her eyes and ears to make observations regarding the current traffic situation. Based on these observations, the driver is assumed to draw conclusions regarding the current state of the host vehicle, denoted $\mathbf{z}^{\mathrm{h}}_k$, and all other vehicles, denoted $\mathbf{z}^{\mathrm{t}}_k$. Additionally, the driver is assumed to make predictions of the future states of the observed objects. Lets denote N samples of this future trajectory as $\mathbf{Z}^{\mathrm{t}}_{k:k+N} = \{\mathbf{z}^{\mathrm{t}}_k, \mathbf{z}^{\mathrm{t}}_{k+1}, \ldots, \mathbf{z}^{\mathrm{t}}_{k+N}\}$.

Using these (stochastic) variables, we propose to model the driver's "threat assessment" of the traffic situation using n_{d} physical measures, denoted β_{1-n_d}, which are described as

$$\beta_i \triangleq g_i^{\mathrm{d}}(\mathbf{z}_k^{\mathrm{h}}, \mathbf{Z}_{k:k+N}^{\mathrm{t}}) \qquad i = 1 \dots n_{\mathrm{d}}. \tag{11}$$

The driver is assumed to deem a situation as critical if

$$\beta_i > \beta_i^{\mathrm{lim}} \qquad \forall\, i = 1 \dots n_{\mathrm{d}}. \tag{12}$$

In this case, the set $\{\beta_i^{\mathrm{lim}}\}_{i=1}^{n_{\mathrm{d}}}$ are driver specific parameters which may be tuned with respect to a specific intervention type, e.g., a warning or an emergency brake intervention. These parameters are typically connected to the driver's safety margins, e.g., an aggressive driver may be modeled using one set of values, whereas a cautions driver is better modeled using a different set. Using (12), the driver acceptance hypotheses defined in (5) can be expressed mathematically as

$$\begin{cases} \mathcal{H}_0^{\mathrm{d}} : \bigcup_{i=1}^{n_{\mathrm{d}}} \beta_i \leq \beta_i^{\mathrm{lim}} \\ \mathcal{H}_1^{\mathrm{d}} : \bigcap_{i=1}^{n_{\mathrm{d}}} \beta_i > \beta_i^{\mathrm{lim}}. \end{cases} \tag{13}$$

The driver's perception of the current and future traffic situation is typically not known to the CA system. Instead, to be able to evaluate (11), we propose to use probabilistic driver models to capture the driver's uncertainty in $\mathbf{z}_k^{\mathrm{h}}$ and $\mathbf{Z}_{k:k+N}^{\mathrm{t}}$ based on the driver's observations. We assume these models can be written on the form

$$\mathbf{z}_k^{\mathrm{h}} = h^{\mathrm{h}}(\mathbf{X}_{1:k}^{\mathrm{h}}, \mathbf{v}_k^{\mathrm{h}}) \tag{14}$$

$$\mathbf{z}_k^{\mathrm{t}} = h^{\mathrm{t}}(\mathbf{X}_{1:k}^{\mathrm{t}}, \mathbf{v}_k^{\mathrm{t}}) \tag{15}$$

$$\mathbf{Z}_{k+1:k+N}^{\mathrm{t}} = f_{\mathrm{d}}(\mathbf{z}_k^{\mathrm{t}}, \mathbf{w}_k^{\mathrm{d}}) \tag{16}$$

where (14) and (15) model how the driver perceives the host vehicle state and target states, respectively, and (16) describes how the driver predicts that the states of the other objects will evolve over time. Each model includes a noise process, denoted $\mathbf{v}_k^{\mathrm{h}}$, $\mathbf{v}_k^{\mathrm{t}}$ and $\mathbf{w}_k^{\mathrm{d}}$, capturing the driver's uncertainty in in the current traffic situation and the future trajectories of the other objects.

4.3 Decision-making

In order for the CA system to make a decision of an autonomous intervention, we propose to make two separate decisions, both of which need to be true. Firstly, is a significant intervention needed to autonomously avoid a collision, see (10), and will the driver accept the intervention, see (13). However, to be able to evaluate (10) and (13) deterministically, the system needs to fully know the state of the host vehicle as well as the current and future state of all other vehicles and even the driver's perception thereof. Instead, we propose to use a probabilistic decision-framework incorporating statistical models to describe system uncertainties.

To use the decision methodology detailed in Appendix A, we need to calculate the probabilities of hypotheses (5) and (6) using the models that we have defined and the observations that have been collected. We start by deriving the expressions for deciding whether an intervention is needed or not.

Criticality decision

Using the decision rule (44), the event that the situation is judged as critical according to the i^{th} measure can be written as the boolean expression,

$$\mathcal{D}_i^{\text{c}} \triangleq \left\{ \frac{\Pr\{\alpha_i > \alpha_i^{\lim} \mid \mathbf{Y}_{1:k}\}}{\Pr\{\alpha_i \leq \alpha_i^{\lim} \mid \mathbf{Y}_{1:k}\}} > c_i^{\text{c}} \right\}, \tag{17}$$

where c_i^{c} is a threshold for indicating how certain the CA system needs to be that there actually is a threat according to the i^{th} measure. Note that, as an autonomous intervention is only initiated if it is also assessed that the driver accepts an intervention, the thresholds in (17) could be tuned such that it requires only a small probability that $\alpha_i > \alpha_i^{\lim}$ without causing undesired interventions.

The probability expressions in (17) can be calculated as

$$\begin{aligned} &\Pr\{\alpha_i > \alpha_i^{\lim} \mid \mathbf{Y}_{1:k}\} = \\ &\qquad \mathbb{E}\left\{\mathcal{I}\{g_i^{\text{c}}(\mathbf{x}_k^{\text{h}}, \mathbf{X}_{k:k+N}^{\text{t}}) > \alpha_i^{\lim}\} | \mathbf{Y}_{1:k}\right\} \end{aligned} \tag{18}$$

where $\mathcal{I}\{\cdot\}$ is an indicator which is one if the expression inside the braces is true and zero otherwise. Note that (18) can be calculated using the motion model (8) and the posterior pdf, $p(\mathbf{x}_k^{\text{h}}, \mathbf{x}_k^{\text{t}} | \mathbf{Y}_{1:k})$, from the tracking system. The probability that the i^{th} measure is below its limit is $1 - \Pr\{\alpha_i > \alpha_i^{\lim} \mid \mathbf{Y}_{1:k}\}$, following the law of total probability.

Finally, the system decides wether the situation is critical or not by evaluating all n_{c} measures using (17) and forming the joint event,

$$\mathcal{D}_{\text{c}} \triangleq \bigcap_{i=1}^{n_{\text{c}}} \mathcal{D}_i^{\text{c}}. \tag{19}$$

The system deems that an imminent intervention is needed if (19) is true.

Driver acceptance decision

Similarly as when the CA system assess the criticality of a situation, we propose to model the driver's acceptance by evaluating the n_{d} measures defined in (11) according to (12). Given the driver models (14) – (16), the driver's decision that there is a threat according to the i^{th} measure can then be modeled as

$$\mathcal{D}_i^{\text{d}} \triangleq \left\{ \frac{\Pr\{\beta_i > \beta_i^{\lim} | \mathbf{X}_{1:k}^{\text{h}}, \mathbf{X}_{1:k}^{\text{t}}\}}{\Pr\{\beta_i \leq \beta_i^{\lim} | \mathbf{X}_{1:k}^{\text{h}}, \mathbf{X}_{1:k}^{\text{t}}\}} > c_i^{\text{d}} \right\} \tag{20}$$

where c_i^{d} is a threshold for indicating that the driver experiences the situation as critical with respect to the i^{th} measure. The probabilities in (20) can be calculated as

$$\begin{aligned} &\Pr\{\beta_i > \beta_i^{\lim} | \mathbf{X}_{1:k}^{\text{h}}, \mathbf{X}_{1:k}^{\text{t}}\} = \\ &\qquad \mathbb{E}\left[\mathcal{I}\left\{g_i^{\text{d}}(\mathbf{z}_k^{\text{h}}, \mathbf{Z}_{k:k+N}^{\text{t}}) > \beta_i^{\lim}\right\} | \mathbf{X}_{1:k}^{\text{h}}, \mathbf{X}_{1:k}^{\text{t}}\right] \end{aligned} \tag{21}$$

where the expected value is found using the driver models (14) – (16).

Similarly, as for the threat assessment of the CA system, the driver assesses the situation on a whole as critical only if the situation is critical with respect to all n_{d} measures, that is

$$\mathcal{D}_{\mathrm{d}} \triangleq \bigcap_{i=1}^{n_{\mathrm{d}}} \mathcal{D}_i^{\mathrm{d}}. \tag{22}$$

For the CA system to be able to assess the driver acceptance, i.e., evaluate (22), it needs to marginalize the states $\mathbf{X}_{1:k}^{\mathrm{h}}$ and $\mathbf{X}_{1:k}^{\mathrm{t}}$ from the expression using the observations, $\mathbf{Y}_{1:k}$. Consequently, the CA system makes the decision as

$$\tilde{\mathcal{D}}_{\mathrm{d}} \triangleq \left\{ \frac{\Pr\{\, \mathcal{D}_{\mathrm{d}} \,|\mathbf{Y}_{1:k}\}}{\Pr\{\neg \mathcal{D}_{\mathrm{d}} | \mathbf{Y}_{1:k}\}} > \tilde{c}^{\mathrm{d}} \right\} \tag{23}$$

where

$$\Pr\{\, \mathcal{D}_{\mathrm{d}} \,|\mathbf{Y}_{1:k}\} = \mathbb{E}\left[\mathcal{I}\left\{\mathcal{D}_{\mathrm{d}}(\mathbf{X}_{1:k}^{\mathrm{h}}, \mathbf{X}_{1:k}^{\mathrm{t}})\right\} |\mathbf{Y}_{1:k}\right] \tag{24}$$

and $\tilde{c}^{\mathrm{d}}$ is a threshold specifying how certain the CA system needs to be that the driver judges the situation as critical given its uncertainty in $\mathbf{X}_{1:k}^{\mathrm{h}}$ and $\mathbf{X}_{1:k}^{\mathrm{t}}$.

Autonomous intervention decision

In order for the system to make an autonomous intervention, it must judge that the driver accepts the intervention and that the situation is critical. As such, the decision to intervene or not is given by

$$\mathcal{D} \triangleq \tilde{\mathcal{D}}_{\mathrm{d}} \bigcap \mathcal{D}_{\mathrm{c}}. \tag{25}$$

In sum, this concludes the presentation of our proposed probabilistic decision-making framework treating both state and prediction uncertainty when considering both the driver's acceptance of an intervention, $\tilde{\mathcal{D}}_{\mathrm{d}}$, and the criticality of the situation from the CA system's perspective $\mathcal{D}_{\mathrm{c}}$.

5 Modeling choices

In order to evaluate the probabilistic decision framework, we need to make assumptions and specific choices regarding the models introduced in the previous section. In this section, we present one such possible set of choices that we use to evaluate the framework in this paper. These choices have been made to illustrate that the framework has appealing properties even under simple modeling assumptions, which leaves room for further improvement on threat assessment and driver acceptance modeling. The selected state representation is given in Section 5.1, whereas Section 5.2 presents one possible choice

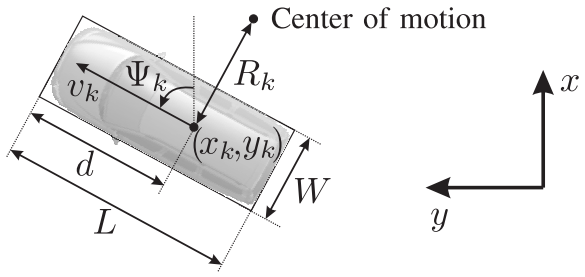


Fig. 4: State representation in a Cartesian coordinate system. The radius of turn R_k is defined as positive when the road user is turning to the left.

of threat assessment measures α_i [see (7)], an object prediction model f_{c} [see (8)] and criticality limits α_i^{lim} [see (9)]. Additionally, Section 5.3 presents a model of the driver's perception h^{h}, h^{t} [see (14) – (15)], her object trajectory predictions f_{d} [see (16)], threat measures β_i [see (11)] and criticality limits β_i^{lim} [see (12)].

5.1 State representation

We choose to have the same state representation for all objects in $\mathbf{x}_k^{\text{t}}$ as well as for the host vehicle, $\mathbf{x}_k^{\text{h}}$, here represented by a generic state vector

$$\mathbf{x}_k = [x_k \; y_k \; \Psi_k \; v_k \; c_k \; \dot{v}_k]^{\text{T}}. \tag{26}$$

In (26), (x_k, y_k) is the road user's position in the ground-fixed Cartesian coordinate system, Ψ_k is the heading angle, v_k is the object's speed over ground, $c_k = 1/R_k$ is the current curvature of the trajectory, where R_k is the radius of turn, and $\dot{v}_k$ is the longitudinal acceleration. The instantaneous center of motion is perpendicular to the heading angle at a fixed distance d from the front end of an object of length L and width W, as illustrated in Fig. 4. These three measures are assumed to be deterministic and known.

It is assumed that the system's uncertainty in host vehicle state, $\mathbf{x}_k^{\text{h}}$ is negligible, i.e., the state is assumed to be fully known and independent of the states of the other objects. Furthermore, it is assumed that the information regarding the other objects coming from the sensor fusion system is described using a Gaussian density, $\mathbf{x}_k^{\text{t}} \sim \mathcal{N}(\bar{\mathbf{x}}_k^{\text{t}}, \mathbf{P}_k^{\text{t}})$.

5.2 Threat assessment

In the framework presented in Section 4, the threat assessment algorithm g_i^{c} is required to be able to estimate the threat measures α_i for any host vehicle state $\mathbf{x}_k^{\text{h}}$ and any obstacle trajectory $\mathbf{X}_{k:k+N}^{\text{t}}$ that can be given by the path prediction f_{c}, see (7) – (8). One such algorithm is proposed in [12], where four threat measures are used to describe the criticality of a traffic situation. These measures are denoted

$$\alpha_{1-4} = \{\alpha_{\text{brake}}, \alpha_{\text{accel}}, \alpha_{\text{left}}, \alpha_{\text{right}}\} \tag{27}$$

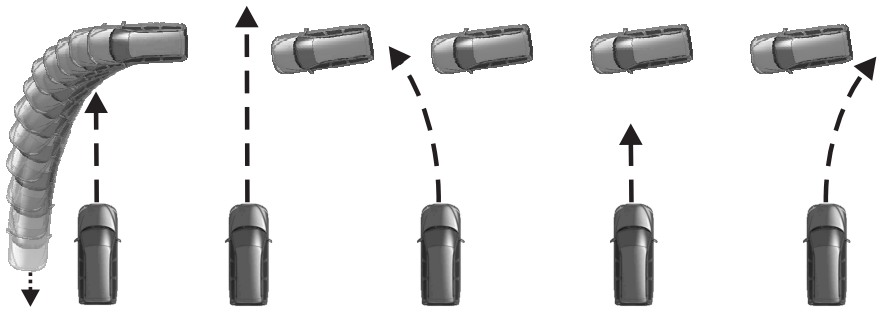


Fig. 5: The algorithm estimates how a collision with a single object can be avoided during a limited prediction horizon $k:k+N$. Acceleration, steering left, braking and steering right are considered as potential evasive manoeuvres.

which represent the fraction of the available tire-to-road friction that the CA system needs to utilize to avoid a collision with a single object by either braking, accelerating or passing the object by turning to the left or to the right, respectively. These options are illustrated in Fig. 5. The criticality limits, see (9), are denoted

$$\alpha_{1-4}^{\text{lim}} = \{\alpha_{\text{brake}}^{\text{lim}}, \alpha_{\text{accel}}^{\text{lim}}, \alpha_{\text{left}}^{\text{lim}}, \alpha_{\text{right}}^{\text{lim}}\}. \tag{28}$$

Since even a short summary of the algorithm would cover several pages, the interested reader is referred to [12] for a description on how to estimate the measures α_{1-4}.

If there are multiple objects present, the threat of each object is evaluated separately. Only the object with the highest threat level is kept for further analysis and decision-making for interventions. Clearly, the threat of the situation as a whole may be higher than the highest threat of a single object, but this problem is not treated in [12]. However, it shall be noted that the proposed decision-making framework is applicable in multi-object scenarios if a multiple object threat assessment algorithm is available. In the continuation of this paper, as the threat assessment is performed independently for each object, we assume that there is only one object in $\mathbf{x}_k^{\text{t}}$.

The future trajectory of the other objects may be given by any suitable prediction model f_{c}, as such, we propose to use a so-called bicycle model [17], i.e.,

$$\mathbf{x}_{k+i}^{\text{t}} = \mathbf{x}_{k+i-1}^{\text{t}} + \begin{bmatrix} \cos(\Psi_{k+i-1}^{\text{t}})\, v_{k+i-1}^{\text{t}} \\ \sin(\Psi_{k+i-1}^{\text{t}})\, v_{k+i-1}^{\text{t}} \\ v_{k+i-1}^{\text{t}}\, c_{k+i-1}^{\text{t}} \\ \dot{v}_{k+i-1}^{\text{t}} \\ \dot{c}_k^{\text{t}}\, \mathcal{I}\{i \leq M_{\dot{c}}^{\text{c}}\} \\ \ddot{v}_k^{\text{t}}\, \mathcal{I}\{i \leq M_{\ddot{v}}^{\text{c}}\} \end{bmatrix} \delta t \tag{29}$$

with the prediction horizon $i = 1 \ldots N$. The time interval between any two samples is given by δt and the curvature rate $\dot{c}_k^{\text{t}}$ and the longitudinal jerk $\ddot{v}_k^{\text{t}}$ is modeled as a noise process given by

$$\begin{bmatrix} \dot{c}_k^{\text{t}} \\ \ddot{v}_k^{\text{t}} \end{bmatrix} \sim \mathcal{N}(0, \mathbf{Q}_{\text{c}}). \tag{30}$$

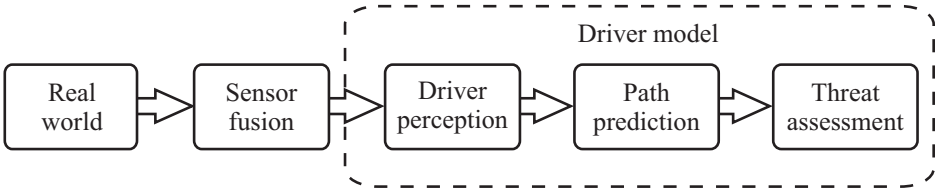


Fig. 6: Contrary to the driver, observing the world with her eyes, the driver model observes the world through a sensor fusion system.

For simplicity, the time intervals $M_{\dot{c}}^{\mathrm{c}}$ and $M_{\ddot{v}}^{\mathrm{c}}$, during which the curvature rate and the longitudinal jerk are applied, respectively, are assumed to be known. The predicted motion given by (29) corresponds to the common maneuver of applying a constant steering wheel angle rate followed by a constant steering wheel angle [18]. The longitudinal motion is given by a constant jerk followed by a constant acceleration, where decelerating objects are predicted to eventually come to a complete stop rather than changing direction.

5.3 The driver as a collision avoidance system

We argue that most drivers are aware that they have a limited ability to perceive and predict the motion of road users, and that this is considered when planning maneuvers. Consequently, if we assume that a driver desires to avoid collisions, the driver (consciously or subconsciously) needs to

1. observe both stationary objects and other road users,

2. make predictions of road user trajectories, and

3. plan safe maneuvers considering these predictions.

These three objectives motivate that the driver can be modeled as a collision avoidance system, consisting of

1. a tracking system (i.e., the driver's eyes or perception), given by h^{h} and h^{t}, see (14) – (15),

2. a prediction model f_{d}, see (16), and

3. a threat assessment algorithm g_i^{d}, see (11).

The driver observes the real world directly in order to judge whether a traffic situation is critical or not. Similarly, the driver model observes the world through a sensor fusion system, as illustrated in Fig. 6.

The driver's perception

Many studies claim that looming, i.e., when an observed object becomes increasingly larger on the perceiver's retina, is one of the most important cues that drivers observe when making decisions on when to brake [19, 20]. Drivers have great difficulties to perceive small looming changes [21, 22] and may hence find it hard to estimate the acceleration of an object.

Modeling of the human perception is not an easy task, but promising attempts have been made, e.g., by using machine vision to estimate the probability that a pedestrian has been detected by the driver [23]. Many more studies are needed to obtain a more complete human perception model. To limit the scope of our analysis in Section 7, we assume that drivers only have difficulties in perceiving the longitudinal acceleration of other road users, whereas position, size and velocity estimates are accurate. This basic assumption is partly supported by [24], which shows that human perception of an object's velocity is much more accurate than the perception of its acceleration. To conclude, the driver's perception, see (14)– (15), is modeled as

$$\mathbf{z}_k^{\mathrm{h}} = h^{\mathrm{h}}(\mathbf{X}_{1:k}^{\mathrm{h}}, \mathbf{v}_k^{\mathrm{h}}) = \mathbf{x}_k^{\mathrm{h}} \tag{31}$$

$$\mathbf{z}_k^{\mathrm{t}} = h^{\mathrm{t}}(\mathbf{X}_{1:k}^{\mathrm{t}}, \mathbf{v}_k^{\mathrm{t}}) = \mathbf{x}_k^{\mathrm{t}} + \mathbf{v}_k^{\mathrm{t}} \tag{32}$$

$$\mathbf{v}_k^{\mathrm{t}} = [0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \; v_{\mathrm{a}}]^{\mathrm{T}}, \; v_{\mathrm{a}} \sim \mathcal{N}(0, \sigma_{\mathrm{a}}^2). \tag{33}$$

The driver's path prediction

The driver's prediction f_{d} of an object's future trajectory $\mathbf{Z}_{k:k+N}^{\mathrm{t}}$, see (16), is given by the same prediction model the system use, see (29), with $\mathbf{z}_k^{\mathrm{t}}$ as input instead of $\mathbf{x}_k^{\mathrm{t}}$. For simplicity, the prediction uncertainty is assumed to originate solely for the driver's uncertainty in the object's acceleration. Hence, the process noise is $\mathbf{w}_k^{\mathrm{d}} = \mathbf{0}$ and the time intervals in (29) are set to $M_{\dot{c}}^{\mathrm{d}} = M_{\ddot{v}}^{\mathrm{d}} = 0$.

The driver's threat assessment

Since the driver is modeled as a CA system, the driver's threat assessment is given by $g_i^{\mathrm{d}} = g_i^{\mathrm{c}}$, i.e., the same algorithm that was used for estimating the threat measures α_i can now be used for estimating the driver's threat measures

$$\beta_{1-4} = \{\beta_{\text{brake}}, \beta_{\text{accel}}, \beta_{\text{left}}, \beta_{\text{right}}\}. \tag{34}$$

The only difference is that the input to the driver's threat assessment g_i^{d} is given by the driver's observations $\mathbf{z}_k^{\mathrm{h}}$ and predictions $\mathbf{Z}_{k:k+N}^{\mathrm{t}}$. The driver's criticality limits (12) are denoted

$$\beta_{1-4}^{\text{lim}} = \{\beta_{\text{brake}}^{\text{lim}}, \beta_{\text{accel}}^{\text{lim}}, \beta_{\text{left}}^{\text{lim}}, \beta_{\text{right}}^{\text{lim}}\}. \tag{35}$$

Driver acceptability for interventions

To summarize,

> An intervention is justified if the driver, *modeled as a* CA *system*, judges that the traffic situation is critical with respect to all threat measures β_{1-4} at the time k when an intervention is triggered, see (20).

In case the driver is distracted, we assume that once an intervention is triggered, the driver will shortly observe the threat and thus be able to judge whether the intervention was motivated or not.

6 Implementation

In this section, the expressions needed to implement the decision-making framework presented in Section 4, using the modeling choices and assumptions introduced in Section 5, are derived.

6.1 Probability approximations

The probabilistic decision-making framework presented in Section 4 basically consists of evaluating probability ratios. These calculations involve solving integrals (calculating the expectations in (18), (21) and (24)) which rarely have analytical solutions, which is also the case for the nonlinear models used in this paper. Instead we resort to approximative solutions.

To arrive at a computationally tractable solution, we propose to use a grid of deterministically chosen sample (grid) points with associated weights to approximate the integral as a weighted sum of evaluation. That is, the probability of an event $\mathcal{E}$ which is dependent on a stochastic variable $\mathbf{x}$ is approximated as

$$\Pr\{\mathcal{E}\} = \mathbb{E}[\mathcal{I}\{\mathcal{E}(\mathbf{x})\}] \approx \sum_{i=1}^{L} w_i \mathcal{I}\{\mathcal{E}(\mathbf{x}^{(i)})\} \tag{36}$$

where $\{\mathbf{x}^{(i)}\}_{i=1}^{L}$ are the deterministically chosen sample points and $w_i \propto p(\mathbf{x}^{(i)})$. As — according to the modeling choices and assumptions we made in Section 5 — there is only uncertainty regarding $\mathbf{x}_k^{\mathrm{t}}$, $\mathbf{w}_k^{\mathrm{c}}$ and $\mathbf{v}_k^{\mathrm{t}}$ we only need to choose sample points to evaluate in these dimensions. We evaluate the criticality and driver acceptance decision separately.

6.2 Criticality decision

The criticality decision is governed by event probabilities on the form of (18). To approximate these using the grid method we construct an augmented state $\tilde{\mathbf{x}}_k^{\mathrm{t}} = [(\mathbf{x}_k^{\mathrm{t}})^T (\mathbf{w}_k^{\mathrm{c}})^T]^T$ which is distributed as

$$\tilde{\mathbf{x}}_k^{\mathrm{t}} \sim \mathcal{N}\left(\begin{bmatrix} \bar{\mathbf{x}}_k^{\mathrm{t}} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{P}_k^{\mathrm{t}} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_{\mathrm{c}} \end{bmatrix} \right). \tag{37}$$

From (37), $L = 49$ grid points are selected as the union of three sigma point sets generated using the Unscented transform with $\kappa = \{0\ ,\ 1-m,\ 1-\frac{m}{3}\}$, respectively, where m is

the length of $\tilde{\mathbf{x}}_k^{\mathrm{t}}$. The interpretation of κ is explained e.g. in [25]. Lets denote this grid point set as

$$\{w_j^{\mathrm{x}}, \tilde{\mathbf{x}}_k^{t,(j)}\}_{j=1}^{L} \tag{38}$$

where $w_j^{\mathrm{x}} \propto p(\tilde{\mathbf{x}}_k^{\mathrm{t},(j)})$ as defined by (37). Each grid point in (38) is then propagated through the motion model (29) to form the transformed grid point set $\{w_j^{\mathrm{x}}, \mathbf{X}_{k:k+N}^{t,(j)}\}_{j=1}^{L}$. The sought probabilities can now be approximated as

$$\begin{aligned}\Pr\{\mathcal{E}|\mathbf{Y}_{1:k}\} &= \mathbb{E}[\mathcal{I}\{\mathcal{E}(\mathbf{x}_t^h, \mathbf{X}_{k:k+N}^{\mathrm{t}})\}|\mathbf{Y}_{1:k}] \\ &\approx \sum_{j=1}^{L} w_i^{\mathrm{x}} \mathcal{I}\{\mathcal{E}(\mathbf{x}_t^h, \mathbf{X}_{k:k+N}^{\mathrm{t},(j)})\}. \end{aligned} \tag{39}$$

for the events $\mathcal{E} = \{g_i^{\mathrm{c}}(\mathbf{x}_k^{\mathrm{h}}, \mathbf{X}_{k:k+N}^{\mathrm{t}}) > \alpha_i^{\mathrm{lim}}\}$ where $i \in [1, 4]$.

6.3 Driver acceptance decision

The evaluation of the driver acceptance is performed in two steps as described by (20) and (23). We start by evaluating the driver's sense of criticality as described by (20) by finding solutions to probabilities on the form of (21).

For a given $\mathbf{x}_k^{\mathrm{h}}$ and $\mathbf{x}_k^{\mathrm{t}}$, these probabilities are easily approximated using the grid method. The proposed driver model, defined in (31) - (32), stipulates that the driver only has added uncertainty regarding the longitudinal acceleration of the other object. To account for this added uncertainty, $M = 3$ grid points in $v_a \sim \mathcal{N}(0, \sigma_a^2)$ are chosen in the same manner as the Unscented Transform with $\kappa = \frac{2}{3}$. Lets donate this grid point set,

$$\{w_l^{\mathrm{z}}, v_a^{(l)}\}_{l=1}^{M}, \tag{40}$$

where $w_l^{\mathrm{z}} \propto p(v_a^{(l)})$. From (40), grid points representing the uncertainty in the driver's perception, $\mathbf{z}_k^{\mathrm{t}}$, are obtained by appending these grid points to $\mathbf{x}_k^{\mathrm{t}}$ according to (32) as

$$\{w_l^{\mathrm{z}}, h^{\mathrm{t}}(\mathbf{x}_k^{\mathrm{t}}, \mathbf{v}_k^{\mathrm{t},(l)})\}_{l=1}^{M}, \tag{41}$$

Propagating the points in (41) through the prediction model in (29) yields a transformed grid point set $\{w_l^{\mathrm{z}}, \mathbf{Z}_{k:k+N}^{\mathrm{t},(l)}\}_{l=1}^{M}$, from which the probabilities on the form of (21) can be approximated in the same manner as (36) and the decision in (20) and (22) can be evaluated.

For the CA system to assess the driver's acceptance of an intervention, the system's uncertainty in $\mathbf{x}_k^{\mathrm{t}}$ needs to be accounted for. Using the method explained above, the decision in (22) can be evaluated for a given $\mathbf{x}_k^{\mathrm{t}}$. Again, choosing a set of $L = 37$ grid points in $\mathbf{x}_k^{\mathrm{t}} \sim \mathcal{N}(\bar{\mathbf{x}}_k^{\mathrm{t}}, \mathbf{P}_k^{\mathrm{t}})$ using the same procedure used to generate (38)[1]. The probabilities on the form of (24) can now be approximated by evaluating the driver acceptance as described above by inserting each grid point in $\mathbf{x}_k^{\mathrm{t}}$ into (41).

[1] In practice, these grid points are chosen as a sub-set of the grid points in (38) where the $\mathbf{w}_k^{\mathrm{c}}$ dimension is excluded

7 Results

In this section, the decision-making framework is applied to the collision scenarios illustrated in Fig. 7, using the grid based implementation method in Section 6 and the parameter setting in Table 1. In order to find a more suitable selection of the driver's threat thresholds β_i^{lim}, it is suggested that extensive testing both in field operational tests and on test tracks should be conducted, but such studies are out of the scope for the present paper. In addition, when estimating how difficult it is for the driver to steer or brake to avoid a collision, it is assumed that the lateral jerk and the longitudinal jerk does not exceed ± 10 m/s^3 and ± 15 m/s^3, respectively.

In the examples, it is assumed that the CA system is equipped with actuators only for automatic braking, whereas steering or acceleration cannot be performed automatically. This is indicated by the parameter values of the threat limits α_i^{lim}, e.g., $\alpha_{\text{accel}}^{\text{lim}} = 0$ in Table 1. The light grey area in Figs. 8–10 indicate when it is too late for the system to prevent a collision by braking. In the graphs, it is assumed the brake system has a capacity of -10 m/s^2, a maximum deceleration rate of -20 m/s^3 and an initial time delay of $t_{\text{d}} = 0.05$ s.

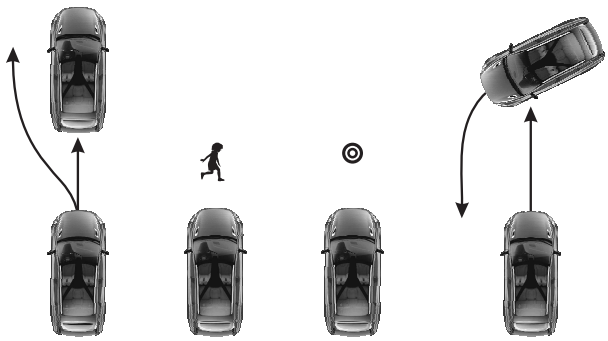


Fig. 7: Collision scenarios with a lead vehicle, a playing child, a trash can and a turning vehicle. The arrows in the rear end collision scenario illustrate scenarios with and without an evasive maneuver by the host vehicle (bottom).

7.1 Collision scenarios on a test track

Results for an authentic rear-end collision scenario are presented in Fig. 8. For safety reasons, the lead vehicle is represented by a soft inflatable car (3x1.7 m) which is attached to a trolley driven by a wire system at 50 km/h. The state of the soft car is estimated using a differential GPS system, as described in [26]. Measurement noise is added to obtain measurements representative of radar and camera-based sensor fusion system, see $\mathbf{P}_k^{\text{t}}$ in Table 1. In a first test, the driver of the host vehicle approaches the lead vehicle at 80 km/h and performs a late evasive steering maneuver. Immediately prior to this maneuver it can be seen that the probability that the driver considers the traffic situation as critical

Table 1: Parameter setting

$\alpha_{\text{brake}}^{\text{lim}}$	5 m/s^2	$\alpha_{\text{accel}}^{\text{lim}}$	0 m/s^2	$\alpha_{\text{left}}^{\text{lim}}$	0 m/s^2	$\alpha_{\text{right}}^{\text{lim}}$	0 m/s^2
$\beta_{\text{brake}}^{\text{lim}}$	7 m/s^2	$\beta_{\text{accel}}^{\text{lim}}$	7 m/s^2	$\beta_{\text{left}}^{\text{lim}}$	7 m/s^2	$\beta_{\text{right}}^{\text{lim}}$	7 m/s^2
c_i^{c}	9	c_i^{d}	0.25	$\tilde{c}^{\text{d}}$	9	σ_{a}^2	4 m^2/s^4
δt	0.05 s	N	40	$M_{\dot{c}}^{\text{c}}$	20	$M_{\ddot{v}}^{\text{c}}$	20
Host vehicle		L_{h}	5 m	W_{h}	2 m	d_{h}	3.9 m
Lead vehicle		L_{t}	3 m	W_{t}	1.7 m	d_{t}	2 m
Playing child		L_{t}	0.6 m	W_{t}	0.6 m	d_{t}	0.3 m
Trash can		L_{t}	0.6 m	W_{t}	0.6 m	d_{t}	0.3 m
Turning vehicle		L_{t}	5 m	W_{t}	2 m	d_{t}	2.5 m
$\mathbf{P}_k^{\text{t}} = \text{diag}\left(0.1^2\ 0.1^2\ \left(1\frac{\pi}{180}\right)^2\ 0.1^2\ (0.01)^2\ 1^2\right)$							
$\mathbf{Q}_{\text{c}} = \text{diag}\left(0.005^2\ 0.5^2\right)$							

increases. However, the probability decreases as soon as the driver initiates an evasive steering maneuver.

In a second test, the host vehicle drives straight into the lead vehicle at 80 km/h without braking or steering. The results in Fig. 8 show that the CA system can prevent an accident without disturbing the driver with unnecessary braking, given the parameter setting in Table 1.

7.2 Simulated collision scenarios

Figure 9 depicts simulated scenarios where the driver is approaching either a playing child, or a trash can, at 50 km/h. Both objects are initially positioned directly in the host vehicle's path and the child is standing still. The uncertainty in the state estimate is given by the covariance matrix $\mathbf{P}_k^{\text{t}}$ in Table 1. The driver is assumed to have accurate estimates of the current position of both objects, and knows that the trash can will remain stationary. In this example, it is assumed that the child only can run back and forth across the road, where the driver's prediction of child's future speed is bounded to the interval ± 4 m/s. The results clearly show that the estimated driver acceptance for interventions is higher when approaching the child, as compared to when approaching the trash can.

7.3 Collision scenario using real radar data

In Fig. 10, the evaluation is performed for an intersection collision scenario using a radar-based sensor fusion system. In order to obtain realistic measurements, a real target vehicle was used. The state estimates $\mathbf{x}_k^{\text{t}}, \mathbf{P}_k^{\text{t}}$ in the scenario displayed in Fig. 10 are provided by a tracker [27], using a further developed version of the radar sensor model presented in [3]. The state of the host vehicle was then simulated to drive at a higher speed (50 km/h) than the actual speed, such that a collision situation was created without endangering the drivers. The results indicate that it is realistic that CA systems can be designed to

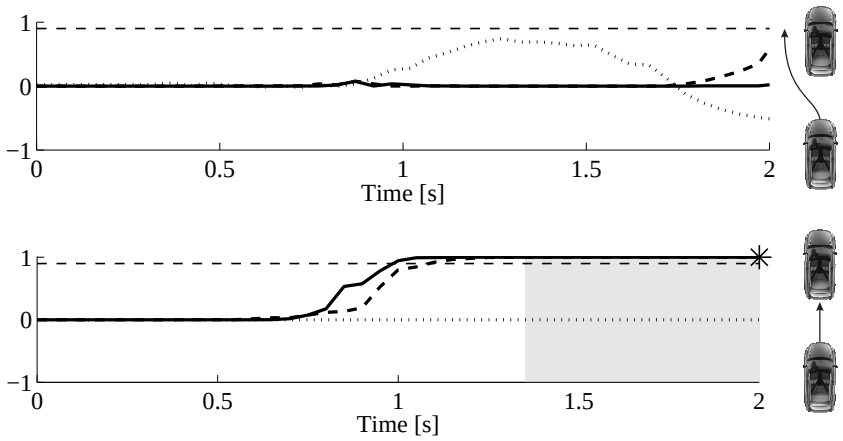


Fig. 8: Rear-end collision scenario with (top) and without (bottom) an evasive steering manoeuvre. The graphs show the probability that the driver will accept a brake intervention, $\Pr\{\tilde{\mathcal{D}}_{\mathrm{d}} \,|\mathbf{Y}_{1:k}\}$ (solid), and the probability that the situation is critical with respect to the CA system, $\Pr\{\mathcal{D}_{\mathrm{c}} \,|\mathbf{Y}_{1:k}\}$ (dashed). The dotted line shows the lateral acceleration (in [g]) of the host vehicle. The star marks the time (t = 2 s) when the vehicles collide if no evasive action is taken. At $t \approx 1.8$ s (top), $\Pr\{\mathcal{D}_{\mathrm{c}} \,|\mathbf{Y}_{1:k}\}$ rises because the host vehicle passes close to the lead vehicle while turning to the right.

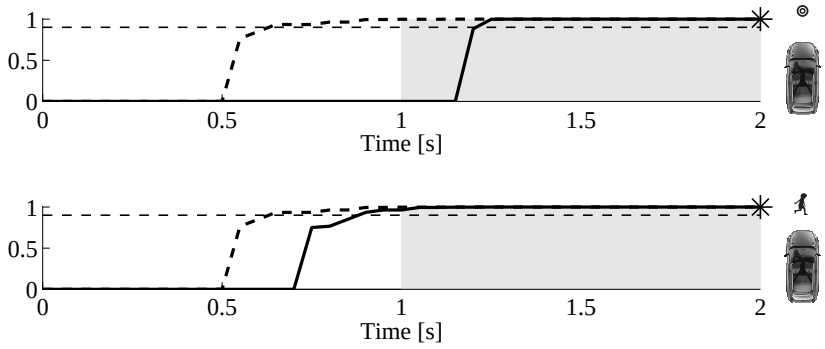


Fig. 9: The probability that the driver will accept a brake intervention, $\Pr\{\tilde{\mathcal{D}}_{\mathrm{d}} \,|\mathbf{Y}_{1:k}\}$ (solid), when approaching a trash can (top) or a playing child (bottom) at 50 km/h. The star marks the time ($t = 2$ s) when a collision occurs if no evasive action is taken. The dashed line shows the probability that immediate braking is needed, $\Pr\{\mathcal{D}_{\mathrm{c}} \,|\mathbf{Y}_{1:k}\}$. When approaching the child both $\Pr\{\tilde{\mathcal{D}}_{\mathrm{d}} \,|\mathbf{Y}_{1:k}\}$ and $\Pr\{\mathcal{D}_{\mathrm{c}} \,|\mathbf{Y}_{1:k}\}$ rise above the decision thresholds c_i^{c} and $\tilde{c}^{\mathrm{d}}$ (dashed, thin) while a collision is avoidable and thus, the CA system can use automatic braking to prevent a collision without disturbing the driver.

autonomously avoid, or at least mitigate, accidents in this common type of accident scenario without disturbing the driver with unnecessary braking. A differential GPS system is used as reference to estimate when a brake intervention no longer can be used to avoid a collision (light grey area).

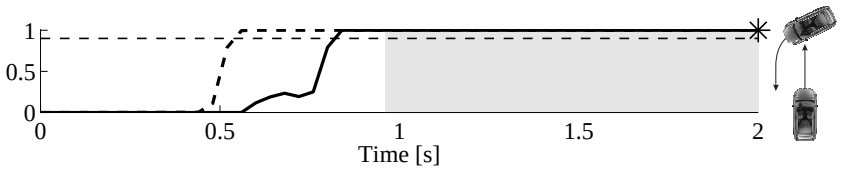


Fig. 10: The probability that the driver accepts an intervention, Pr$\{\tilde{\mathcal{D}}_{\mathrm{d}}\,|\mathbf{Y}_{1:k}\}$ (solid), and the probability that braking is needed, Pr$\{\mathcal{D}_{\mathrm{c}}\,|\mathbf{Y}_{1:k}\}$ (dashed), using data collected by a radar based sensor fusion system. The host vehicle (bottom) is approaching a turning vehicle (top). The star marks the time (t = 2 s) when a collision occurs if no evasive action is taken.

8 Conclusions

In this paper, we presented a probabilistic framework for decision-making in CA systems. We introduced a driver acceptance model for system interventions and showed that the use of this model has several appealing properties. Specifically, the driver model enables the system to perform earlier interventions in situations where the future trajectories of other road users are difficult for the driver to predict. Moreover, the proposed framework formally handles both measurement and prediction uncertainties.

Appendix A: Bayesian decision-making

In this section we summarize the basic bayesian decision-making theory used in this paper. More details can be found in good textbooks on the subject, e.g., [28].

Let there be two hypotheses; $\mathcal{H}_0$ and $\mathcal{H}_1$. Given some observations $\mathbf{y}$, we wish to make a decision regarding which hypothesis is true. Let α_i be the decision that hypothesis $\mathcal{H}_i$ is the correct one and suppose we can express a loss $\lambda_{i\,j} = L(\alpha_i|\mathcal{H}_j)$ for making the decision α_i given that the true state is $\mathcal{H}_j$. The risk $R(\alpha_i|\mathbf{y})$ is defined as the expected loss associated with a particular decision,

$$R(\alpha_i|\mathbf{y}) = \sum_{j=0,1} \lambda_{i\,j} Pr\{\mathcal{H}_j|\mathbf{y}\}. \tag{42}$$

The minimum risk is achieved by choosing $\mathcal{H}_1$ over $\mathcal{H}_0$ if $R(\alpha_1|\mathbf{y}) < R(\alpha_0|\mathbf{y})$, i.e. if

$$\frac{p(\mathbf{y}|\mathcal{H}_1)}{p(\mathbf{y}|\mathcal{H}_0)} > \frac{(\lambda_{1\,0} - \lambda_{0\,0})}{(\lambda_{0\,1} - \lambda_{1\,1})} \frac{Pr\{\mathcal{H}_0\}}{Pr\{\mathcal{H}_1\}}. \tag{43}$$

This is equivalent to evaluating the well-known likelihood ratio. It may be convenient to evaluate the posterior probabilities rather than the likelihood, *e.g* in a particle filter implementation. Applying Bayes formula to the left hand side of equation (43) gives the equivalent rule

$$\frac{Pr\{\mathcal{H}_1|\mathbf{y}\}}{Pr\{\mathcal{H}_0|\mathbf{y}\}} \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \frac{\lambda_{1\,0} - \lambda_{0\,0}}{\lambda_{0\,1} - \lambda_{1\,1}} = \gamma. \tag{44}$$

The above relation suggests to choose $\mathcal{H}_1$ over $\mathcal{H}_0$ if the ratio $Pr\{\mathcal{H}_1|\mathbf{y}\}/Pr\{\mathcal{H}_0|\mathbf{y}\}$ exceeds a threshold γ.

References

[1] "Annual statistical report," *European Road Safety Observatory*, 2008.

[2] M. Distner, M. Bengtsson, T. Broberg, and L. Jakobsson, "City safety—A system addressing rear-end collisions at low speeds," in *Proc. 21th Enhanced Safety Vehicles Conf.*, paper 09-0371, 2009.

[3] J. Gunnarsson, L. Svensson, L. Danielsson, and F. Bengtsson, "Tracking vehicles using radar detections," in *Proc. IEEE Intell. Veh. Symp.*, 2007, pp. 296–302.

[4] J. Hillenbrand, A. Spieker, and K. Kroschel, "A multilevel collision mitigation approach — its situation assessment, decision making, and performance tradeoffs," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 4, pp. 528–540, Dec. 2006.

[5] A. Broadhurst, S. Baker, and T. Kanade, "A prediction and planning framework for road safety analysis, obstacle avoidance and driver information," in *Proc. 11th World Congr. Intell. Transp. Syst.*, Oct. 2004.

[6] J. Jansson and F. Gustafsson, "A framework and automotive application of collision avoidance decision making," *Automatica*, vol. 44, no. 9, pp. 2347–2351, 2008.

[7] M. Goodrich and E. Boer, "Designing human-centered automation: Trade-offs in collision avoidance system design," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 1, pp. 40–54, Mar. 2000.

[8] J.-E. Källhammar, K. Smith, J. Karlsson, and E. Hollnagel, "Shouldn't cars react as drivers expect?" in *Proc. 4th International Driving Symp. on Hum. Fact. in Driver Assessment, Traning and Veh. Design*, 2007, pp. 9–15.

[9] A. Eidehall and L. Petersson, "Statistical threat assessment for general road scenes using Monte Carlo sampling," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 1, pp. 137–147, Mar. 2008.

[10] N. Kaempchen, B. Schiele, and K. Dietmayer, "Situation assessment of an autonomous emergency brake for arbitrary vehicle-to-vehicle collision scenarios," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 4, pp. 678–687, Dec. 2009.

[11] M. Althoff, O. Stursberg, and M. Buss, "Model-based probabilistic collision detection in autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 2, pp. 299–310, June 2009.

[12] M. Brännström, E. Coelingh, and J. Sjöberg, "Model-based threat assessment for avoiding arbitrary vehicle collisions," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 658–669, Sept. 2010.

[13] J. McCall and M. Trivedi, "Driver behavior and situation aware brake assistance for intelligent vehicles," *Proc. of the IEEE*, vol. 95, no. 2, pp. 374–387, Feb. 2007.

[14] T. Sattel and T. Brandt, "From robotics to automotive: Lane-keeping and collision avoidance based on elastic bands," *Veh. Syst. Dyn.*, vol. 46, no. 7, pp. 597–619, July 2008.

[15] D. Ferguson, M. Darms, C. Urmson, and S. Kolski, "Detection, prediction, and avoidance of dynamic obstacles in urban environments," in *Proc. IEEE Intell. Veh. Symp.*, 2008, pp. 1149–1154.

[16] P. Falcone, M. Ali, and J. Sjöberg, "Predictive threat assessment via reachability analysis and set invariance theory," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 3, pp. xx–yy, Sept. 2011.

[17] T. D. Gillespie, *Fundamentals of vehicle dynamics*. Warrendale, PA: SAE, 1992.

[18] H. Godthelp, "Vehicle control during curve driving," *Hum. Factors*, vol. 28, no. 2, pp. 211–221, Apr. 1986.

[19] H. Terry, S. Charlton, and J. Perrone, "The role of looming and attention capture in drivers' braking respones," *Accident Analysis & Prevention*, vol. 40, no. 4, pp. 1375–1382, 2008.

[20] P. Fancher, Z. Bareket, and R. Ervin, "Human-centered design of an acc-with-braking and forward-crash-warning system," *Veh. Syst. Dyn., 36*, vol. 2, no. 3, pp. 203–223, 2001.

[21] J. Barton, T. Cohn, and M. Tomizuka, "Towards a complete human driver model: The effect of vision on driving performance," in *Proc. Amer. Contr. Conf.*, 2006, pp. 2591–2598.

[22] D. Delorme and B. Song, "Human driver model for SmartAHS," *Tech. Rep., Calif. PATH Program, Inst. of Transp. Studies, Berkeley*, 2001.

[23] D. Engel and C. Curio, "Pedestrian detectability: Predicting human perception performance with machine vision," in *Proc. IEEE Intell. Veh. Symp.*, 2011, pp. 429–435.

[24] D. Rosenbaum, "Perception and extrapolation of velocity and acceleration," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 1, no. 4, pp. 395–403, 1975.

[25] S. Julier and J. Uhlmann, "A general method for approximating nonlinear transformations of probability distributions," *Dept. of Engineering Science, University of Oxford, Tech. Rep.*, 1996.

[26] M. Brännström, J. Sjöberg, L. Helgesson, and M. Christiansson, "A real-time implementation of an intersection collision avoidance system," in *18th World Congr. of the Int. Fed. of Automatic Control (IFAC), Italy*, 2011.

[27] F. Bengtsson and L. Danielsson, "Designing a real time sensor data fusion system with application to automotive safety," *Dept. of Signals and systems, Chalmers University of Technology, Tech. Rep.*, Gothenburg, Apr. 2008.

[28] R. Duda, P. Hart, D. Stork *et al.*, *Pattern classification*. 2nd ed., New York: Wiley, 2001.

Paper III

Extended object tracking using a radar resolution model

L. Hammarstrand, F. Sandblom, L. Svensson, and J. Sörstedt

Accepted for publication in
IEEE Transactions on Aerospace and Electronic Systems

Extended object tracking using a radar resolution model

Lars Hammarstrand, Fredrik Sandblom,
Lennart Svensson, and Joakim Sörstedt

Abstract

This paper concerns the problem of vehicle tracking when multiple radar reflection centers could be resolved on each vehicle. For this extended target tracking problem we propose a radar sensor model, capable of describing such measurements, incorporating sensor resolution. Furthermore, we introduce approximations to handle the inherently complex data association problem. The evaluation in terms of describing measured data and resulting tracking performance shows that the model effectively exploits the information in multiple vehicle detections.

1 Introduction

Advanced automotive active safety systems often use sensors, such as radar and camera, to gather observations on the traffic environment around the vehicle. Through a tracking framework, these observations are refined to estimates of, e.g., position of other vehicles, pedestrians and the road. Based on the estimates, dangerous situations can be detected and decisions of appropriate actions are taken. For example, the system may warn the driver of an impending collision or intervene by braking or steering in order to avoid the collision or mitigate its consequences. For the active safety system to be able to make effective decisions, it is of great importance that the provided estimates meet the requirements in terms of accuracy and detail. To achieve this with a cost efficient system, the tracking framework needs to have an accurate description of the statistical properties of the sensor observations [1].

Many of the active safety systems on the market today are solely or partly radar based. Except from being robust against different weather conditions, the radar offers accurate measurements of range and range rate to objects. Furthermore, the radar has a long history of use in, e.g., airborne applications, and there exists a vast amount of literature on how to design a tracking system based on radar observations, see [2, 3] and the references therein. There are, however, important differences between target tracking in airborne applications and vehicle tracking for active safety systems. In airborne radar applications the aim is to track aircrafts at distances of tens of kilometers, whereas in automotive active safety systems the distances to the objects of interest are in the order of tens of meters. At such short distances, the radar resolution is typically finer than the physical extent of objects.

Where, in airborne radar applications, the targets behave as point sources [2,3], in automotive scenarios the radar is typically capable of detecting multiple features (reflection centers) on the same object. In the radar literature this type of target is referred to as an extended or distributed object/target [4].

Receiving multiple detections from a vehicle offers a possibility to extract detailed information about the object. For example, the spread of the individual detections gives information regarding the physical extent of the object as well as its orientation [1]. However, multiple measurements per object also introduce some considerable difficulties compared to the point source case. For one, the algorithms and models developed using the point source assumption are no longer valid. Additionally, an accurate sensor model is more complex as the detections are spread over large parts of the object and not accurately described as originating from a single point. The sensor model also needs to consider the possibility that a target can generate multiple detections in contrast to at most one in the point source case. The uncertainty in the number of target detections makes the data association problem more intricate. The aim of this paper is to develop a computationally tractable sensor model that accurately describes the radar detections from this type of object (vehicles). The ultimate purpose being to improve the tracking of vehicles for automotive active safety systems.

Although the classical point source assumption does not hold for extended objects, little attention has been given to find a more suitable tracking formulation. A good overview of different contributions up to 2004 can be found in [5], covering extended object tracking and the closely related problem of tracking groups of targets. More recent suggestions include, [6, 7] where a formal Bayesian tracking framework is proposed for estimating the centroid of the extended targets (or target groups). The object extension is modelled as an ellipse and it is assumed that multiple measurements can originate from each object. The elliptical shape of each object is described using a positive definite random matrix. By including these matrices in the state vector, both the target centroid and object extension are jointly estimated from data. Although the proposed approach shows promising results which are robust against object shape, it is difficult to exploit object specific shape information using this framework, when such information is available. Gilholm et. al. [8] propose a particle filter solution where the detections from the extended object are modelled by a non-homogenous Poisson point process with a known but arbitrary spatial intensity. Using this description, it is possible to include information about the shape of the objects, but due to the limited flexibility of the Poisson distribution, it is often impossible to incorporate specific knowledge regarding expected number of target returns. The probabilistic multi-hypothesis tracker (PMHT) [9] relaxes the point source criterion by modeling the measurement to target associations as stochastic and independent, and has been applied to extended object tracking in, e.g., [10]. Although the PMHT does not directly provide covariance estimates, the method is useful if the number of detections originating from each target cannot be accurately modeled.

For the problem of tracking vehicles using radar observations, there are reasons

to believe that both the spatial distribution and the number of detections from a vehicle can be accurately modelled. For example, the study in [11] indicates that radar return from vehicles mainly originate from a number of specifically strong reflection centers (point sources), such as the headlamps and the wheel housings, see Fig. 3. Furthermore, if reflection centers are located within a resolution cell, the echoes from these reflectors are merged into a single joint detection (cluster detection). In [12–15], a model that captures the general behavior of a detection from a cluster including two sources/targets is used in conjunction with a description of the probability that the two targets are unresolved. Using this probabilistic description, the data association hypotheses and measurement model are expanded to also consider a merged detection from the two targets. A similar approach is proposed in [16], using a Gaussian approximation of the two-source cluster detection density originally derived in [17]. Although the solutions referred to here consider the influence of merged measurements, they are limited to handle only two sources, and the result is not easily expanded to the more general case of merging multiple sources.

Inspired by the findings in [11], we propose a radar sensor model describing the spatial distribution of vehicle detections as well as a probabilistic description over the number of vehicle detections. The proposed model also considers the effects of merging a general number of target reflections (limited resolution). As such, we are able to both incorporate shape information and expected number of vehicle detections, as well as describe the statistical behavior of the measurements. More specifically, the model family describes the radar reflections from a vehicle as originating from a set of reflection centers and, depending on the resolution of the sensor, reflectors likely to render a merged detection are grouped. The number of detections from each group is modelled as well as the distribution of the resulting detections. By associating measurements to reflector groups, instead of individual reflectors or reflector clusters, the number of association hypotheses is significantly reduced.

Furthermore, we derive a vehicle tracking framework based on our proposed sensor model. The framework is based on a linear minimum mean square error (LMMSE) estimator where the needed densities are estimated using the unscented transform (UT) [18]. A generalized version of the joint probabilistic data association (JPDA) technique [2, 19] is used to handle data association uncertainties. The proposed model is compared to the commonly used point source model in two aspects: model likelihood and tracking performance. The evaluation clearly indicates that the proposed model has significant benefits in both aspects.

The paper is organized as follows. In Section 2 the tracking problem is formalized and the necessary notation is introduced. Section 3 presents the radar sensor model, and in Section 4 we show how this model can be used in a tracking framework. Finally, Section 5 presents evaluation results of our proposed sensor model and the derived tracking framework.

2 Problem formulation

This article studies the problem of tracking vehicles with known physical dimensions, using multiple radars mounted on the host vehicle. The objective is twofold. First, to derive a family of detailed statistical models describing the radar returns from the vehicles. Second, to develop a vehicle tracking framework based on this model with the ultimate aim to improve the estimates of, i.e., the position and the velocity of the vehicles.

This section is partitioned as follows. The state parameters to be estimated are defined in Section 2.1 together with a model of how they evolve over time. Section 2.2 describes the necessary background properties of the radar observations, and Section 2.3 discusses in more detail the needed properties of the radar sensor model and the tracking framework for our specific problem.

2.1 State parametrization

All the parameters of interest are collected in the discrete time state vector $\mathbf{z}_k$, where k is the discrete time index corresponding to continues time instance t_k. The state vector contains both the states of the surrounding vehicles and the host vehicle. Each vehicle, l, is described by the sub-state vector

$$\mathbf{z}_k^l = \left[\zeta_{x,k}^l \; \zeta_{y,k}^l \; \psi_k^l \; v_k^l \; c_k^l \; \dot{v}_k^l\right]^T, \tag{1}$$

where $(\zeta_{x,k}^l, \zeta_{y,k}^l)$ is the position of vehicle l expressed in a global Cartesian coordinate system. As illustrated in Fig. 1, Ψ_k^l is the heading angle and v_k^l is the speed in the heading direction of vehicle l and $\dot{v}_k^l$ is its time derivative. The variable c_k^l represents the curvature of the current path of the l^{th} vehicle. The state vectors of all vehicles are stacked to form the complete state vector

$$\mathbf{z}_k = \left[\left(\mathbf{z}_k^h\right)^T \; \left(\mathbf{z}_k^1\right)^T \; \left(\mathbf{z}_k^2\right)^T \; \ldots \; \left(\mathbf{z}_k^{N_v}\right)^T\right]^T, \tag{2}$$

where $\mathbf{z}_k^h$ is the host vehicle state and N_v is the number of surrounding vehicles.

The state vector evolves over time as stipulated by the motion model,

$$\mathbf{z}_k = \mathbf{f}_{k-1}\left(\mathbf{z}_{k-1}, \mathbf{e}_{k-1}\right), \tag{3}$$

where $\mathbf{f}_{k-1}(\cdot)$ is a non-linear function and $\mathbf{e}_k$ is a noise process included to reflect both model uncertainties and the dynamics of the vehicles. Assuming that the vehicles move independently, we can consider the motion of each vehicle separately. To describe the motion of a vehicle we use a slightly modified version of the simplified bicycle model derived in [20], where the difference lies in the use of curvature instead of yaw-rate, $\dot{\psi}_k = v_k c_k$.

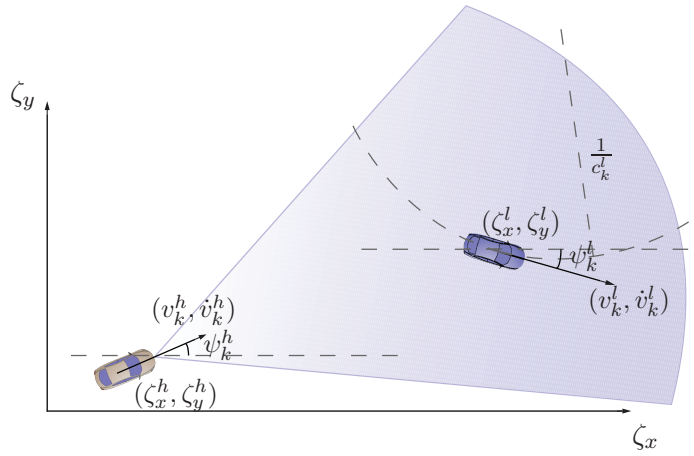


Figure 1: Coordinate system and state parametrization used in this paper.

2.2 Radar observations

For each discrete time k a radar provides M_k detections, where M_k^t detections originate from the tracked vehicles and M_k^c are clutter detections. All detections are stored in the unordered (unlabeled) measurement vector,

$$\mathbf{y}_k = \left[\left(\mathbf{y}_k^1\right)^T \left(\mathbf{y}_k^2\right)^T \cdots \left(\mathbf{y}_k^{M_k}\right)^T \right]^T. \tag{4}$$

Each detection is defined as

$$\mathbf{y}_k^i = \left[r_k^i \ \dot{r}_k^i \ \phi_k^i \right]^T, \tag{5}$$

where r_k^i is related to the range, $\dot{r}_k^i$ to the range rate, and ϕ_k^i to the angle to the object that gave rise to the detection relative to the sensor.

Let us define an ordered collection of the detections originating from the tracked vehicles as $\mathbf{y}_k^t$ and a collection of those originating from clutter as $\mathbf{y}_k^c$. These vectors are related to the measurement vector, $\mathbf{y}_k$, through an unknown permutation matrix, $\mathbf{\Pi}_p^{M_k}$, with dimension $[M_k \text{x} M_k]$,

$$\mathbf{y}_k = \left(\mathbf{\Pi}_p^{M_k} \otimes \mathbf{I}_{3\text{x}3}\right) \begin{bmatrix} \mathbf{y}_k^c \\ \mathbf{y}_k^t \end{bmatrix}. \tag{6}$$

where $\otimes$ is the Kronecker product and $\mathbf{I}_{3\text{x}3}$ is a three-by-three identity matrix. The purpose of $\mathbf{\Pi}_p^{M_k}$ is to describe mathematically that the measurement origin (data association) is unknown. In our model, all permutation matrices $\mathbf{\Pi}_p^{M_k}$ are equally likely, which means that the order of the detections in $\mathbf{y}_k$ is completely unknown (random). The treatment of this uncertainty is an important part in the derivation of the tracking framework and is further detailed in Section 4. However, let us first define $\mathbf{y}_k^c$ and $\mathbf{y}_k^t$ in more detail.

Clutter detections

It is commonly assumed, see e.g. [2], that $\mathbf{y}_k^c$ is described by a homogenous Poisson process in the observation space according to

$$\mathbf{y}_{k,i}^c \sim \text{Uniform}(V), \quad M_k^c \sim \text{Poisson}(\mu V), \tag{7}$$

where $\mathbf{y}_{k,i}^c$ is the i^{th} clutter measurement, μ is the clutter intensity and V is the volume of the observation space. In addition, we assume that the clutter detections are independent from each other and the target detections.

Target detections

Given $\mathbf{z}_k$, we assume it is possible to partition the visible reflections centers into N_k^g well separated groups, where each group can render multiple detections. Furthermore, we assume that the number of target detections for group n, $M_{k,n}^t$, has a probability mass function

$$\Pr\left\{M_{k,n}^t \middle|\ \mathbf{z}_k\right\}. \tag{8}$$

that we can model. Conditioned on $M_{k,n}^t$, the detections from group n can be described using a sensor model

$$\mathbf{y}_{k,n}^t = \mathbf{h}_k^n\left(\mathbf{z}_k, \mathbf{w}_k, M_{k,n}^t\right), \tag{9}$$

where $\mathbf{w}_k$ is a measurement noise process capturing both model uncertainties and measurement disturbances. From (9) we can generate

$$\mathbf{y}_k^t = \left[\left(\mathbf{y}_{k,1}^t\right)^T \ \left(\mathbf{y}_{k,2}^t\right)^T \ \cdots \ \left(\mathbf{y}_{k,N_k^g}^t\right)^T\right]^T, \tag{10}$$

and the total number of target detections is given by

$$M_k^t = \sum_{n=1}^{N_k^g} M_{k,n}^t. \tag{11}$$

2.3 Objectives

The main objective of this paper is to improve tracking performance by accurately modeling the radar response from the vehicles. To accomplish this, we need to derive an accurate model of the radar detections which is also suitable to be used in a tracking framework. In this section we discuss the objectives of the radar sensor model and the vehicle tracking framework separately.

Sensor model

The aim of the sensor model is to describe the statistical behavior of the measurements, given $\mathbf{z}_k$. The behavior of the clutter detections is readily given by (7), but modelling the target detections (8) – (9) is more complicated. It is crucial that these models capture the behavior of the vehicle detections from different aspect angles and at all ranges [21]. A vehicle radar response model, i.e., expressions for (8) – (9), that considers these aspects is derived in Section 3.

Tracking framework

Assuming that the number of vehicles is known, the objective of the tracking filter is to recursively calculate the posterior probability density function (pdf) $p\left(\mathbf{z}_k \middle| \mathbf{Y}_k\right)$, where $\mathbf{Y}_k \triangleq \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k\}$ contains all the available observations up to and including time k. From $p\left(\mathbf{z}_k \middle| \mathbf{Y}_k\right)$, it is then possible to compute estimates and uncertainty measures of $\mathbf{z}_k$. The calculation of $p\left(\mathbf{z}_k \middle| \mathbf{Y}_k\right)$ is feasible if we have knowledge regarding two specific models [2], namely the motion model, defined in (3) and the sensor model, defined by (7) - (9).

To arrive at a computationally tractable solution, we restrict our tracking filter to an *lmmse* estimator of $\mathbf{z}_k$. As such, only the first two moments of $p\left(\mathbf{z}_k \middle| \mathbf{Y}_k\right)$ need to be calculated, i.e.,

$$\hat{\mathbf{z}}_{k|k} = \mathrm{E}\left\{\mathbf{z}_k \middle| \mathbf{Y}_k\right\}, \quad \hat{\mathbf{P}}_{k|k} = \mathrm{Cov}\left\{\mathbf{z}_k \middle| \mathbf{Y}_k\right\}. \tag{12}$$

However, due to non-linearities in both the process and measurement model it is difficult to find an exact solution to (12). Instead, filters which approximate these moments are commonly used, e.g., the *extended Kalman filter* (EKF) [22] or the *unscented Kalman filter* (UKF) [18]. The latter is derived for the proposed sensor model in Section 4, treating the uncertainty in measurement origin (data association) modelled by the unknown permutation matrix $\mathbf{\Pi}_p^{M_k}$.

3 Radar sensor model

Our proposed sensor model is based on the findings presented in [11], where the radar response from vehicles is modeled as originating from reflection centers (features) on the vehicles more likely to reflect the incident radar wave. Due to limitations in radar signal bandwidth, pulse duration and antenna aperture size, radar sensors are not capable of resolving reflection centers that are too closely spaced. As such, not all of these reflectors are always resolvable and the response from some might merge to form a joint detection. In [11], a mapping is proposed for how to transform the vehicle states to a set of reflector positions in observation space. Additionally, a scheme is described for how to form clusters of those reflection centers that are unresolvable and how to model detections from these clusters. As this model was developed for simulation purposes, rather than for use in a tracking system, it neglects important probabilistic descriptions needed in

the tracking context. For example, the model requires the received signal strength of each individual reflection center to be known. Moreover, conditioned on the signal strength and $\mathbf{z}_k$, both which reflectors are clustered and the positions of the resulting clusters are deterministic. In the tracking context, it is not realistic that the signal strength is known a-priori and consequently, we do not know which reflectors are clustered or the position of the resulting detections from the vehicle.

In this section, we derive a radar sensor model using a stochastic description of the received signal amplitude from each reflector center, arriving at a model more suitable in a tracking framework. The derivation is conducted in four steps which are shown in Fig. 2 and summarized as follows. First, based on the model in [11], the positions of the reflection centers of the vehicles in $\mathbf{z}_k$ are mapped to the observation space. Second, we form all possible clusters of reflection centers which may generate merged detections. Due to uncertainty regarding which reflectors are resolved and which are not, the resulting probability density of reflector cluster could be highly multi-modal. Third, to alleviate this multi-modality, we group reflectors which may belong to the same cluster, and approximate the cluster density by marginalizing over all cluster possibilities. The result is a description of reflector groups capable of generating multiple measurements. Finally, depending on the probability of detecting the possible clusters in each group, we find an expression for (8). Assuming that the measurement noise is additive and Gaussian, we now write (9) on the form

$$\mathbf{y}_{k,n}^t = \mathcal{G}_n\left(\mathbf{z}_k, M_{k,n}^t\right) + \mathbf{w}_k,\ \mathbf{w}_k \sim \mathcal{N}\left(\mathbf{0}, \mathbf{W}_k^{M_{k,n}^t}\right) \tag{13}$$

where

$$\mathbf{W}_k^{M_{k,n}^t} = \mathbf{I}_{M_{k,n}^t \mathrm{x} M_{k,n}^t} \otimes \mathbf{W}_k. \tag{14}$$

The function $\mathcal{G}_n(\cdot)$ maps $\mathbf{z}_k$ to the target measurement vector for group n, given knowledge regarding the number of measurements generated by the group, $M_{k,n}^t$. Note that although we condition on $\mathbf{z}_k$ and the number of detections from the group, $\mathcal{G}_n(\mathbf{z}_k, M_{k,n}^t)$ is still stochastic due to uncertainty in which reflectors that are clustered (we call this clustering uncertainty).

The following sections present the derivation of the distribution of $\mathcal{G}_n(\mathbf{z}_k, M_{k,n}^t)$ and $M_{k,n}^t$ using the steps described above. To simplify notation, the time dependence is omitted and all stochastic variables are conditioned on $\mathbf{z}_k$, even though it is not explicitly stated.

3.1 Reflection center model

According to the model in [11], the studied radar only receives reflections from a discrete set of points on a vehicle, so called *reflection centers*. The different reflection centers are divided into two categories: point reflectors and plane reflectors. Fig. 3 displays the configuration of point reflectors suggested in [11], where the

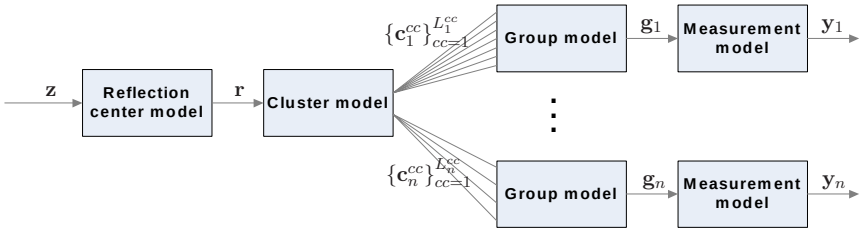


Figure 2: Schematic view of the measurement generation process in our proposed radar sensor model. The notation used in the figure is introduced in subsequent sections.

reflectors are placed in the vehicle wheel houses and corners. Associated with each reflector is a visibility region, indicated by cones in Fig. 3; a reflector can only render a reflection if the sensor is within this region. The plane reflectors are modelled as circle sectors typically describing the sides of the vehicle. Furthermore, it is assumed that the radar only can receive a reflection from these plane reflector if there is a point on the surface which normal points directly towards the sensor. The reflecting point on a surface therefore depends on the position of the sensor, and may change over time as the vehicle moves relative to the sensor platform.

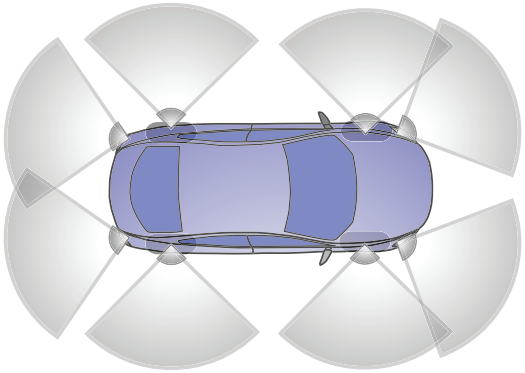


Figure 3: Vehicle reflection centers with associated visibility regions.

Given the vehicles' positions and physical dimensions, each reflector i has a deterministic position in observation space, denoted $\mathbf{r}_i = [r_i, \dot{r}_i, \phi_i]^T$ and expected signal power σ_i, expressed as

$$\left[(\mathbf{r}_i)^T \ \sigma_i\right]^T = \mathcal{R}_i(\mathbf{z}), \tag{15}$$

where the mapping $\mathcal{R}$ is defined in Appendix A. Although the physical dimensions of the observed objects are assumed known in this paper, in a sensor data fusion system, information regarding object extent could be provided by, e.g., a vision sensor and/or vehicle-to-vehicle communication.

In addition to the position of the reflector in observation space, it is also important to model the signal amplitude, A_i, of the received reflection. This is an

important model feature as the probability of detecting a reflector depends on the strength of the received echo, and the position of a merged detection depends on the relative amplitude of the included echoes. The amplitude model used in [11] is a deterministic function of the radar antenna pattern as well as the position and visibility of the reflectors. We instead propose to use a Swerling I model [23] for the amplitude of the reflected signal, where the reflection amplitudes are modeled as fluctuating according to the Rayleigh distribution,

$$A_i \sim \text{Rayleigh}(\sigma_i). \tag{16}$$

As is shown in the coming section, using a stochastic amplitude model instead of a deterministic enables us to describe uncertainty regarding number of vehicle detections as well as their positions.

3.2 Cluster model

In Section 3.1 we presented a model for the vehicle response from a radar with infinite resolution through the mapping $\mathbf{z} \xrightarrow{\mathcal{R}} \mathbf{r}$. However, a sensor with limited resolution cannot resolve too closely spaced reflectors. To model this behavior, a resolution cell is used

$$\mathbf{\Delta_d} = [\Delta_r \ \Delta_{\dot{r}} \ \Delta_\phi]^T \tag{17}$$

and two radar responses which are not separated more than $\mathbf{\Delta_d}$, in all three dimensions, yield a merged detection. Unfortunately, the situation is more complicated for multiple reflectors, as unresolved clusters can be formed in several ways.

Cluster formation

In [11], the following algorithm is used to map reflectors into clusters, an operation here denotes as $\mathbf{r} \xrightarrow{\mathcal{C}} \mathbf{c}^{cc}$:

i) Find the reflector with the strongest amplitude, A_i.

ii) Form a cluster by identifying the reflectors which are within the resolution cell (centered at $\mathbf{r}_i$).

iii) Repeat *i*) and *ii*) with the remaining reflectors, until no reflectors are left.

The clustering algorithm above, can be used to divide the set of all visible reflectors into clusters and we refer to a description of all resulting clusters as a *cluster constellation*. However, it is important to note that since the amplitudes of the reflections are stochastic (in contrast to [11]), several different cluster constellations may be possible, even for a given $\mathbf{z}$. For notation, we construct a list of all possible constellations, and introduce the variable cc as a pointer to the cluster constellations in that list. The total number of constellations in the list is denoted N^{cc} (and consequently $cc \in \{1, 2, \ldots, N^{cc}\}$), and the number of reflector clusters

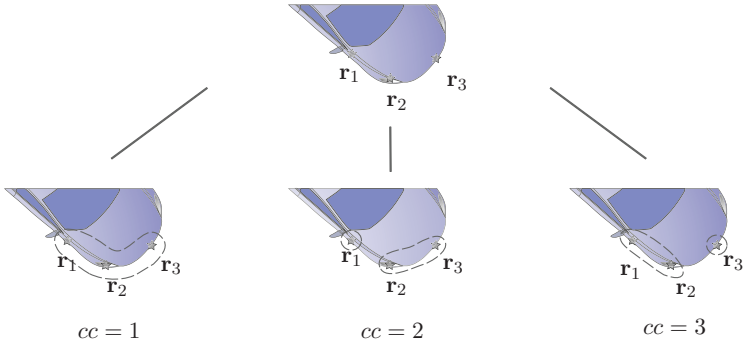


Figure 4: Example of a vehicle with three reflectors (top) and three different cluster constellations (bottom). The dashed line corresponds to a reflector cluster.

in constellation cc is L^{cc}. Fig. 4 illustrates these concepts by showing three possible cluster constellations in a simple example. Here, $N^{cc} = 3$ with $L^1 = 1$, $L^2 = 2$ and $L^3 = 2$ as the first constellation clusters all reflectors and the other two contains two clusters each.

Cluster i in constellation cc, containing J reflectors[1] with indices $i_1, \ldots, i_J$, can at most generate one detection which in that case can be modelled as $\mathbf{y}_i^{cc} = \mathbf{c}_i^{cc} + \mathbf{w}_i^{cc}$ where $\mathbf{w}_i^{cc} \sim \mathcal{N}(\mathbf{0}, \mathbf{W})$. The signal component, $\mathbf{c}_i^{cc}$, is modelled as a weighted sum [11] of reflector components:

$$\mathbf{c}_i^{cc} = \sum_{l=1}^{J} w_{i_l} \mathbf{r}_{i_l}, \tag{18}$$

where

$$w_{i_l} = \frac{A_{i_l}}{\sum_{l=1}^{J} A_{i_l}}. \tag{19}$$

Although this is a rather simplified model of the underlying physical phenomenon of merged measurements (such as target glint) [4,24], it serves the purposes for our radar sensor model. Since the amplitudes are stochastic, so are the weights (19) and the signal component of the cluster (18). As for the reflector detections, the received amplitude of a cluster is also Rayleigh distributed but with the parameter

$$\sigma_i^{cc} = \sqrt{\sum_{l=1}^{J} \sigma_{i_l}^2}. \tag{20}$$

Cluster density

For a cluster, the distribution of its position is defined by (16), (18) - (19), which is difficult to evaluate. As the aim is to use the proposed sensor model in a Kalman

[1]The notation for the number of reflectors will change as we can get more specific. In this section we use J to indicate the number of reflectors in a generic cluster.

filter framework, it is convenient to approximate $p\left(\mathbf{c}_i^{cc}\middle|\,\mathbf{z}\right)$ as a Gaussian density with the same first two moments as the underlying distribution.

Let overscore denote the expected value of stochastic variables, such that, e.g., $\bar{A}_i = E\{A_i\}$. Further, let $\Delta\mathbf{r}_{i_l} = \mathbf{r}_{i_l} - \bar{\mathbf{c}}_i^{cc}$, $\Delta w_{i_l} = w_{i_l} - \bar{w}_{i_l}$ and set $S_i = \sum_{l=1}^{J} A_{i_l}$. The first moment of $\mathbf{c}_i^{cc}$, as given by (18), is

$$\bar{\mathbf{c}}_i^{cc} = \sum_{l=1}^{J} \bar{w}_{i_l}\mathbf{r}_{i_l} \tag{21}$$

and after some manipulations an expression for the covariance can be found as

$$\mathbf{C}_i^{cc} = \sum_{s,\,t=1}^{J} \Delta\mathbf{r}_{i_s}\left(\Delta\mathbf{r}_{i_t}\right)^T E\left\{\Delta w_{i_s}\Delta w_{i_t}\right\}. \tag{22}$$

The position of each reflector, $\mathbf{r}_{i_l}$, is given by transformation (15), but we also need to express $\bar{w}_{i_l}$ and $\mathrm{Cov}\{w_{i_s}, w_{i_t}\}$. As the moments of a Rayleigh distribution are well known, approximations of these quantities are readily found through Taylor expansion,

$$w_{i_l} = \frac{A_{i_l}}{S_i} \approx \frac{\bar{A}_{i_l}}{\bar{S}_i} + \frac{A_{i_l}}{\bar{S}_i} - \frac{S_i\bar{A}_{i_l}}{\bar{S}_i^2}. \tag{23}$$

More details on the derivation of the mean and covariance of $\mathbf{c}_i^{cc}$ as well as the approximations used are found in Appendix B.

To summarize, we propose a stochastic mapping of reflectors into cluster constellations, $\mathbf{r} \xrightarrow{\mathcal{C}(\cdot)} \{\mathbf{c}^{cc}\}_{cc=1}^{N^{cc}}$, where N^{cc} is deterministic but both cc and $\mathbf{c}^{cc} = [\mathbf{c}_1^{cc}, \ldots, \mathbf{c}_{L^{cc}}^{cc}]$ are stochastic. The density of each cluster in each constellation, $\mathbf{c}_i^{cc}$, is approximated as a Gaussian density,

$$p\left(\mathbf{c}_i^{cc}\middle|\,cc, \mathbf{z}\right) = \mathcal{N}\left(\mathbf{c}_i^{cc};\ \bar{\mathbf{c}}_i^{cc}, \mathbf{C}_i^{cc}\right), \tag{24}$$

where $\bar{\mathbf{c}}_i^{cc}$ and $\mathbf{C}_i^{cc}$ are given by (21) and (22), respectively.

3.3 Group model

In multi-target scenarios, the total number of possible clusters, $\sum_{cc=1}^{N^{cc}} L^{cc}$, can be significant. Hence, it could be difficult to find a computationally feasible solution for associating measurements to individual clusters. To mitigate this difficulty, we suggest to form reflector groups containing reflectors that are likely to get clustered, and describe the measurement distribution by marginalizing over the cluster constellations. Let a group be a set of reflectors, formed such that for every reflector i in the group, all other reflectors belonging to one or more clusters with reflector i are also included. As a consequence, each reflector i in the group is positioned within $\Delta_{\mathbf{d}}$ to at least one other reflector in the group. The number of groups, N^g, is then the total number of such partitions of the reflectors. Fig. 5

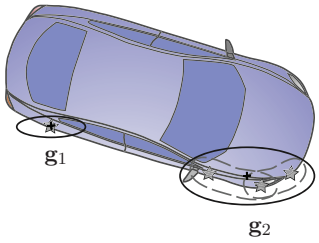


Figure 5: The formation of two reflector groups (solid lines).

displays a scenario where two groups, i.e., $N^g = 2$, are formed; one containing only one reflection center in the rear wheel and the other composed of three reflection centers in the front. A suboptimal, but simplified, solution to the association problem is obtained by associating the detections to the reflector groups. By ignoring which specific cluster in a group that gave rise to a detection, the number of hypotheses are reduced substantially.

Each group is viewed as an entity which can generate multiple and independent detections. The number of detections from group n is denoted by M_n^t, and the signal components (the positions) as $\tilde{\mathbf{g}}_n = \left[\left(\mathbf{g}_{n,1}\right)^T, \ldots, \left(\mathbf{g}_{n,M_n^t}\right)^T\right]^T$. Using this notation, (13) can be written as

$$\mathbf{y}_n^t = \tilde{\mathbf{g}}_n + \mathbf{w}, \tag{25}$$

As only reflectors within each group can form clusters with each other, we can consider the cluster constellations for each group independently. For group n we can generate N_n^{cc} cluster constellations, and we let cc in this case indicate one specific constellation in this group and L_n^{cc} denote the number of clusters in this cluster constellation. A new list of cluster constellations is generated for each group and cc is used to index one of the constellations. Further, let $\mathbf{c}_{n,l}^{cc}$ denote the signal component of the l^{th} cluster in constellation cc for group n, and $P_n^{cc}(l)$ denote the detection probability of this cluster - a probability easily computed from the Rayleigh assumption in (16) and (20). If we assume that all possible cluster constellations are equally likely, we can describe the pdf of $\tilde{\mathbf{g}}_n$ by approximating its components $\mathbf{g}_{n,i}$ as independent and identically distributed with the density[2]

$$p\left(\mathbf{g}_{n,i}\right) = \frac{1}{N_n^{cc}} \sum_{cc=1}^{N_n^{cc}} \sum_{l=1}^{L_n^{cc}} q_{n,l}^{cc} p_{\mathbf{c}_{n,l}^{cc}}(\mathbf{g}_{n,i}), \tag{26}$$

where the weights $q_{n,l}^{cc}$ are defined as

$$q_{n,l}^{cc} = \frac{P_n^{cc}(l)}{\sum_{m=1}^{L_n^{cc}} P_n^{cc}(m)}. \tag{27}$$

[2]In (26), the notation $p_{\mathbf{c}_{n,l}^{cc}}\left(\mathbf{g}_{n,i}\right)$ should be interpreted as the pdf of $\mathbf{c}_{n,l}^{cc}$ evaluated at $\mathbf{g}_{n,i}$.

To further simplify the implementation of a tracking algorithm based on this model we make a Gaussian approximation, $p(\mathbf{g}_{n,i}) \approx \mathcal{N}(\mathbf{g}_{n,i}; \bar{\mathbf{g}}_{n,i}, \mathbf{C}_n)$. Using the approximation in (24), the expected value, $\bar{\mathbf{g}}_n = \mathrm{E}\{\mathbf{g}_{n,i}\}$ is given by

$$\bar{\mathbf{g}}_n = \frac{1}{N_n^{cc}} \sum_{cc=1}^{N_n^{cc}} \sum_{l=1}^{L_n^{cc}} q_{n,l}^{cc} \bar{\mathbf{c}}_{n,l}^{cc} \tag{28}$$

and the second moment, $\mathbf{C}_n = \mathrm{E}\left\{(\mathbf{g}_{n,i} - \bar{\mathbf{g}}_n)(\mathbf{g}_{n,i} - \bar{\mathbf{g}}_n)^T\right\}$ by

$$\mathbf{C}_n = \sum_{cc=1}^{N_n^{cc}} \sum_{l=1}^{L_n^{cc}} \frac{q_{n,l}^{cc}}{N_n^{cc}} \left(\mathbf{C}_{n,l}^{cc} + \left(\bar{\mathbf{g}}_n - \bar{\mathbf{c}}_{n,l}^{cc}\right) \left(\bar{\mathbf{g}}_n - \bar{\mathbf{c}}_{n,l}^{cc}\right)^T \right), \tag{29}$$

where $\bar{\mathbf{c}}_{n,l}^{cc}$ and $\mathbf{C}_{n,l}^{cc}$ is given by (21) and (22), respectively.

Additionally, to complete the description of the groups, we need to calculate the probability mass function M_n^t, the number of detections originating from group n. By again assuming that all cluster constellations are equally likely, we have

$$\Pr\left\{M_n^t\right\} = \frac{1}{N_n^{cc}} \sum_{cc=1}^{N_n^{cc}} \Pr\left\{M_n^t \middle| \, cc\right\}, \tag{30}$$

where $\Pr\left\{M_n^t \middle| \, cc\right\}$ is easily calculated from $P_n^{cc}(l)$.

In summary; we group closely spaced reflectors and use the cluster description to calculate the expected signal component each group, its covariance matrix and the probability mass function for M_n^t, the number of detections from group n.

3.4 Target measurement model

In Sections 3.1, 3.2 and 3.3 we describe three mappings, $\mathcal{R}(\cdot)$, $\mathcal{C}(\cdot)$ and $\mathcal{G}(\cdot)$, relating the position of the vehicles to the measurement distribution. The procedure can be depicted as

$$\mathbf{z} \xrightarrow{\mathcal{R}(\cdot)} \mathbf{r} \xrightarrow{\mathcal{C}(\cdot)} \{\mathbf{c}^{cc}\}_{cc=1}^{N^{cc}} \xrightarrow{\mathcal{G}(\cdot)} \mathbf{g}_n, M_n^t,$$

where the first two mappings are deterministic whereas the last two are stochastic due to uncertainty in the resolution capabilities of the sensor. The transformation $\mathbf{z} \xrightarrow{\mathcal{R}(\cdot)} \mathbf{r}$ describes the signal components of strong (vehicle related) radar reflectors in observation space. By modelling the resolution capability of the sensor, $\mathcal{C}(\cdot)$, we form a set of cluster constellations, $\{\mathbf{c}^{cc}\}_{cc=1}^{N^{cc}}$. To reduce the complexity of the data association problem, we form groups of reflectors that belong to the same cluster constellations, $\mathcal{G}(\cdot)$. The groups are described by their spatial density, $p(\mathbf{g}_n) \approx \mathcal{N}(\mathbf{g}_n; \bar{\mathbf{g}}_n, \mathbf{C}_n)$, and the probability mass function $\Pr\left\{M_n^t\right\}$, describing the number of target detections from each group. The result is a description of target measurements originating from groups of reflectors, where each group

n generates M_n^t independent and identically distributed measurements. The i^{th} detection from group n,

$$\mathbf{y}_{n,i}^t \sim \mathcal{N}(\bar{\mathbf{g}}_n, \mathbf{C}_n + \mathbf{W}), \tag{31}$$

is an independent of all other detections (conditioned on $\mathbf{z}$). The complete target measurement vector $\mathbf{y}^t$ can be generated by drawing the number of target detections from the group, M_n^t, according to (30), for each group $n = 1 \dots N^g$. Subsequently, construct $\mathbf{y}_n^t$ by generating M_n^t independent realizations of $\mathbf{y}_{n,i}^t$ conforming to (31). The complete target measurement vector is formed by concatenating all group measurements as defined in (10). This concludes the derivation of our sensor model.

4 Tracking framework

In this section we present a tracking framework for recursively calculating the posterior density, $p\left(\mathbf{z}_k \middle| \mathbf{Y}_k\right)$, using the proposed sensor model derived in Section 3. To handle uncertainty in the number of target and clutter detections as well as the random permutation matrix, $\mathbf{\Pi}_p^{M_k}$, in the calculation of $p\left(\mathbf{z}_k \middle| \mathbf{Y}_k\right)$, it is convenient to introduce a data association hypothesis vector, $\boldsymbol{\lambda}$. The purpose of this vector is to associate detection number j in $\mathbf{y}_k$ to a certain group, n. Consequently, $\boldsymbol{\lambda}(j) = n$, if measurement j originates from group n and, $\boldsymbol{\lambda}(j) = 0$, if it is to be regarded as clutter. Using this description, the sensor model can be written as

$$p\left(\mathbf{y}_k^j \middle| \boldsymbol{\lambda}, \mathbf{z}_k\right) = \begin{cases} \mathcal{N}\left(\mathbf{y}_k^j; \bar{\mathbf{g}}_{\boldsymbol{\lambda}(j)}, \mathbf{C}_{\boldsymbol{\lambda}(j)} + \mathbf{W}_k\right) & \text{if } \boldsymbol{\lambda}(j) \neq 0 \\ \left(\frac{1}{V}\right) & \text{if } \boldsymbol{\lambda}(j) = 0. \end{cases} \tag{32}$$

This formulation makes it possible to associate measurements to group n according to the support of $\Pr\left\{M_n^t \middle| \mathbf{z}\right\}$. That is, each group can generate multiple detections, each carrying information regarding the state of the vehicle. This distinguishes our sensor model from the classical point source model.

By considering all possible data association hypothesis, the posterior density can be formed as

$$p\left(\mathbf{z}_k \middle| \mathbf{Y}_k\right) = \sum_{\boldsymbol{\lambda}} p\left(\mathbf{z}_k \middle| \boldsymbol{\lambda}, \mathbf{Y}_k\right) \Pr\left\{\boldsymbol{\lambda} \middle| \mathbf{Y}_k\right\}, \tag{33}$$

where $p\left(\mathbf{z}_k \middle| \boldsymbol{\lambda}, \mathbf{Y}_k\right)$ is the posterior density without data association uncertainty and $\Pr\left\{\boldsymbol{\lambda} \middle| \mathbf{Y}_k\right\}$ is the probability of that association. Due to, e.g., non-linearities in (15) and the dimensionality of the data association problem, it is difficult to find an exact solution to (33). Instead, we resort to an approximate solution.

In the literature it is possible to find several possible approaches, such as *particle filters* [25], *Multiple Hypothesis Tracking* (MHT) filters [26] or the *Probabilistic Multiple Hypothesis Tracker* (PMHT) [9] to handle or simplify these types of problems. To make the implementation suitable for real-time applications with limited

capacity to batch measurements, we propose a Kalman-like filter framework [27] employing a generalized version of the *Joint Probabilistic Data Association* (JPDA) algorithm [19]. The generalization of the JPDA algorithm for this problem consists in allowing multiple measurements to originate from the same group, in contrast to at most one in the original JPDA formulation. The conditional posterior density, $p\left(\mathbf{z}_k \middle| \boldsymbol{\lambda}, \mathbf{Y}_k\right)$, can be calculated for each $\boldsymbol{\lambda}$, and we approximate this distribution by a Gaussian density with mean $\hat{\mathbf{z}}^{\boldsymbol{\lambda}}_{k|k}$ and covariance $\mathbf{P}^{\boldsymbol{\lambda}}_{k|k}$. Estimates of $\hat{\mathbf{z}}^{\boldsymbol{\lambda}}_{k|k}$ and $\mathbf{P}^{\boldsymbol{\lambda}}_{k|k}$ are found using the UT [18] through

$$\hat{\mathbf{z}}^{\boldsymbol{\lambda}}_{k|k} = \hat{\mathbf{z}}_{k|k-1} + \mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{zy}} \left(\mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{yy}}\right)^{-1} \left(\mathbf{y}^{\boldsymbol{\lambda}}_k - \hat{\mathbf{y}}^{\boldsymbol{\lambda}}_{k|k-1}\right) \tag{34}$$

$$\mathbf{P}^{\boldsymbol{\lambda}}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{zy}} \left(\mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{yy}}\right)^{-1} \left(\mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{zy}}\right)^{T}, \tag{35}$$

where $\hat{\mathbf{z}}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ are estimates of the mean and covariance of $\mathbf{z}_k \middle| \mathbf{Y}_{k-1}$, respectively. The covariance, $\mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{yy}}$, is the innovation covariance under the data association hypothesis $\boldsymbol{\lambda}$ and, $\mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{zy}}$ is the corresponding cross covariance between the state and the measurements. In accordance with the JPDA idea, the resulting Gaussian mixture (33) is also approximated as a single Gaussian where the contribution from the individual densities are weighted by their hypothesis probability, $\Pr\left\{\boldsymbol{\lambda} \middle| \mathbf{Y}_k\right\}$. Given a Gaussian prior density, $p\left(\mathbf{z}_{k-1} \middle| \mathbf{Y}_{k-1}\right)$, the proposed approach is briefly outlined below.

1. *State prediction*: Estimate $\hat{\mathbf{z}}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ by propagating $\mathbf{z}_{k-1} \middle| \mathbf{Y}_{k-1}$ through the motion model (3) using the unscented transform.

2. *Measurement prediction*: Transform $\mathbf{z}_k \middle| \mathbf{Y}_{k-1}$ to the observation space, using the series of mappings derived in Section 3 and the unscented transform to retain an approximation of the predicted group measurement

   $$\hat{\mathbf{y}}^t_{k,n} = \hat{\mathbf{g}}_n = \mathrm{E}\left\{\mathbf{g}_n \middle| \mathbf{Y}_{k-1}\right\}, \tag{36}$$

   as well as the predicted group covariances

   $$\mathbf{P}^n_{\mathbf{gg}} = \mathrm{Cov}\left\{\mathbf{g}_{n,i}, \mathbf{g}_{n,i} \middle| \mathbf{Y}_{k-1}\right\}, \tag{37}$$

   $$\mathbf{P}^{nm}_{\mathbf{gg}} = \mathrm{Cov}\left\{\mathbf{g}_{n,i}, \mathbf{g}_{m,j} \middle| \mathbf{Y}_{k-1}\right\}, \tag{38}$$

   where $i \neq j$ if $n = m$. The innovation covariance, $\mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{yy}}$, is given by (37) – (38) and the measurement noise covariance, $\mathbf{W}_k$.

3. *Data association*: Use $\hat{\mathbf{g}}_n$ and $\mathbf{P}^n_{\mathbf{yy}} = \mathbf{P}^n_{\mathbf{gg}} + \mathbf{W}_k$ to perform measurement gating. Then generate the set of all possible measurement to group association hypotheses and calculate their probabilities $\Pr\left\{\boldsymbol{\lambda} \middle| \mathbf{Y}_k\right\}$.

4. *Measurement update*: For all $\boldsymbol{\lambda}$, form the needed entities in (34) - (35) and approximate $p\left(\mathbf{z}_k \middle| \boldsymbol{\lambda}, \mathbf{Y}_k\right)$. Finally, $p\left(\mathbf{z}_k \middle| \mathbf{Y}_k\right)$, is found by marginalizing the data association hypotheses, see (33).

A detailed description of the different steps is given in the following sections.

4.1 State prediction

The state prediction is performed by calculating a Gaussian approximation of the predicted density

$$p\left(\mathbf{z}_k \middle| \mathbf{Y}_{k-1}\right) = \int p\left(\mathbf{z}_k \middle| \mathbf{z}_{k-1}\right) p\left(\mathbf{z}_{k-1} \middle| \mathbf{Y}_{k-1}\right) d\mathbf{z}_{k-1}. \tag{39}$$

This approximation is found by propagating $p\left(\mathbf{z}_{k-1} \middle| \mathbf{Y}_{k-1}\right)$ through (3) using the unscented transform [18] as

$$p\left(\mathbf{z}_k \middle| \mathbf{Y}_{k-1}\right) \approx \mathcal{N}\left(\mathbf{z}_k; \hat{\mathbf{z}}_{k|k-1}, \mathbf{P}_{k|k-1}\right). \tag{40}$$

Assuming the vehicles move independently, the unscented transform can be performed for each vehicle separately.

4.2 Measurement prediction

In Section 3 we derived a radar sensor model conditioned on the state through a series of mappings. To form the expressions (34) - (35) we need estimates of the first and the second-order moments (36), (37) and (38). Again we use the unscented transform to find approximations of these moments. However, for our proposed group measurement model, the approximation is not as straightforward as for the motion model. As such, it requires some additional discussion.

Using the unscented transform described in [18], we choose $2n_z+1$ deterministic sigma points with associated weights, where n_z is the dimensionality of $\mathbf{z}_k$. The sigma points and their weights are chosen such that they capture the first two moments of (40) exactly. Let us denote the set of sigma points with associated weights as,

$$\left\{\mathbf{Z}_k^i, u_k^i\right\}_{i=1}^{2n_z+1}. \tag{41}$$

Although not required, we choose the weights such that $u_k^i > 0, \forall\, i = 1 \dots 2n_z+1$. This is used to avoid risks associated with using the unscented transform in high dimensions, such as the risk of estimating non-positive definite covariance matrices and a mean situated far away from each propagated sigma point.

By propagating each sigma point through the mapping (15), we receive the sigma point sets

$$\left\{\mathbf{R}_k^i, u_k^i\right\}_{i=1}^{2n_z+1}, \quad \left\{\mathbf{\Sigma}_k^i, u_k^i\right\}_{i=1}^{2n_z+1} \tag{42}$$

where $\mathbf{R}_k^i$ describe the reflector positions of the i^{th} sigma point in observation space and $\mathbf{\Sigma}_k^i$ their expected signal power. From (42) we can form estimates of the first two moments of $\mathbf{r}_k \middle| \mathbf{Y}_{k-1}$ according to

$$\hat{\mathbf{r}}_{k|k-1} \approx \sum_i u_k^i \mathbf{R}_k^i \tag{43}$$

$$\mathbf{P}_{\mathbf{rr}} \approx \sum_i u_k^i \left(\mathbf{R}_k^i - \hat{\mathbf{r}}_{k|k-1}\right)\left(\mathbf{R}_k^i - \hat{\mathbf{r}}_{k|k-1}\right)^T \tag{44}$$

where $\hat{\mathbf{r}}_{k|k-1}$ is the estimate of the reflector position. The covariance matrix, $\mathbf{P}_{\mathbf{rr}}$, has the following structure

$$\mathbf{P}_{\mathbf{rr}} = \begin{bmatrix} \mathbf{P}_{\mathbf{rr}}^{11} & \mathbf{P}_{\mathbf{rr}}^{12} & \dots & \mathbf{P}_{\mathbf{rr}}^{1n_r} \\ \mathbf{P}_{\mathbf{rr}}^{21} & \mathbf{P}_{\mathbf{rr}}^{22} & \dots & \mathbf{P}_{\mathbf{rr}}^{2n_r} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P}_{\mathbf{rr}}^{n_r 1} & \dots & \dots & \mathbf{P}_{\mathbf{rr}}^{n_r n_r} \end{bmatrix}, \tag{45}$$

where n_r is the total number of reflectors on the vehicles in $\mathbf{z}_k$. From $\mathbf{\Sigma}_k$, the estimated expected signal power is similarly attained as

$$\hat{\boldsymbol{\sigma}}_{k|k-1} \approx \sum_i u_k^i \mathbf{\Sigma}_k^i, \tag{46}$$

which is used in (30) to account for reflector visibility under state uncertainty, primarily in target heading. In practice, we only need to consider those reflectors which are visible, i.e., for which $\hat{\sigma}_{k|k-1}^i > 0$.

From (43), we determine which reflectors belong to the same group. As only reflectors within each group are able to form clusters with each other, we consider the cluster constellations for each group independently. Following the algorithm in Section 3.3, we form all possible cluster constellations for each group. Using the mapping (26) we can calculate two of the sought moments, (36) and (37), as

$$\hat{\mathbf{g}}_n = \frac{1}{N_n^{cc}} \sum_{cc=1}^{N_n^{cc}} \sum_{l=1}^{L_n^{cc}} q_{n,l}^{cc} \hat{\mathbf{c}}_{n,l}^{cc} \tag{47}$$

$$\mathbf{P}_{\mathbf{gg}}^n = \sum_{cc=1}^{N_n^{cc}} \sum_{l=1}^{L_n^{cc}} \frac{q_{n,l}^{cc}}{N_n^{cc}} \left(\mathbf{P}_{\mathbf{c}_l \mathbf{c}_l}^{cc,n} + \left(\hat{\mathbf{g}}_n - \hat{\mathbf{c}}_{n,l}^{cc} \right) \left(\hat{\mathbf{g}}_n - \hat{\mathbf{c}}_{n,l}^{cc} \right)^T \right). \tag{48}$$

Note that the weights, $q_{n,l}^{cc}$, are dependent on the estimated expected amplitudes, $\hat{\boldsymbol{\sigma}}_{k|k-1}$, through P_n^{cc}. Values for $\hat{\mathbf{c}}_{n,l}^{cc}$ and $\mathbf{P}_{\mathbf{c}_l \mathbf{c}_l}^{cc,n}$ are found using the same approximations as in the derivation of (21) - (22), detailed in Appendix B, and assuming independence between w_i and $\mathbf{r}_{k,i}$,

$$\hat{\mathbf{c}}_{n,l}^{cc} = \mathrm{E}\left\{ \mathbf{c}_{n,l}^{cc} \middle| cc, \mathbf{Y}_{k-1} \right\} \approx \sum_{i=1}^{J_{n,l}^{cc}} \bar{w}_{l_i} \hat{\mathbf{r}}_{k|k-1}^{l_i} \tag{49}$$

$$\begin{aligned} \mathbf{P}_{\mathbf{c}_l \mathbf{c}_l}^{cc,n} &= \mathrm{Cov}\left\{ \mathbf{c}_{n,l}^{cc} \middle| cc, \mathbf{Y}_{k-1} \right\} \approx \\ &\sum_{i,j=1}^{J_{n,l}^{cc}} \left(\mathbf{P}_{\mathbf{rr}}^{l_i l_j} + \left(\hat{\mathbf{r}}_{k|k-1}^{l_i} - \hat{\mathbf{c}}_{n,l}^{cc} \right) \left(\hat{\mathbf{r}}_{k|k-1}^{l_j} - \hat{\mathbf{c}}_{n,l}^{cc} \right)^T \right) \\ &\quad \times \mathrm{E}\left\{ w_{l_i} w_{l_j} \right\}, \end{aligned} \tag{50}$$

where, $l_1, \dots, l_{J_{n,l}^{cc}}$ are the indexes to the reflectors in the cluster under consideration. Using (47), (48) we can form the Gaussian approximation of the i^{th} predicted

measurement from group n as

$$p\left(\mathbf{y}_{k,n,i}^{t}\middle|\,\mathbf{Y}_{k-1}\right) \approx \mathcal{N}(\mathbf{y}_{k,n,i}^{t}; \hat{\mathbf{g}}_{n}, \mathbf{P}_{\mathbf{gg}}^{n} + \mathbf{W}_{k}). \tag{51}$$

The additional sought covariance, $\mathbf{P}_{\mathbf{gg}}^{nm}$, assesses the covariance between two detections from the same group or alternatively two detections from different groups. For the filter to be able to perform the state update for all groups jointly, it is important to accurately assess these correlations. Conditioned on $\mathbf{z}_k$, it is assumed that these detections are uncorrelated. Hence, the correlation only comes from uncertainty in $\mathbf{z}_k$ and we can approximate the group cross covariance as,

$$\mathbf{P}_{\mathbf{gg}}^{nm} = \sum_{i}^{J_n^{cc}} \sum_{j}^{J_m^{cc}} \bar{w}_{n_i} \bar{w}_{n_j} \mathbf{P}_{\mathbf{rr}}^{n_i m_j}, \tag{52}$$

where $\{n_i\}_{i=1}^{J_n^{cc}}$ and $\{m_i\}_{i=1}^{J_m^{cc}}$ are the indices of all the reflectors in each group, respectively. The weights, $\bar{w}_{n_i}$ and $\bar{w}_{n_i}$, are calculated using the assumption that all reflectors in each group are clustered as, for example,

$$\bar{w}_{n_i} = \mathrm{E}\left\{\frac{A_{n_i}}{\sum_{j}^{J_n^{cc}} A_{n_j}}\right\}. \tag{53}$$

The cross covariance between detections from the same group is simply found when $n = m$.

4.3 Data association

In difference to standard JPDA, the generalized version of the JPDA algorithm proposed here considers the possibility that a single track can generate multiple measurements. To avoid unlikely data association hypotheses, we employ an ellipsoidal gate [2] centered at the group mean using (51). From the gated measurements and knowledge regarding the maximum number of detections generated by group n ($\max M_{k,n}^{t}$), it is possible to construct the set of all *local hypotheses*, i.e., the set of all feasible associations between $\mathbf{y}_k$ and group n. By combining local hypotheses from all groups in an admissible fashion (such that each detection in $\mathbf{y}_k$ is associated to precisely one group, or classified as clutter) we obtain a *global hypothesis*, described by the vector $\boldsymbol{\lambda}$. The hypothesis probability can be expressed as

$$\Pr\left\{\boldsymbol{\lambda}\middle|\,\mathbf{Y}_k\right\} \propto p\left(\mathbf{y}_k\middle|\,\boldsymbol{\lambda}, \mathbf{Y}_{k-1}\right) \Pr\left\{\boldsymbol{\lambda}\middle|\,\mathbf{Y}_{k-1}\right\}. \tag{54}$$

The likelihood of the data association hypothesis is found through the Gaussian approximation (51) and the clutter model (7) as

$$p\left(\mathbf{y}_k\middle|\,\boldsymbol{\lambda}, \mathbf{Y}_{k-1}\right) = \mathcal{N}\left(\mathbf{y}_k^{\boldsymbol{\lambda}}, \hat{\mathbf{y}}_{k|k-1}^{\boldsymbol{\lambda}}, \mathbf{P}_{\mathbf{yy}}^{\boldsymbol{\lambda}}\right)\left(\frac{1}{V}\right)^{M_k^c}. \tag{55}$$

where

$$\hat{\mathbf{y}}^{\boldsymbol{\lambda}}_{k|k-1} = \left[\left(\hat{\mathbf{g}}_{\boldsymbol{\lambda}(j_1)}\right)^T, \left(\hat{\mathbf{g}}_{\boldsymbol{\lambda}(j_2)}\right)^T, \ldots, \left(\hat{\mathbf{g}}_{\boldsymbol{\lambda}(j_m)}\right)^T \right]^T \tag{56}$$

$$\mathbf{y}^{\boldsymbol{\lambda}}_{k} = \left[\left(\mathbf{y}^{\boldsymbol{\lambda}(j_1)}_{k}\right)^T, \left(\mathbf{y}^{\boldsymbol{\lambda}(j_2)}_{k}\right)^T, \ldots, \left(\mathbf{y}^{\boldsymbol{\lambda}(j_m)}_{k}\right)^T \right]^T \tag{57}$$

for $\{j_1, \ldots, j_m\} = \{j : \boldsymbol{\lambda}(j) \neq 0\}$, and similarly $\mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{yy}}$ is constructed as

$$\mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{yy}} = \begin{bmatrix} \mathbf{P}^{\boldsymbol{\lambda}(j_1)}_{\mathbf{gg}} + \mathbf{W}_k & \ldots & \mathbf{P}^{\boldsymbol{\lambda}(j_1)\boldsymbol{\lambda}(j_m)}_{\mathbf{gg}} \\ \vdots & \ddots & \vdots \\ \mathbf{P}^{\boldsymbol{\lambda}(j_m)\boldsymbol{\lambda}(j_1)}_{\mathbf{gg}} & \ldots & \mathbf{P}^{\boldsymbol{\lambda}(j_m)}_{\mathbf{gg}} + \mathbf{W}_k \end{bmatrix}. \tag{58}$$

The expected signal component of the group n, $\hat{\mathbf{g}}_n$, used in (56) is found in (47). Expressions for the covariance components in (58) are found in (48) for the group covariance, $\mathbf{P}^n_{\mathbf{gg}}$, and in (52) for the cross covariances between group n and m, $\mathbf{P}^{nm}_{\mathbf{gg}}$.

The data association vector, $\boldsymbol{\lambda}$, provides perfect knowledge regarding the number of clutter detections, M^c_k, and the number of detections from group n, $M^t_{k,n}$. Hence, the prior probability for the association vector in (54) can be partitioned as

$$\begin{aligned} \Pr\left\{\boldsymbol{\lambda} \middle| \mathbf{Y}_{k-1}\right\} &= \Pr\left\{\boldsymbol{\lambda}, \mathbf{M}^t_k, M^c_k \middle| \mathbf{Y}_{k-1}\right\} \\ &= \Pr\left\{\boldsymbol{\lambda} \middle| \mathbf{M}^t_k, M^c_k\right\} \Pr\left\{M^c_k\right\} \Pr\left\{\mathbf{M}^t_k \middle| \mathbf{Y}_{k-1}\right\}, \end{aligned} \tag{59}$$

where $\mathbf{M}^t_k = \begin{bmatrix} M^t_{k,1} & \ldots & M^t_{k,N^g_k} \end{bmatrix}^T$ and N^g_k is number of groups. As M^c_k is assumed to be Poisson distributed, we have

$$\Pr\left\{M^c_k\right\} = (\mu V)^{M^c_k} exp(-\mu V)/M^c_k!. \tag{60}$$

Furthermore, $\Pr\left\{\boldsymbol{\lambda} \middle| \mathbf{M}^t_k, M^c_k\right\}$ is found using combinatorics,

$$\Pr\left\{\boldsymbol{\lambda} \middle| \mathbf{M}^t_k, M^c_k\right\} = \prod_{n=1}^{M^g_k} \begin{pmatrix} M_k - \sum_{m=1}^{n-1} M^t_{k,m} \\ M^t_{k,n} \end{pmatrix}^{-1}. \tag{61}$$

Finally, the probability of the total number of target detections,

$$\Pr\left\{\mathbf{M}^t_k \middle| \mathbf{Y}_{k-1}\right\} = \prod_{n=1}^{N^g_k} \Pr\left\{M^t_{k,n} \middle| \mathbf{Y}_{k-1}\right\} \tag{62}$$

where $\Pr\left\{M^t_{k,n} \middle| \mathbf{Y}_{k-1}\right\}$ is approximated using estimated expected signal amplitude, $\hat{\boldsymbol{\sigma}}_{k|k-1}$ in (30).

4.4 Measurement update

To perform the measurement update defined by (34) - (35), we first need to construct the included entities. The predicted mean and covariance are already given in Section 4.1 and $\hat{\mathbf{y}}^{\boldsymbol{\lambda}}_{k|k-1}$ and $\mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{yy}}$ are given by (56) and (58), respectively. However, the cross covariance $\mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{zy}}$ needs to be estimated.

Using the sigma point sets in (41) and (42), $\mathbf{P}_{\mathbf{zr}} = \mathrm{Cov}\left\{\mathbf{z}_k, \mathbf{r}_k \middle| \mathbf{Y}_{k-1}\right\}$ can be estimated as

$$\mathbf{P}_{\mathbf{zr}} \approx \sum_i u^i_k \left(\mathbf{Z}^i_k - \hat{\mathbf{z}}_{k|k-1}\right)\left(\mathbf{R}^i_k - \hat{\mathbf{r}}_{k|k-1}\right)^T. \tag{63}$$

From (63), $\mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{zy}}$ is calculated in three steps. First, approximating the relation between reflectors and clusters as in (21) we find

$$\mathbf{P}^{cc,n}_{\mathbf{zc}_l} = \sum_i^N \bar{w}_{l_i} \mathbf{P}^{l_i}_{\mathbf{zr}}, \tag{64}$$

where l_i lists all reflectors in cluster l in cluster constellation cc. Using (64) and (28), it is easy to find

$$\mathbf{P}^n_{\mathbf{zg}} = \frac{1}{N^{cc}_n} \sum_{cc=1}^{N^{cc}_n} \sum_{l=1}^{L^{cc}_n} q^{cc}_{n,l} \mathbf{P}^{cc,n}_{\mathbf{zc}_l}, \tag{65}$$

from which we can finally construct the sought cross covariance, $\mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{zy}}$, as

$$\mathbf{P}^{\boldsymbol{\lambda}}_{\mathbf{zy}} = \left[\mathbf{P}^{\boldsymbol{\lambda}(j_1)}_{\mathbf{zg}}, \ldots, \mathbf{P}^{\boldsymbol{\lambda}(j_m)}_{\mathbf{zg}}\right], \tag{66}$$

for $\{j_1, \ldots, j_m\} = \{j : \boldsymbol{\lambda}(j) \neq 0\}$.

The posterior mean and covariance estimates under data association hypothesis, $\boldsymbol{\lambda}$, are found through inserting (56), (66) and (58) into (34)-(35). The posterior density (33) is a weighted sum of the posterior densities for all $\boldsymbol{\lambda}$ weighted by (54). The mean and covariance of a Gaussian mixture model are readily calculated though moment matching, see e.g. [2]. This concludes the derivation of the filter framework for estimating the position of vehicles using possible unresolved radar detections.

5 Evaluation

In this section we compare the proposed extended target model, denoted $\mathcal{M}_1$, with that of a point source model (basic model), $\mathcal{M}_2$, similar to those presently used in the automotive industry. The evaluation is performed in two steps, First, we compare the ability the models to explain radar observations. Second, we compare the estimation error, $\mathbf{e}^i_k = \left\|\mathrm{E}\left\{\mathbf{z}^i_k | \mathbf{Y}_k, \mathcal{M}_n\right\} - \mathbf{z}^i_k\right\|_2$, of the tracking

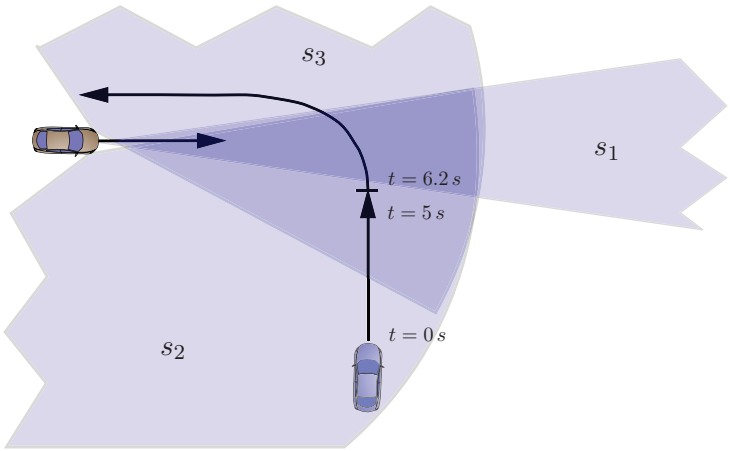


Figure 6: Host vehicle equipped with three radar sensors, one mechanically scanned 77 Ghz long range radar, denoted s_1, and two medium range 24 GHz radars looking to the right and left, denoted s_2 and s_3, respectively. In the evaluated tracking scenario the host vehicle is traveling at constant speed at a straight path. The target vehicle drives at a crossing path stopping in front of the host vehicle before making a left turn.

system derived in Section 4 with one based on $\mathcal{M}_2$. The evaluation is limited to single target vehicle scenarios, i.e. $\mathbf{z}_k = \left[\left(\mathbf{z}_k^h\right)^T, \left(\mathbf{z}_k^1\right)^T\right]^T$.

For both evaluations, radar observations are collected from three sensors, one long-range radar at 77 *GHz* (denoted s_1) and two medium-range radars at 24 *GHz* (denoted s_2 and s_3), mounted on the host vehicle as illustrated in Fig. 6. Sensor s_1 has an update rate of 10 *Hz*, a field of view of 16^o and a detection range of approx. 150 m, whereas s_2 and s_3 cover a 150^o field of view up to approx. 70 m using 13 independent receive beams, each delivering detections every $40ms$. The resolution cell for the two types of sensors are, $\mathbf{\Delta}_d^{s_1} = [2m,\ .5m/s,\ 3.5^o]$ and $\mathbf{\Delta}_d^{s_{2,3}} = [2m,\ 6m/s,\ \infty]$, respectively, where $\mathbf{\Delta}_d^{s_{2,3}}$ is used to describe the resolution in each of the receive beams of s_2 and s_3. The corresponding measurement noise covariance for a point target is specified as $\mathbf{W}_k^{s_1} = \text{diag}\left(\left[.4,\ 2,\ \frac{1\pi}{180}\right]\right)^2$ and $\mathbf{W}_k^{s_{2,3}} = \text{diag}\left(\left[.4,\ .5,\ \frac{1.2\pi}{180}\right]\right)^2$. Target vehicle reference position, $\mathbf{z}_k^i$, and host vehicle position measurements are acquired using accurate DGPS measurements. We proceed by introducing the point source model, then explain the two comparisons and their results respectively.

5.1 Point source model

To evaluate the tracking performance gained in terms of estimation error by considering the vehicles as extended objects, we compare our tracking system with one based on a point source model. To make the comparison as fair as possible, we use the same state parametrization and both models exploit knowledge regarding

the physical dimension of the observed vehicle. Given $\mathbf{z}_k^1$ the model compensates for offset errors by positioning the expected target measurement, $\hat{\mathbf{y}}_{\mathcal{M}_2}^1$, on the intersection between the line of sight between the radar sensor and $\left(\zeta_{x,k}^1, \zeta_{x,k}^1\right)$ and the vehicle frame. Using this model we design a probabilistic data association filter (PDAF) [28], where only one measurement may originate from the target and the presence of multiple measurements are modelled as clutter described by a homogeneous Poisson process.

5.2 Sensor model comparison

The ability to explain a set of given observations can be compared by evaluating the *log-likelihood ratio*

$$\ell(\mathbf{y}_k, \mathbf{z}_k) = \log\left(\frac{p\left(\mathbf{y}_k \middle| \mathbf{z}_k, \mathcal{M}_1\right)}{p\left(\mathbf{y}_k \middle| \mathbf{z}_k, \mathcal{M}_2\right)}\right) \tag{67}$$

where the model specific likelihood functions can be partitioned as

$$p\left(\mathbf{y}_k \middle| \mathbf{z}_k, \mathcal{M}_i\right) = \sum_{\boldsymbol{\lambda}_k \in \boldsymbol{\mathcal{L}}_k} p\left(\mathbf{y}_k \middle| \boldsymbol{\lambda}_k, \mathbf{z}_k, \mathcal{M}_i\right) P\left\{\boldsymbol{\lambda}_k \middle| \mathbf{z}_k, \mathcal{M}_i\right\}. \tag{68}$$

In the following sections, we present expressions for $p\left(\mathbf{y}_k \middle| \boldsymbol{\lambda}_k, \mathbf{z}_k, \mathcal{M}_i\right)$ and $P\{\boldsymbol{\lambda}_k \middle| \mathbf{z}_k, \mathcal{M}_i\}$ for the different models as well as the evaluation of (67) for radar measurements from two types of radar sensors.

Likelihood function

The likelihood ratio test is commonly used to compare hypotheses, in this case which model is more likely to have produced the radar measurements. The test is reasonable if the number of tuning parameters are the same for the compared models. For our proposed sensor model, $\mathcal{M}_1$, (68) is given by (55) and (59), with the minor difference that we do not have to integrate over $\mathbf{z}_k$. The corresponding densities for $\mathcal{M}_2$ are obtained similarly as

$$p\left(\mathbf{y}_k \middle| \boldsymbol{\lambda}_k, \mathbf{z}_k, \mathcal{M}_2\right) = \begin{cases} \mathcal{N}(\mathbf{y}_k^j; \hat{\mathbf{y}}_{\mathcal{M}_2}^1, \mathbf{W}_k)(\frac{1}{V})^{M_k^c} & : M_k^t = 1 \\ (\frac{1}{V})^{M_k^c} & : M_k^t = 0 \end{cases}$$

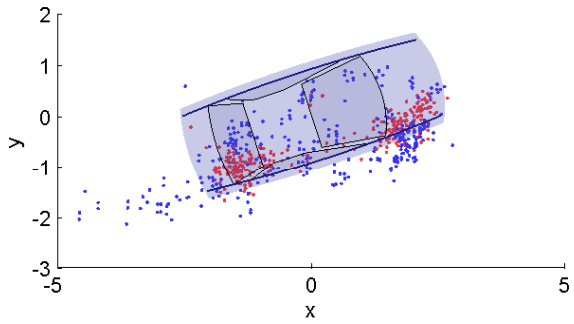
$$\Pr\left\{\boldsymbol{\lambda}_k \middle| \mathbf{z}_k, \mathcal{M}_2\right\} = \begin{cases} \frac{P_D}{M_k} \frac{e^{-\mu V}(\mu V)^{M_k^c}}{M_k^c!} & : M_k^t = 1 \\ (1 - P_D)\frac{e^{-\mu V}(\mu V)^{M_k^c}}{M_k^c!} & : M_k^t = 0. \end{cases}$$

The probability of detection is modelled in the same way for both models and the clutter intensity, μ, is estimated from data using nonparametric PDA [29]. Note that the same parameters are used for tuning both models, i.e., the measurement noise covariance and the probability of detection. All other parameters are taken directly from the sensor specification.

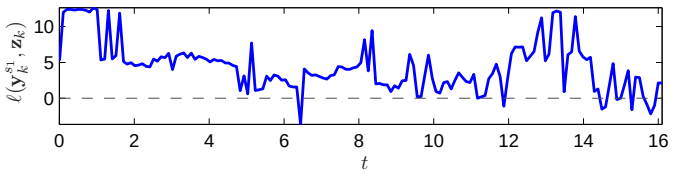
Results

The log-likelihood ratio (67) is evaluated using radar measurements from two sensors of different types, s_1 and s_2. The data is collected while the host vehicle drives straight towards the target vehicle at an angle of 18^o (offset from the side of the vehicle) starting at a distance of $40m$. A scatter plot of the collected data is illustrated in Fig. 7a.

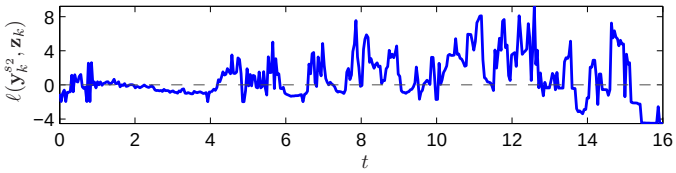
Figures 7b and 7c display (67) evaluated for measurements delivered by sensor s_1 and s_2, respectively. Both figures show a clear advantage in favour of our proposed model. This is especially clear for sensor s_1 which has higher resolution than sensor s_2 and where we often receive multiple detections neatly concentrated to the front and rear wheel housings, see Fig. 7a. For sensor s_2, the detections are spread along the side of the target vehicle (some even positioned outside the vehicle frame). This behavior is also modelled in $\mathcal{M}_1$ but the advantage is not as dominant as in the case of sensor s_1.



(a) Scatter plot of the evaluated target measurements from the s_1 (red) and s_2 (blue) radars.



(b) Log-likelihood ratio s_1



(c) Log-likelihood ratio s_2

Figure 7: Log-likelihood ratio comparison between our model, $\mathcal{M}_1$, and the simpler point source model, $\mathcal{M}_2$.

5.3 Tracking filter comparison

The tracking filters based on $\mathcal{M}_1$ and $\mathcal{M}_2$ are evaluated using data from s_1, s_2 and s_3 in the scenario depicted in Fig. 6. This particular scenario is chosen, both because it is a relevant scenario for active safety systems addressing intersection accidents [30] and because it is challenging for a radar based tracking system.

In the evaluation, both filters are initiated in the reference state, $\mathbf{z}_0^1$, provided by the DGPS system with initial covariance $\mathbf{P}_0 = \text{diag}([1,\ 1,\ \frac{3\pi}{180},\ .5,\ .001,\ 1.5])^2$ and use the same process noise parameters, $\sigma_{\dot{v}}^2 = 9$ and $\sigma_{\dot{c}}^2 = \frac{1}{70}$. The filter implementation for our proposed model is given in Section 4 and for the point source model we employ a standard UKF.

The result of the comparison is shown in Fig. 8 in terms of absolute longitudinal and lateral positioning error, e_x and e_y, in the reference vehicle coordinate frame as well as absolute velocity error, e_v, and heading angle error, e_ψ, in the global coordinate frame. The result indicates a clear advantage for $\mathcal{M}_1$ in terms of accurate and stable positioning of the target vehicle, as well as velocity and heading estimates. Worth noting is the later part of the scenario, $t \in [7,\ 10]s$, where the two vehicles are close and many of the features on the reference vehicle are resolved. In this part of the scenario the target vehicle starts to turn, something that confuses the simpler point source model while our model still manages to position the vehicle well. Additionally, the jump in the heading estimation error at $t \approx 9.8s$ is explained by the filters only receiving measurements from the rear part of the vehicle for some updates. As measurements again appear from the front of the vehicle, it is clear that our proposed model is able to take more advantage of the new information to deduce the heading of the vehicle more accurately than the point source model.

6 Conclusion

In this paper we have proposed an accurate and tractable radar sensor model capable of describing both multiple detections from a vehicle and their relation to the limited sensor resolution. Furthermore, we have developed a framework for tracking vehicles based on this model. The evaluation of the sensor model shows that our model is clearly better than the reference model at describing the vehicle radar detections from the two evaluated sensors. Additionally, the evaluation of the tracking performance indicates substantial benefits using our model compared to the reference model.

The reference model is notably simpler than the proposed one, and clearly penalized in the evaluation when multiple features are resolved. However, it is presently used in many systems and our comparison show that a widely used family of tracking frameworks can be adapted to incorporate the new measurement model, with improved performance as a result.

By formally including sensor resolution in the model, it can be used for a wide range of sensors and targets by changing appropriate parameters according to the

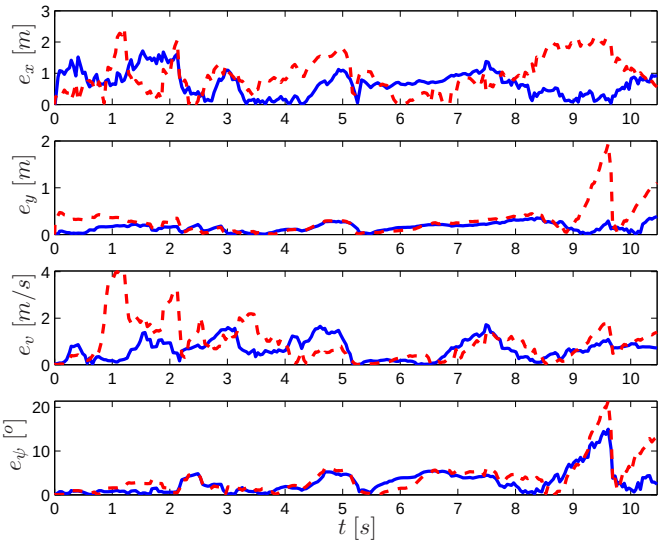


Figure 8: Comparison of the absolute estimation error from our tracking framework (solid blue) and the point source model (dashed red) for longitudinal and lateral positioning error in the reference vehicle coordinate frame, e_x and e_y, as well as absolute velocity error, e_v, and heading angle error, e_ψ, in the global coordinate frame.

sensor specification. This feature is of most importance to the automotive industry as it allows for sensors to be more easily replaced or updated.

Appendix A: Reflector mapping

The reflector mapping is divided into two parts; first the reflector is positioned in the observation space, second, the the expected signal amplitude is modelled.

Appendix A.1: Reflector position

Assuming that point reflector with index i is positioned at $\mathbf{x}_i = (x_i, y_i)$ in a local coordinate system with the origin in the center of target vehicle j. The global position of the reflector is then given by

$$\begin{bmatrix} \zeta_x^{j,i} \\ \zeta_y^{j,i} \end{bmatrix} = \begin{bmatrix} \zeta_{x_k}^{j} \\ \zeta_{y_k}^{j} \end{bmatrix} + \mathbf{R}(\psi_k^j) \begin{bmatrix} x_i \\ y_i \end{bmatrix} \tag{69}$$

where $\mathbf{R}(\cdot)$ is a 2x2 rotational matrix. Similarly, assuming that sensor, s, is mounted on the host vehicle at $\mathbf{x}_s = (x_s, y_s)$ and with an angle of ψ_s, the global

position of the sensor is defined as

$$\begin{bmatrix} \zeta_x^s \\ \zeta_y^s \end{bmatrix} = \begin{bmatrix} \zeta_{x_k}^h \\ \zeta_{y_k}^h \end{bmatrix} + \mathbf{R}(\psi_k^h) \begin{bmatrix} x_s \\ y_s \end{bmatrix} \tag{70}$$

Additionally, we define the relative angle between reflector i and sensor s according to

$$\alpha_{i,s} = \arctan\left(\frac{\zeta_y^{j,i} - \zeta_y^s}{\zeta_x^{j,i} - \zeta_x^s}\right) \tag{71}$$

Using these relations, the mapping $\mathbf{z}_k \xrightarrow{\mathcal{R}_i(\cdot)} \mathbf{r}_i$ is defined as

$$r_i = \sqrt{(\zeta_x^{j,i} - \zeta_x^s)^2 + (\zeta_y^{j,i} - \zeta_y^s)^2} \tag{72}$$

$$\begin{aligned} \dot{r}_i = &\left(v_k^j \cos(\psi_k^l - \alpha_{i,s}) + v_\perp^j \cos\left(\frac{\pi}{2} + \arg(\mathbf{x}_i) - \alpha_{i,s}\right)\right) \\ &- \left(v_k^h \cos(\psi_k^h - \alpha_{i,s}) + v_\perp^h \cos\left(\frac{\pi}{2} + \arg(\mathbf{x}_s) - \alpha_{i,s}\right)\right) \end{aligned} \tag{73}$$

$$\phi_i = \alpha_{i,s} - \psi_k^h - \psi_s \tag{74}$$

where $v_\perp^j = v_k^j c_k^j \|\mathbf{x}_i\|$ and $v_\perp^h = v_k^h c_k^h \|\mathbf{x}_s\|$ are the velocity component due to rotation of the target and host vehicle, respectively.

Appendix A.2: Signal amplitude

The signal power of the sensor is characterized by two functions, the reciprocal antenna gain pattern, $A_a(\phi)$, and the signal attenuation, $A_r(r)$. Associated with each reflector is a visibility function, $\nu_\sigma^i(\alpha_{i,s}, \psi_k^j)$ dependent on the relative angle between the observing sensor and the reflector and the heading of the target vehicle. Using these models, the expected return amplitude of reflector i is calculated as,

$$\sigma_i = A_a(\phi_i) A_r(r_i) \nu_\sigma^i(\alpha_{i,s}, \psi_k^j). \tag{75}$$

Appendix B: Gaussian cluster density approximation

In this section we derive the gaussian approximation of the cluster density in (21) and (22). Let a cluster i consist of N reflectors positioned at $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_N$ in measurement space. According to (18) the signal component of the cluster is given by

$$\mathbf{c}_i = \sum_{n=1}^{N} w_n \, \mathbf{r}_n,$$

where in this case the reflector positions, $\mathbf{r}_n$, are known whereas the weights, w_n, are stochastic. The weights are expressed in terms of the received signal amplitudes of each reflector, A_n,

$$w_n = \frac{A_n}{\sum_{m=1}^{N} A_m}.$$

where

$$A_n \sim \text{Rayleigh}(\sigma_n).$$

To find a Gaussian approximation of cluster density we need to find the first two moments of $\mathbf{c}_i$.

Appendix B.1: Mean approximation

The mean of $\mathbf{c}_i$ is

$$\bar{\mathbf{c}}_i = \mathrm{E}\{\mathbf{c}_i\} = \sum_{n=1}^{N} \bar{w}_n \mathbf{r}_n. \tag{76}$$

where $\bar{w}_n = \mathrm{E}\{w_n\}$ is not trivial to express. However, monte-Carlo simulations indicate that the approximation

$$\bar{w}_n \approx \frac{\bar{A}_n}{\sum_{m=1}^{N} \bar{A}_m} \tag{77}$$

where

$$\bar{A}_n = \mathrm{E}\{A_n\} = \sigma_n \sqrt{\frac{\pi}{2}}, \tag{78}$$

yields a reasonable approximation of (76).

Appendix B.2: Covariance approximation

The remaining difficulty is to approximate the covariance matrix of $\mathbf{c}_i$,

$$\mathbf{C}_i = E\left\{ (\mathbf{c}_i - \bar{\mathbf{c}}_i)(\mathbf{c}_i - \bar{\mathbf{c}}_i)^T \right\} \tag{79}$$

The first aim is to find a representation which more robust to approximations. For notation, set $\Delta\mathbf{r}_n = \mathbf{r}_n - \bar{\mathbf{c}}_i$ and $\Delta w_n = w_n - \bar{w}_n$. In the following, we will use the relations

$$\sum_{n=1}^{N} w_n = 1, \tag{80}$$

which follows from the definition of w_n, and

$$\sum_{n=1}^{N} \bar{w}_n \Delta\mathbf{r}_n = 0, \tag{81}$$

which is clear due to $E\{\mathbf{r} - \bar{\mathbf{r}}\} = 0$. These relations and notations yield

$$\begin{aligned}
\mathbf{C}_i &= E\left\{ \left(\sum_{n=1}^{N} w_n(\mathbf{r}_n - \bar{\mathbf{r}}) \right) \left(\sum_{n=1}^{N} w_n(\mathbf{r}_n - \bar{\mathbf{r}}) \right)^T \right\} \\
&= E\left\{ \left(\sum_{n=1}^{N} w_n \Delta\mathbf{r}_n \right) \left(\sum_{n=1}^{N} w_n \Delta\mathbf{r}_n \right)^T \right\} \\
&= E\left\{ \left(\sum_{n=1}^{N} \Delta w_n \Delta\mathbf{r}_n \right) \left(\sum_{n=1}^{N} \Delta w_n \Delta\mathbf{r}_n \right)^T \right\} \\
&= \sum_{n=1}^{N} \sum_{m=1}^{N} \Delta\mathbf{r}_n \Delta\mathbf{r}_m^T E\{\Delta w_n \Delta w_m\}.
\end{aligned} \tag{82}$$

Recall that the matrices $\Delta\mathbf{r}_n \Delta\mathbf{r}_m^T$ are known for all indices n and m. Hence, we will now strive to find approximations for the scalar factors

$$\mathrm{Cov}\{w_n, w_m\} = E\{\Delta w_n \Delta w_m\}. \tag{83}$$

To this end, we use the Taylor approximation

$$\begin{aligned}
w_n &= \frac{A_n}{S_i} \\
&\approx \frac{\bar{A}_n}{\bar{S}_i} + \frac{A_n - \bar{A}_n}{\bar{S}_i} - \frac{(S_i - \bar{S}_i)\bar{A}_n}{\bar{S}_i^2} \\
&= \frac{\bar{A}_n}{\bar{S}_i} + \frac{A_n}{\bar{S}_i} - \frac{S_i \bar{A}_n}{\bar{S}_i^2}
\end{aligned} \tag{84}$$

where $S_i = \sum_{m=1}^{N} A_m$ and

$$\bar{A}_n = E\{A_n\} \tag{85}$$

$$\bar{S}_i = E\left\{ \sum_{m=1}^{N} A_m \right\}. \tag{86}$$

Thus, we get

$$\begin{aligned}
\mathrm{Cov}\{w_n, w_m\} &\approx E\left\{ \left(\frac{A_n}{\bar{S}_i} - \frac{S_i \bar{A}_n}{\bar{S}_i^2} \right) \left(\frac{A_m}{\bar{S}_i} - \frac{S_i \bar{A}_m}{\bar{S}_i^2} \right) \right\} \\
&= \frac{E\{A_n A_m\}}{\bar{S}_i^2} - \frac{E\{A_n S_i\}\bar{A}_m}{\bar{S}_i^3} \\
&\quad - \frac{E\{A_m S_i\}\bar{A}_n}{\bar{S}_i^3} + \frac{E\{S_i^2\}\bar{A}_n \bar{A}_m}{\bar{S}_i^4}.
\end{aligned} \tag{87}$$

To evaluate (87), we essentially only need the relations

$$E\{A_n\} = \sigma_n \sqrt{\pi/2} \tag{88}$$

$$\bar{A_n^2} = E\left\{A_n^2\right\} = 2\sigma_n^2 \tag{89}$$

$$E\{A_n\, A_m\} = \begin{cases} E\left\{A_n^2\right\} & \text{if } n = m \\ E\{A_n\}\, E\{A_m\} & \text{otherwise.} \end{cases} \tag{90}$$

References

[1] W. Koch, "Advanced sensor models: Benefits for target tracking and sensor data fusion," in *Multisensor Fusion and Integration for Intelligent Systems, 2006 IEEE International Conference on*, sep 2006, pp. 565–570.

[2] S. Blackman and R. Popoli, *Design and analysis of modern tracking systems*. Norwood, MA: Artech House, 1999.

[3] Y. Bar-Shalom and W. D. Blair, *Multitarget-Multisensor Tracking Volume III: Applications and Advances*. Norwood, MA: Artech House, 2000.

[4] R. Ostrovityanov and F. Basalov, *Statistical Theory of Extended Radar Targets*. Artech House, 1985.

[5] M. Waxman and O. Drummond, "A bibliography of cluster (group) tracking," in *Signal and Data Processing of Small Targets 2004. Edited by Drummond, Oliver E. Proceedings of the SPIE*, vol. 5428, 2004, pp. 551–560.

[6] W. Koch and R. Saul, "A Bayesian approach to extended object tracking and tracking of loosely structured target groups," in *Information Fusion, 2005 8th International Conference on*, vol. 1, 2005.

[7] W. Koch, "On bayesian tracking of extended objects," in *Multisensor Fusion and Integration for Intelligent Systems, 2006 IEEE International Conference on*, Sept. 2006, pp. 209–216.

[8] K. Gilholm, S. J. Godsill, S. Maskell, and D. J. Salmond, "Poisson models for extended target and group tracking," *SPIE*, October 2005.

[9] D. F. Crouse, S. Member, M. Guerriero, and P. Willett, "A Critical Look at the PMHT," *Journal of Advances in Infor- mation Fusion*, 2009.

[10] M. Wieneke and W. Koch, "Probabilistic tracking of multiple extended targets using random matrices," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, ser. Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference, vol. 7698, Apr. 2010.

[11] M. Bühren and B. Yang, "Simulation of automotive radar target lists using a novel approach of object representation," in *IEEE Intelligent Vehicles Symposium (IV)*, Tokyo, Japan, June 2006.

[12] W. Koch and G. Van Keuk, "Multiple hypothesis track maintenance with possibly unresolved measurements," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 33, no. 3, pp. 883–892, July 1997.

[13] H. Blom and E. Bloem, "Bayesian tracking of two possibly unresolved maneuvering targets," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 43, no. 2, pp. 612–627, April 2007.

[14] ——, "Hybrid SIR joint particle filtering under limited sensor resolution," in *Information Fusion, 2007 10th International Conference on*, 2007, pp. 1–8.

[15] S. Jeong and J. Tugnait, "Tracking of two targets in clutter with possibly unresolved measurements," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 44, no. 2, pp. 748–765, April 2008.

[16] K.-C. Chang and Y. Bar-Shalom, "Joint probabilistic data association for multitarget tracking with possibly unresolved measurements and maneuvers," *Automatic Control, IEEE Transactions on*, vol. 29, no. 7, pp. 585–594, Jul 1984.

[17] L. Ng and Y. Bar-Shalom, "Model of unresolvable measurements for multitarget tracking," in *Preceedings of the OCEANS 1981 conference*, Sept 1981, pp. 982–987.

[18] S. Julier and J. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, 2004.

[19] T. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Sonar tracking of multiple targets using joint probabilistic data association," *IEEE Journal of Oceanic Engineering*, vol. 8, July 1983.

[20] J. Gunnarsson, L. Svensson, L. Danielsson, and F. Bengtsson, "Tracking vehicles using radar detections," in *Intelligent Vehicles Symposium, 2007. Proceedings IEEE*, Istanbul, June 2007.

[21] F. E. Daum and R. J. Fitzgerald, "Importance of resolution in multiple-target tracking," *Signal and Data Processing of Small Targets 1994*, vol. 2235, no. 1, pp. 329–338, 1994. [Online]. Available: http://link.aip.org/link/?PSI/2235/329/1

[22] M. Arulampalam, N. Gordon, M. Orton, and B. Ristic, "A variable structure multiple model particle filter for gmti tracking," *Proceedings of the Fifth International Conference on Information Fusion. FUSION 2002. (IEEE Cat.No.02EX5997)*, vol. vol.2, pp. 927 – 34, 2002.

[23] P. Swerling, "Probability of detection for fluctuating targets," *Information Theory, IRE Transactions on*, vol. 6, no. 2, pp. 269–308, 1960.

[24] J. Dunn and D. Howard, "Radar target amplitude, angle, and Doppler scintillation from analysis of the echo signal propagating in space," *IEEE Transactions on Microwave Theory and Techniques*, vol. 16, no. 9, pp. 715–728, 1968.

[25] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman filter: particle filters for tracking applications*. Norwood, MA: Artech House, 2004.

[26] D. Reid, "An algorithm for tracking multiple targets," *Automatic Control, IEEE Transactions on*, vol. 24, no. 6, pp. 843–854, Dec 1979.

[27] R. Kalman, "A new approach to linear filtering and prediction problems," *Trans. ASME, Ser. D. Journal Basic Eng.*, vol. 82, pp. 24–45, 1960.

[28] Y. Bar-Shalom and E. Tse, "Tracking in a cluttered environment with probabilistic data association," *Automatica*, vol. 11, September 1975.

[29] Y. Bar-Shalom and T. Fortmann, *Tracking and data association*. Academic-Press, Boston, 1988.

[30] M. Brännström, E. Coelingh, and J. Sjöberg, "Threat assessment for avoiding collisions with turning vehicles," in *Intelligent Vehicles Symposium, 2009 IEEE*, June 2009, pp. 663–668.

Paper IV

A new vehicle motion model for improved predictions and situation assessment

J. Sörstedt, L. Svensson, F. Sandblom, and L. Hammarstrand

Accepted for publication in
IEEE Transactions on Intelligent Transportation Systems

A new vehicle motion model for improved predictions and situation assessment

Joakim Sörstedt, Lennart Svensson,
Fredrik Sandblom, and Lars Hammarstrand

Abstract

Reliable and accurate vehicle motion models are of vital importance for automotive active safety systems, for a number of reasons. First of all, these models are necessary in tracking algorithms which provide the safety system with information. Second, the motion model is often used by the safety application to make long term predictions of the future traffic situation. These predictions are then part of the basic data used by the system to determine if, when and how to intervene. In this paper we suggest a framework for designing accurate vehicle motion models. The resulting models differ from conventional ones in that the expected control input from the driver is included. By also providing a methodology for a formal treatment of the uncertainties, a model structure well suited, e.g., in a tracking algorithm is obtained. To utilize the framework in an application will require careful design and validation of sub-models for calculating expected driver control input. We illustrate the potential of the framework, by examining the performance for a specific model example using real measurements. The properties are compared to those of a constant acceleration model. Evaluations indicate that the proposed model yields better predictions and that it has an ability to estimate the prediction uncertainties.

1 Introduction

A current trend in today's automotive industry is to equip vehicles with more and more active safety systems. These have the objective of aiding the driver in different accident prone situations, such as unintentional lane departures or dangers caused by distracted drivers. Based on sensor data, active safety systems try to assess the situation and detect dangerous scenarios. Some examples of active safety systems, which are currently available on the market are given in [1]. To improve these systems in the future, detailed vehicle motion models will be most valuable. The reasons for this are at least threefold:

1. The motion model is an integral part of the tracking system which provides the safety system with information regarding the surrounding environment. More precise motion models would improve the tracking system's ability to handle complex traffic scenarios.

2. An active safety system needs to decide when and how to intervene. In theory, an accurate motion model enables the safety applications to make better long term predictions, and thereby allow decisions to be made earlier and with greater confidence, [2].

3. Careful modelling of specific situations of interest, e.g., related to different driver maneuvers, can help the safety application to classify and understand the traffic situation.

The motion models predominantly used in tracking systems are derived from the fundamental laws of dynamics and assume that the input is approximately white Gaussian noise. Such models have been employed extensively in numerous applications, and often with satisfactory results, [3], [4], [5]. However, there are reasons to believe that traditional models can be improved to better meet the demands of future safety systems. For instance, more accurate long term predictions would be most useful for a decision making algorithm.

A main weakness in traditional models is that they do not capture the influence the driver has on the vehicle motion. Under normal conditions, it is reasonable to believe that the vehicle is completely controlled by the driver. Attempts to model driver behaviour have appeared previously in the literature. In [6], a driver behaviour model is used to design an adaptive cruise controller and in [7], similar ideas have been used for threat assessment. Other examples of driver behavior models have been developed for micro traffic simulators [8], and in the ongoing EU-projects ISI-PADAS [9] and ITERATE [10]. A closely related problem exists in the field of robotics where algorithms are designed to find a collision free path through a partly unknown environment, see, e.g., the textbook [11] and the references therein. We believe that neither of these models fulfill the requirements from a future active safety system. For instance, the models in [6], [11] are not suitable from a tracking perspective as they do not provide a stochastic description of the vehicle motion. Similarly, the idea in [7] is not well suited for decision making as it does not capture the variations in driving styles and driving preferences among different drivers. Finally, since the models for micro process simulators are designed for simulation purposes they lack a description of uncertainties in predictions as well as the ability to adapt model parameters to observed data. These properties are essential for filtering and decision making purposes and make these models inappropriate alternatives in their current form. The same arguments also hold for the models so far presented in the ISI-PADAS and ITERATE projects.

In this paper, we continue to develop the model framework introduced in [12], [13]. The idea is that the driver controls the vehicle by making a trade off between different objectives such as the desire to travel fast and to travel comfortably. To describe the expected driver control signal, we use a cost function which reflects the driver objectives. By minimizing this cost function the most likely control signal can be calculated. Compared to [12], [13], the model framework is here extended to cover a more generic model structure. More importantly, it is also adapted to handle uncertainties in the cost function parameters. This allows us to design

models which not only contain knowledge regarding typical driver behaviour, but which are also able to capture variations among different drivers and to adapt as the driver behavior changes.

The main contribution of this paper is the model structure described in Section 3. This structure suggests a method for including the expected driver input in a model structure, and provides guidelines on how to construct cost functions. In contrast to other commonly used methods, our approach has the advantage that model parameters have a clear interpretation. Having said that, a significant amount of work still remains before this approach can be used by an application. First of all, the cost function needs to be adjusted and validated for the intended application. Secondly, an efficient implementation for minimizing the cost function has to be developed. To illustrate the performance of the suggested model structure, we present one cost function and evaluate it using real measurements. In the studied examples, the proposed model explains the data better and provides better predictions, compared to a constant acceleration (CA) model.

The paper is organized as follows. Section 2 presents the notation and some of the most commonly used motion models in tracking applications. In Section 3, a general description of the proposed modelling framework is provided. A specific design of the cost function is described in Section 4 and later evaluated on measured data in Section 5. Finally, Section 6 contains the conclusions.

2 Background

The vehicle motion model describes the evolution of a time-discrete vector $\mathbf{x}_k \in \mathcal{R}^{n_x}$, referred to as the *state vector* of the model. For vectors and matrices (indicated by boldface) we use the sub-index k as notation for a discrete time instant with a continuous counterpart in t_k, whereas the notation $x(k)$ is used for scalars. The continuous time interval between two *samples* is constant and denoted $T_s = t_k - t_{k-1}$. We consider a state vector partitioned as

$$\mathbf{x}_k = [(\mathbf{x}_k^h)^T \ (\mathbf{z}_k^t)^T \ \mathbf{r}_k^T]^T, \tag{1}$$

where $\mathbf{x}_k^h$ is the state vector for the host vehicle, $\mathbf{z}_k^t$ is the state vector for the nearest target vehicle in front of the host vehicle and the vector $\mathbf{r}_k$ contains parameters describing the road. Our main interest in this article is to derive a model for the vector $\mathbf{x}_k^h$. For $\mathbf{z}_k^t$ and $\mathbf{r}_k$ we use standard models which are described in this section. The model we propose for the host vehicle can also be used for one or several target vehicles, but for simplicity we apply it only on the host vehicle.

We restrict the scope to Markovian models on the form

$$\mathbf{x}_k = \mathbf{f}_{k-1}(\mathbf{x}_{k-1}, \mathbf{e}_{k-1}), \tag{2}$$

where $\mathbf{f}_{k-1}(\cdot)$ is a possibly time-varying and nonlinear function of $\mathbf{x}_{k-1}$ and a stochastic noise vector $\mathbf{e}_{k-1} \in \mathcal{R}^{n_e}$. The noise process accounts for model uncertainties, and is traditionally modelled as an independent and identically distributed (i.i.d.) process with zero mean.

An important subgroup of the general model structure (2) is the family of linear motion models defined by the matrices $\mathbf{A}_{k-1} \in \mathcal{R}^{n_x \times n_x}$ and $\mathbf{B}_{k-1} \in \mathcal{R}^{n_x \times n_e}$ such as

$$\mathbf{x}_k = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{B}_{k-1}\mathbf{e}_{k-1}. \tag{3}$$

These motion models are popular in tracking applications because they enable simple and efficient algorithms such as the Kalman filter [14], for estimating and predicting the states.

A frequently used linear motion model, is the CA model. In this model the state vector $\mathbf{z}_k^t$ is parameterized as

$$\mathbf{z}_k^t = [\eta_x^t(k)\ \eta_y^t(k)\ \dot{\eta}_x^t(k)\ \dot{\eta}_y^t(k)\ \ddot{\eta}_x^t(k)\ \ddot{\eta}_y^t(k)]^T, \tag{4}$$

where $(\eta_x^t(k), \eta_y^t(k))$ is the center position of the vehicle and $(\dot{\eta}_x^t(k), \dot{\eta}_y^t(k))$ and $(\ddot{\eta}_x^t(k), \ddot{\eta}_y^t(k))$ are the first and second order time derivatives of $\eta_x^t(k)$ and $\eta_y^t(k)$. In the version of the CA model employed here, the noise process $\mathbf{e}_k$ determines the *jerk* between sample instances. The acceleration, velocity and position increments are thus given by $\mathbf{e}_k T_s$, $\mathbf{e}_k T_s^2/2$ and $\mathbf{e}_k T_s^3/6$, respectively. The system matrices then take the form

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & T_s & 0 & \frac{T_s^2}{2} & 0 \\ 0 & 1 & 0 & T_s & 0 & \frac{T_s^2}{2} \\ 0 & 0 & 1 & 0 & T_s & 0 \\ 0 & 0 & 0 & 1 & 0 & T_s \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \ \mathbf{B} = \begin{bmatrix} \frac{T_s^3}{6} & 0 \\ 0 & \frac{T_s^3}{6} \\ \frac{T_s^2}{2} & 0 \\ 0 & \frac{T_s^2}{2} \\ T_s & 0 \\ 0 & T_s \end{bmatrix}, \tag{5}$$

where the noise process is modelled as $\mathbf{e}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_e)$, where $\mathbf{C}_e = \text{diag}[\sigma^2_{\ddot{\eta}_x}\ \sigma^2_{\ddot{\eta}_y}]$.

In essence, the CA model is a kinematic particle model [15], where the object motion is decoupled in the different dimensions. Other examples, where the object's motion in the different dimensions are instead coupled, are the coordinated turn (CT) models [3], [4]. More detailed vehicle motion models, which for instance take wheel slip and wheel angle into account, have been derived for applications such as vehicle stability and traction control. The textbook [16] provides more information on dynamic vehicle models, and describes the *single track model* or *bicycle model* which has been used for vehicle tracking, at least in a simplified form, see, e.g., the study [17].

In this paper, we use a curved *road coordinate system* such that the position of the host vehicle (η_x^h, η_y^h) and target vehicle (η_x^t, η_y^t) are given relative to the road. This coordinate system, previously used, e.g., in [18], is defined in Fig. 1. The same figure also contains a Cartesian coordinate system (ξ_x, ξ_y), which will be needed later in Section 4. To find a relation between these coordinate systems, a model of the road curvature is required. Here we follow suggestions from e.g. [19] and use a clothoid model. The road curvature is then described by a local linear

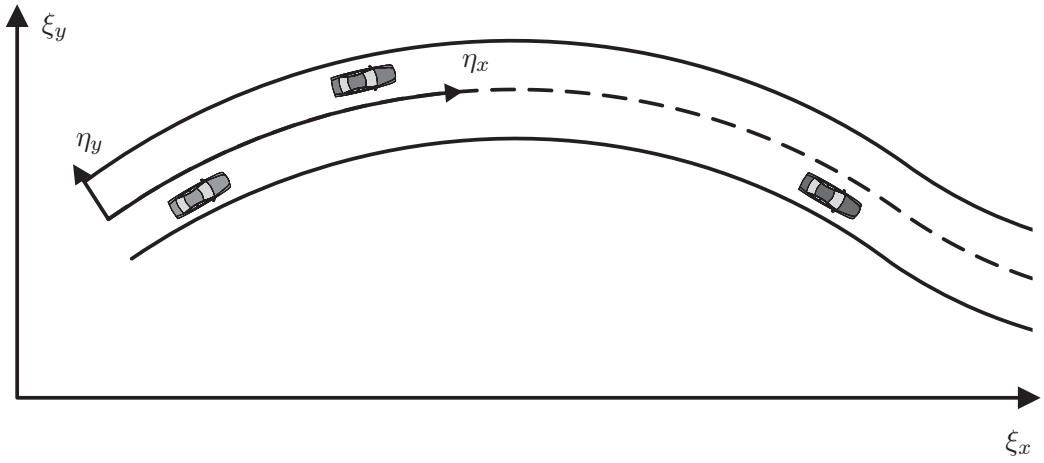


Figure 1: This figure illustrates the global Cartesian coordinate system (ξ_x, ξ_y), and the curved road coordinate system (η_x, η_y).

approximation about the host vehicle, i.e.,

$$c(\eta_x^h + \Delta\eta_x) = c_0 + \Delta\eta_x c_1. \tag{6}$$

The parameter c_0 is the local curvature at the center of the road and c_1 the curvature change rate as a function of distance $\Delta\eta_x$ from the host vehicle. We include the clothoid parameters $c_0(k)$, $c_1(k)$ and the lane width $L_w(k)$ in the road state vector

$$\mathbf{r}_k = [L_w(k)\ c_0(k)\ c_1(k)]^T. \tag{7}$$

The evolution of $\mathbf{r}_k$ is modelled as

$$\mathbf{r}_k = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & T_s\dot{\eta}_x(k-1) \\ 0 & 0 & 1 \end{bmatrix} \mathbf{r}_{k-1} + \mathbf{v}_{k-1}^r, \tag{8}$$

where $\mathbf{v}_{k-1}^r \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_\mathbf{r})$.

3 Motion model framework

In this section, we describe how the motion model is extended to incorporate the effect of the driver. We base the modelling idea on a set of postulates which are highlighted in the text. The equations for calculating the driver input are also presented and important model properties are discussed.

3.1 Model structure

We will here motivate and present the structure and some key components in the proposed model.

Postulate 1. *The driver controls the motion of the vehicle by steering and adjusting the acceleration.*

As the classical motion models fail to acknowledge the influence of the driver, there is, at least theoretically, room for substantial improvements if we can model the driver actions. Naturally, the vehicle obeys the laws of physics and the models in Section 2 are still appropriate even though the input is no longer white noise. In a general parameterization we suggest a model

$$\mathbf{x}_k^h = \mathbf{f}_{k-1}(\mathbf{x}_{k-1}^h, \mathbf{u}_{k-1}(\mathbf{x}_{k-1}), \mathbf{v}_{k-1}), \tag{9}$$

where $\mathbf{u}_{k-1}(\mathbf{x}_{k-1})$ is the driver input signal and $\mathbf{v}_{k-1}$ is a white noise process. Modelling the driver input $\mathbf{u}_{k-1}(x_{k-1})$ is a key component here and the topic of Sections 3.2 and 4.

The complete state vector for the host vehicle is given by $\mathbf{x}_k^h$, but the kinematic state of the host vehicle is described by the vector $\mathbf{z}_k^h$ which is a subvector of $\mathbf{x}_k^h$. The propagation of this kinematic state vector is described by the model

$$\mathbf{z}_k^h = \mathbf{f}_{k-1}^h(\mathbf{z}_{k-1}^h, \mathbf{u}_{k-1}(\mathbf{x}_{k-1}) + \mathbf{v}_{k-1}^z). \tag{10}$$

Here we use a modified CA model

$$\mathbf{z}_k^h = \mathbf{A}\mathbf{z}_{k-1}^h + \mathbf{B}(\mathbf{u}_{k-1}(\mathbf{x}_{k-1}) + \mathbf{v}_{k-1}^z), \tag{11}$$

where

$$\mathbf{u}_{k-1}(\mathbf{x}_{k-1}) = \begin{bmatrix} \dddot{\eta}_x^h(k) \\ \dddot{\eta}_y^h(k) \end{bmatrix} \tag{12}$$

and $\mathbf{v}_{k-1}^z \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_\mathbf{z})$. There are other model structures, like the bicycle model, which often describe vehicle motion more accurately than the chosen CA-model. However, as we shall see, the CA-model leads to a relatively simple implementation.

There is reason to believe that the variations in behavior between different drivers can be large. We therefore allow the function $\mathbf{u}_{k-1}(\mathbf{x}_{k-1})$ to be parameterized by a vector $\boldsymbol{\theta}_{k-1}$, which is intended to capture variations in driver preferences and driving styles. For an example of what parameters $\boldsymbol{\theta}_{k-1}$ may contain see Section 4.

The evolution of $\boldsymbol{\theta}_k$ is described by the model

$$\boldsymbol{\theta}_k = \mathbf{g}_{k-1}(\boldsymbol{\theta}_{k-1}, \mathbf{v}_{k-1}^\theta), \tag{13}$$

i.e., it is assumed to be independent of all other elements in $\mathbf{x}_{k-1}^h$. We use

$$\mathbf{g}_{k-1}(\boldsymbol{\theta}_{k-1}, \mathbf{v}_{k-1}^\theta) = \gamma\boldsymbol{\theta}_{k-1} + (1-\gamma)\boldsymbol{\mu} + \mathbf{v}_{k-1}^\theta, \tag{14}$$

where $\mathbf{v}_{k-1}^\theta \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_\theta)$ while $\gamma \in [0, 1)$, $\boldsymbol{\mu}$ and $\mathbf{Q}_\theta$ are design parameters. The proposed process model is such that when an element in $\boldsymbol{\theta}_{k-1}$ is not observable,

its posterior distribution approaches a prior distribution representing the set of all drivers. To illustrate the convergence of the first two moments we study

$$E[\boldsymbol{\theta}_{k+l}] = \gamma^l E[\boldsymbol{\theta}_k] + \boldsymbol{\mu}(1-\gamma^l) \tag{15}$$

$$Cov[\boldsymbol{\theta}_{k+l}] = \gamma^{2l} Cov[\boldsymbol{\theta}_k] + \frac{1-\gamma^{2l}}{1-\gamma^2}\mathbf{Q}_\theta. \tag{16}$$

Thus, γ determines convergence speed, whereas $\boldsymbol{\mu}$ and $\mathbf{Q}_\theta/(1-\gamma^2)$ are the limit values of the first two moments as the prediction length $l \to \infty$.

The task of modelling driver behaviour in terms of $\mathbf{u}_{k-1}(\mathbf{x}_{k-1})$ and $\mathbf{v}_{k-1}$ is an extensive and difficult problem. To simplify the problem we assume that it is divided into subproblems for which the uncertainties are smaller and $\mathbf{v}_{k-1}$ has a unimodal distribution.

> **Postulate 2.** *In normal traffic, the actions of a driver can be divided into a set of different categories.*

Each category typically corresponds to a driver intention such as: *changing lanes*, *following the lane* or *turning left*, but it could also capture other aspects of the situation such as if the driver is distracted or not. Given the intention of the driver, it is easier to capture the driver behavior. We use a scalar parameter $m(k)$ to indicate the driver action which is currently active.

The complete state vector for the host vehicle is now given by

$$\mathbf{x}_k^h = [(\mathbf{z}_k^h)^T \ \boldsymbol{\theta}_k^T \ m(k)]^T. \tag{17}$$

Note that the function $\mathbf{u}_{k-1}(\mathbf{x}_{k-1})$ depends on both $\boldsymbol{\theta}_k$ and $m(k)$ and the implications of this are further discussed in Section 3.3.

3.2 Calculating the driver input $\mathbf{u}_k(\mathbf{x}_k)$

In the suggested framework, the driver operates the vehicle such that three main objectives, or driving rules, are fulfilled.

> **Postulate 3.** *The driver strives to control the car in a safe and comfortable manner, and at a preferred velocity.*

The safety preference corresponds to the driver's desire to control the vehicle such that it stays in the preferred lane and at a safe distance to other vehicles and objects. Distance is here both a spatial and temporal measure. Simultaneously, the driver asks for a comfortable journey and is reluctant to be put under large accelerating forces and jerks. The third objective captures the driver's desire to maintain a preferred velocity.

Based on this assumption, we can represent the driver preferences mathematically using a cost function

$$\text{cost}(\mathbf{u}_{k-1}, \mathbf{x}_{k-1}). \tag{18}$$

Suggestions for cost functions that penalize trajectories which are in conflict with the above preferences are given in Section 4. As we will see, the cost function is implicitly dependent on the vector $\boldsymbol{\theta}_{k-1}$.

An important driver ability is to make decisions based on both the present and upcoming situation.

> **Postulate 4.** *The driver plans ahead and tries to find the optimal route, such that the driver preferences are considered for the near - and not only the immediate - future.*

We wish to incorporate this property in our model, such that the expected driver input at time $k-1$ enables a desirable journey also for the near future. The cost function is therefore evaluated for a trajectory of driver input signals and corresponding state vectors. By selecting the trajectory with the lowest cost, we can produce a prediction of the vehicle motion, that incorporates knowledge regarding the road, other objects, speed limits, etc.

In a sense, we are hereby approximating the driver by an optimal controller [20], where the optimality criterion is defined by the cost function. The optimal driver input is obtained as $\mathbf{u}_{k-1}(\mathbf{x}_{k-1}) \triangleq$

$$\underset{\mathbf{u}_{k-1}}{\operatorname{argmin}} \quad \min_{\{\mathbf{u}_k,\ \dots\ ,\mathbf{u}_{k+N-2}\}} \quad \sum_{n=k-1}^{k+N-2} \operatorname{cost}(\mathbf{u}_n, \mathbf{x}_n), \tag{19}$$

where the future state vectors $\mathbf{x}_n,\ n > k-1$, are calculated using the model (9). The driver input, $\mathbf{u}_{k-1}$, is usually a nonlinear function of the state $\mathbf{x}_{k-1}$ which turns the modified CA model in (11) into a nonlinear motion model. Here we assume that the driver completely controls the car such that there are no uncertainties in the motion model; we set $\mathbf{v}_n = \mathbf{0}$ for $n = k-1, \dots, k+N-3$.

To solve (19), we take an approach which is very common in optimal control and which is advocated, e.g., in Betts [21]. Instead of parameterizing the problem in the free variables $\mathbf{u}_{k-1}, \dots, \mathbf{u}_{k+N-2}$, we solve for the optimal sequence of state vectors $\mathbf{z}_k, \dots, \mathbf{z}_{k+N-2}$. The motion model (11) (with $\mathbf{v}^z_{n-1} = \mathbf{0}$ for $n = k, \dots, k+N-2$) now enters as linear constraints on the state sequence. As a consequence, the optimization problem becomes more high dimensional, but with much nicer properties. In this alternative form, the optimization can be solved conveniently using a built in routine in TOMLAB (`http://tomopt.com/tomlab/`) called SNOPT. SNOPT solves the optimization problem using a efficient sequential quadratic programming method [22]. Note, however, that depending on the choice of cost function, the optimization problem (19) may be of different complexity. The properties of the optimization problem hence needs to be considered when designing the cost functions. For the cost functions presented in Section 4, which are designed so that their first and second derivatives both exist and are continuous, and for the test scenario described in Section 5, SNOPT computes the optimal trajectory in less than 50 ms on a conventional computer.

3.3 Model features

The idea of approximating the driver as an optimal controller according to an objective function has been discussed previously, e.g., in [6]. An important difference in the new motion model is the inclusion of the parameters $\boldsymbol{\theta}_k$ and $m(k)$. In this section, we discuss the vital role these parameters have for situation assessment and to produce reliable predictions.

Driver preferences, $\boldsymbol{\theta}_k$

The purpose of the parameter $\boldsymbol{\theta}_k$ is to capture the individual driving style, and allow the driver preferences to vary over time. Naturally, additional parameters add complexity to the problem, but the benefits obtained by introducing the vector $\boldsymbol{\theta}_k$ are arguably more significant.

First of all, as time goes by the system gains information regarding $\boldsymbol{\theta}_k$ by tracking the vehicle trajectory $\mathbf{z}_1^h$, $\mathbf{z}_2^h, \ldots, \mathbf{z}_k^h$. Due to improved knowledge about $\boldsymbol{\theta}_k$, predictions become more accurate and biases are reduced. The objective is, of course, to attain this performance for (virtually) all driving styles.

Second, by describing the uncertainties in $\boldsymbol{\theta}_k$ we can obtain a parameterized and improved description of the uncertainties in $\mathbf{z}_k^h$. While the cost function framework provides an estimate of the expected control signal, the random vector $\boldsymbol{\theta}_k$ affects also the variance of $\mathbf{u}_{k-1}(x_{k-1})$. This is a key advantage of the model framework, which owes to the fact that the uncertainties in $\boldsymbol{\theta}_k$ can be propagated to $\mathbf{z}_k^h$ according to (10). In situations where the optimal trajectory is essentially independent of $\boldsymbol{\theta}_k$, i.e., when most drivers would act the same, the predictions are more reliable. On the other hand, in other circumstances the optimal trajectory may be highly dependent on the driving style, which would lead to a lower confidence in the predictions as $\boldsymbol{\theta}_k$ is not fully known. As we shall see in Section 5, the variations in prediction uncertainties coincide fairly well with the observed prediction errors.

Driver intention, $m(k)$

It may seem like a drawback to include the hypotheses parameter, $m(k)$, into the state vector, as it leads to several difficulties. First, to provide a complete description of the model in (9) one must design a process model for $m(k)$, given by the probability function $P(m(k)|\mathbf{x}_{k-1})$. A second complication is that a predetermined list of possible hypotheses is also needed; of course, depending on the position of the vehicles, not all of these hypotheses may be likely. Nonetheless, for many applications, it is also potentially the greatest advantage with the framework since it enables straightforward and formal derivations of the posterior distribution of $m(k)$, as demonstrated in [13]. In particular, once the intention specific motion models have been developed, information regarding $m(k)$ is extracted simply by examining the past trajectories of the vehicles. Thereby, one may obtain a powerful tool to draw conclusions regarding $m(k)$ and thus, the traffic situation.

However, it is beyond the scope of this article to fully develop and illustrate cost functions for multiple intentions. Thus, even though we would like to point out the potential gains with including $m(k)$ it is not properly investigated here. The state vector used in the remainder of this article is therefore reduced to $\mathbf{x}_k^h = [(\mathbf{z}_k^h)^T \boldsymbol{\theta}_k^T]^T$.

4 Designing cost functions

The cost functions should be designed to capture the typical behavior of the driver. We have partly based our choices on behavior studies used to design roads [23] and common practise for driving taught at driving schools. As was argued in Postulate 2, the actions of the driver can be separated into different categories. Each of these will typically require different cost functions. In this paper we focus on the hypothesis that the driver intends to follow the right lane of the road. Designing cost functions to describe the normal driver behavior is different compared to most previous contributions which have been focused on predicting dangerous situations [24], [25], [6].

We divide the cost function into four different components related to the longitudinal velocity, $c_{lo}(\mathbf{u}_n, \mathbf{x}_n)$, the lateral positioning of the vehicle, $c_{la}(\mathbf{u}_n, \mathbf{x}_n)$, the comfort of the trajectory, $c_c(\mathbf{u}_n, \mathbf{x}_n)$, and the interaction with other vehicles, $c_{in}^1(\mathbf{u}_n, \mathbf{x}_n)$ and $c_{in}^2(\mathbf{u}_n, \mathbf{x}_n)$. Among these, $c_{la}(\mathbf{u}_n, \mathbf{x}_n)$, $c_{in}^1(\mathbf{u}_n, \mathbf{x}_n)$ and $c_{in}^2(\mathbf{u}_n, \mathbf{x}_n)$ cover the safety aspect discussed in Postulate 3. Similarly, $c_c(\mathbf{u}_n, \mathbf{x}_n)$ ensures that the trajectory is comfortable and $c_{lo}(\mathbf{u}_n, \mathbf{x}_n)$ that the preferred speed of the driver is maintained. A weighted sum of the different components constitutes the total cost

$$\text{cost}(\mathbf{u}_n, \mathbf{x}_n) = \alpha_{lo} c_{lo}(\mathbf{u}_n, \mathbf{x}_n) + \alpha_{la} c_{la}(\mathbf{u}_n, \mathbf{x}_n) + \alpha_c c_c(\mathbf{u}_n, \mathbf{x}_n) + \alpha_{in} c_{in}(\mathbf{u}_n, \mathbf{x}_n) \tag{20}$$

where the parameters $\alpha_{lo}, \alpha_{la}, \alpha_c$ and α_{in} decide how the driver gives priority to the different costs.

As mentioned in previous sections, driving preferences, and thus driving style, are characterized by the vector $\boldsymbol{\theta}_k$. In our implementation, $\boldsymbol{\theta}_k$ determines α_{lo} and α_{in}, but also the parameters $\dot{\eta}_{\text{x-ref}}^h(k)$ and $\bar{t}_h(k)$ related, respectively, to $c_{lo}(\mathbf{u}_n, \mathbf{x}_n)$ and $c_{in}(\mathbf{u}_n, \mathbf{x}_n)$ defined below. We will return to the parameterization of $\boldsymbol{\theta}_k$ and its process model in Section 5.2. The remaining weight parameters are assumed constant and are given the values $\alpha_{la} = \alpha_c = 1$.

4.1 Longitudinal cost function, $c_{lo}(\cdot, \cdot)$

The longitudinal cost function reflects the driver's desire to maintain a specific speed, given by the parameter $\dot{\eta}_{\text{x-ref}}^h$. To punish trajectories for which the speed of the host vehicle deviates from the reference speed, we use

$$c_{lo}(\mathbf{u}_n, \mathbf{x}_n) = (\dot{\eta}_x^h(n) - \dot{\eta}_{\text{x-ref}}^h(n))^2. \tag{21}$$

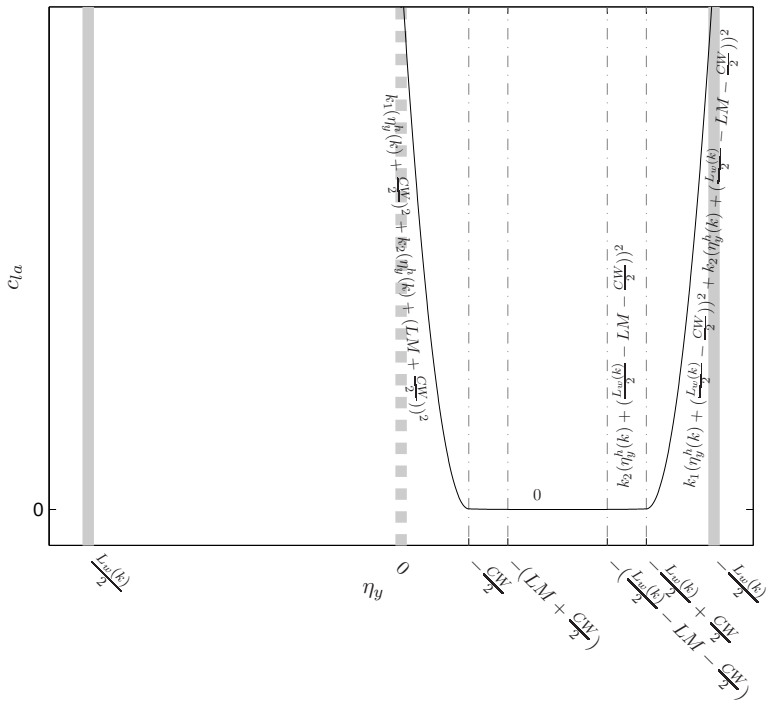


Figure 2: Lateral cost function for following the right lane. The figure specifies the mathematical expressions for the different road regions.

If this cost function is omitted, the optimal driver tends to slow down as that leads to a safe and comfortable journey. Like the other parameters in $\boldsymbol{\theta}_k$, $\dot{\eta}^h_{\text{x-ref}}(k)$ is partially unknown initially. However, $\dot{\eta}^h_{\text{x-ref}}(k)$ is special since it sometimes varies significantly over time, e.g., when the vehicle exits or enters a highway.

4.2 Lateral cost function, $c_{la}(\cdot, \cdot)$

The objective with the lateral cost function is to associate large costs to trajectories outside the driver's preferred lane; a cost related to the desire to drive safely. Fig. 2 defines how $c_{la}(\mathbf{u}_n, \mathbf{x}_n)$ depends on the lateral position η_y. The background in Figure 2 consists of a road, portrayed with thick grey lines divided into different cost segments. In the expressions, CW is the width of the car and LM is a margin to the lane edge, whereas k_1 and k_2 are design parameters used to tune the cost function. We use $k_1 = 1000$ and $k_2 = 3$.

4.3 Comfort cost function, $c_c(\cdot, \cdot)$

The comfort cost corresponds to the driver's desire to steer the vehicle in a smooth and comfortable manner. Smooth trajectories are accomplished by increasing the cost for large accelerations and jerks. However, the acceleration perceived by the

driver is not equal to the acceleration of the vehicle expressed in curved road coordinates. Instead, the global Cartesian coordinate system (ξ_x, ξ_y) is used. It is easy to construct a nonlinear mapping $T : \mathbb{R}^2 \to \mathbb{R}^2$ from road coordinates to Cartesian coordinates, i.e., T is such that $T(\eta_x, \eta_y) = [\xi_x \ \xi_y]^T$. Using the mapping T it is also possible to compute $(\ddot{\xi}^h_x(n), \ddot{\xi}^h_y(n))$ and $(\dddot{\xi}^h_x(n), \dddot{\xi}^h_y(n))$ from $\mathbf{z}^h_n$, $\mathbf{r}_n$ and $(\dddot{\eta}^h_x(n), \dddot{\eta}^h_y(n))$.

To simplify the expressions we introduce the notation $\ddot{\xi}^h(n) = \sqrt{\ddot{\xi}^h_x(n)^2 + \ddot{\xi}^h_y(n)^2}$ and $\dddot{\xi}^h(n) = \sqrt{\dddot{\xi}^h_x(n)^2 + \dddot{\xi}^h_y(n)^2}$. The cost function is divided into two parts:

$$c_{\mathrm{c}}(\dddot{\eta}_x(n), \dddot{\eta}^h_y(n), \mathbf{z}^h_n, \mathbf{r}_n) = c_{\mathrm{a}}(\ddot{\xi}(n)) + c_{\mathrm{j}}(\dddot{\xi}(n)), \tag{22}$$

where $c_{\mathrm{a}}(\cdot)$ is related to the acceleration, and $c_{\mathrm{j}}(\cdot)$ is related to the jerk.

Empirical studies indicate that people are fairly insensitive to jerks and accelerations up to a certain level, above which the motions feel more unpleasant (possibly with the exception of accelerating forces). We use functions $c_{\mathrm{a}}(\cdot)$ and $c_{\mathrm{j}}(\cdot)$ of the form

$$c(x) = \begin{cases} \frac{x^2}{g^2} & \text{when } x < g, \\ \left(\frac{5}{6} + \frac{x^2}{6g^2}\right)^6 & \text{otherwise.} \end{cases} \tag{23}$$

As a consequence, we get cost functions which increase more rapidly for $x > g$ and which have continuous derivatives. Based on information from [23] we set $g = 2$ for $c_{\mathrm{a}}(\cdot)$ and $g = 1.5$ for $c_{\mathrm{j}}(\cdot)$.

4.4 Cost functions for vehicle interaction, $c_{in}(\cdot, \cdot)$

An important aspect in the proposed motion model is that it not only incorporates the interaction between the vehicle and the road, e.g., in $c_{la}(\mathbf{u}_n, \mathbf{x}_n)$, but also the interaction with other vehicles. Here we define a cost function which seeks to penalize trajectories where vehicles are too close. We limit our design in this article to situations when there is one more vehicle close to the host vehicle. The vehicle interaction cost contains two parts

$$c_{in}(\mathbf{u}_n, \mathbf{x}_n) = c^1_{in}(\mathbf{u}_n, \mathbf{x}_n) + c^2_{in}(\mathbf{u}_n, \mathbf{x}_n), \tag{24}$$

which concern the temporal and spatial distances to the other vehicle, respectively.

To measure the temporal distance we introduce the *time headway*

$$t_h(k) = \frac{\eta^t_x(k) - \eta^h_x(k)}{\dot{\eta}^h_x(k)}, \tag{25}$$

which reflects the time it takes for the host vehicle to reach the current position of the target vehicle. A typical driver is reluctant to allow the time headway to

become too small, whereas it matters less if t_h is, say 4 seconds or 10 seconds. The cost function

$$c_{int}^1(\mathbf{u}_n, \mathbf{x}_n) = \begin{cases} \frac{(\bar{t}_h - t_h(k))^2}{t_h(k)} & \text{if } 0 < t_h(k) \leq \bar{t}_h \\ 0 & \text{otherwise,} \end{cases} \tag{26}$$

is selected to enable our model to capture this behavior, and $\bar{t}_h$ is included in the state vector $\boldsymbol{\theta}_k$.

The spatial distance is here defined as the distance between the longitudinal positions of the two vehicles

$$d(k) = \eta_x^t(k) - \eta_x^h(k). \tag{27}$$

In this case, we design the cost function for a scenario where the target vehicle is initially positioned ahead of the host vehicle, on a single lane road. A reasonable cost function for the spatial distance should be large when $d(k)$ is small (and even larger when $d(k)$ is negative). We use the cost function

$$c_{int}^2(\mathbf{u}_n, \mathbf{x}_n) = \begin{cases} 0 & \text{if } \overline{d} \leq d \\ \frac{(\overline{d}-d)^2}{(\overline{d}-\underline{d})^2} & \text{if } \underline{d} \leq d \leq \overline{d} \\ \left(\frac{5}{6} + \frac{(\overline{d}-d)^2}{6(\overline{d}-\underline{d})^2}\right)^6 & \text{if } d \leq \underline{d}. \end{cases} \tag{28}$$

Note the resemblance to (23), if we set $x = \overline{d} - d$ and $g = \overline{d} - \underline{d}$. The parameter values used in our evaluations are $\overline{d} = 5$ and $\underline{d} = 2$.

5 Evaluation results

To evaluate our model we study how well it explains (predicts) two different driving sequences. The sequences are real traffic situations, collected using the same driver on straight, busy, two-lane roads at two different places in a city. The host vehicle is positioned in the middle of the road and is traveling behind a target vehicle, see Fig. 3. The two vehicles are closely spaced, such that the host vehicle has to slow down when the target vehicle slows down, see Fig. 4 and 5.

The host vehicle is equipped with a 77 GHz radar sensor, which measures the range and range rate to the target vehicle. Based on these and internal measurements on the host vehicle speed and acceleration, we estimate the trajectories $\mathbf{Z}_L^h = \left[\mathbf{z}_1^h, \ldots, \mathbf{z}_L^h\right]$ and $\mathbf{Z}_L^t = \left[\mathbf{z}_1^t, \ldots, \mathbf{z}_L^t\right]$ using a Kalman smoother. As benchmark we use a CA model also for the host vehicle. For notation, we introduce

$$\begin{aligned} \mathcal{M}_1 &: \text{ constant acceleration model} \\ \mathcal{M}_2 &: \text{ proposed motion model,} \end{aligned}$$

to indicate the choice of model.

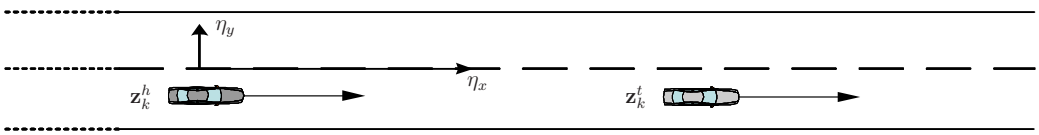


Figure 3: *Test scenario:* Host vehicle, described by a state vector $\mathbf{z}_k^h$, travels on a straight road behind a second vehicle, described by $\mathbf{z}_k^t$.

5.1 Evaluation criteria

We will mainly study two different properties of the model, where the first concerns the prediction capability and the variations in prediction uncertainties. More specifically, we will compare the mean and the variance of the density[1] $p(\mathbf{z}_k^h|\mathbf{Z}_{k-1}^h, \mathbf{Z}_{k-1}^t, \mathcal{M}_i)$, for $i = 1$ and 2, to the observed trajectories.

The second property is the model likelihood

$$p(\mathbf{Z}_L^h, \mathbf{Z}_L^t|\mathcal{M}_i) = p(\mathbf{Z}_L^h|\mathbf{Z}_L^t, \mathcal{M}_i)p(\mathbf{Z}_L^t|\mathcal{M}_i), \tag{29}$$

which is a standard criterion in model testing, see [2]. As we are mainly interested in the ratio between the two likelihoods, our evaluations below study $p(\mathbf{Z}_L^h|\mathbf{Z}_L^t, \mathcal{M}_i)$, since $p(\mathbf{Z}_L^t|\mathcal{M}_i)$ does not depend on $\mathcal{M}_i$.

5.2 Filtering $\boldsymbol{\theta}_k$

In this section we describe how $\boldsymbol{\theta}_k$ is treated in the evaluation.

Motivation

We notice that both criteria involve the computation of densities of the host state vector, either $p(\mathbf{z}_k^h|\mathbf{Z}_{k-1}^h, \mathbf{Z}_{k-1}^t, \mathcal{M}_i)$ or $p(\mathbf{Z}_L^h|\mathbf{Z}_L^t, \mathcal{M}_i)$. For the CA model, these densities are completely determined by the process noise covariance matrix $\mathbf{C}_e = \text{diag}[\sigma^2_{\dot{\eta}_x} \; \sigma^2_{\dot{\eta}_y}]$. In the evaluations, we select $\sigma^2_{\dot{\eta}_x}$ and $\sigma^2_{\dot{\eta}_y}$ in order to maximize $p(\mathbf{Z}_L^h|\mathbf{Z}_L^t, \mathcal{M}_1)$.

For the proposed motion model, these densities are, in principal, evaluated in the same way. We will describe the evaluation steps for the density $p(\mathbf{Z}_L^h|\mathbf{Z}_L^t, \mathcal{M}_2)$ first. Later, at the end of Section 5.2, we explain how the algorithm can be adapted to instead compute $p(\mathbf{z}_k^h|\mathbf{Z}_{k-1}^h, \mathbf{Z}_{k-1}^t, \mathcal{M}_2)$. To simplify the calculations, we decompose the likelihood as

$$p(\mathbf{Z}_L^h|\mathbf{Z}_L^t, \mathcal{M}_2) = \prod_{k=1}^{L} p(\mathbf{z}_k^h|\mathbf{Z}_{k-1}^h, \mathbf{Z}_L^t, \mathcal{M}_2), \tag{30}$$

[1] In this section, all densities are conditioned on the initial states, $\mathbf{z}_0^h$ and $\mathbf{z}_0^t$, even though these variables are omitted for brevity.

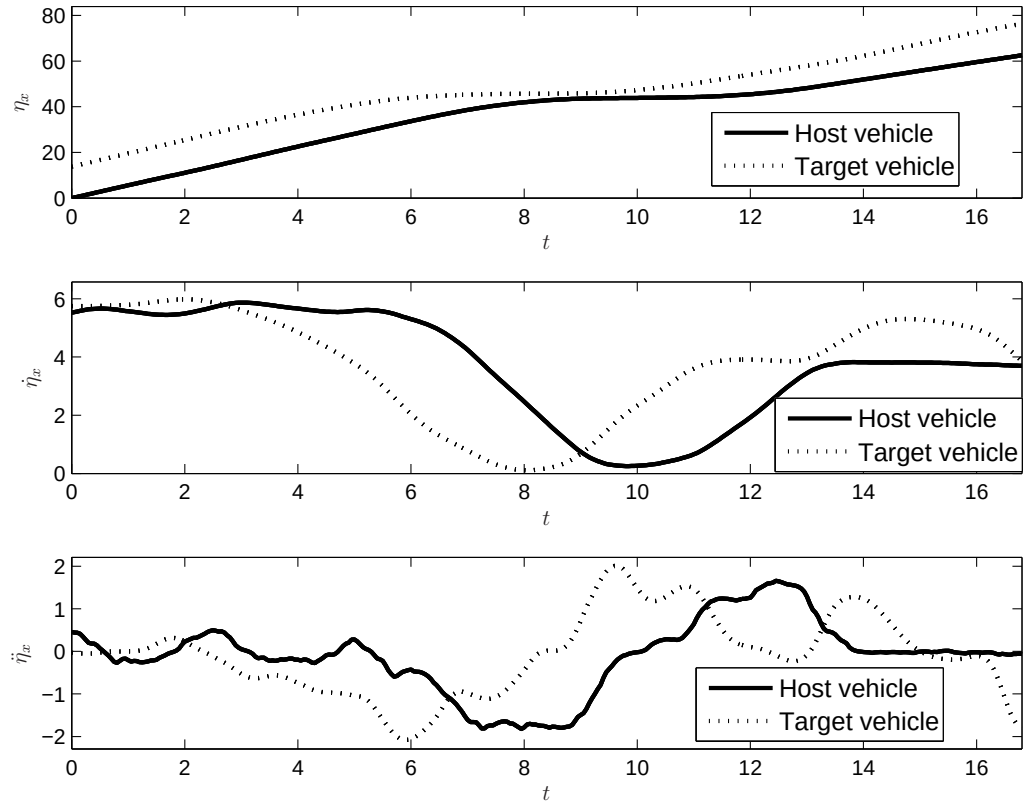


Figure 4: An illustration of the positions, velocities and accelerations of the two vehicles, in the first sequence, as functions of time.

and our primary goal is therefore to find $p(\mathbf{z}_k^h|\mathbf{Z}_{k-1}^h, \mathbf{Z}_L^t, \mathcal{M}_2)$. Note that this expression is not conditioned on $\boldsymbol{\theta}_k$. By marginalizing the dependence on $\boldsymbol{\theta}_k$, the variance of $\boldsymbol{\theta}_k$ will influence the uncertainties in the predictions of $\mathbf{z}_k^h$.

To evaluate

$$p(\mathbf{z}_k^h|\mathbf{Z}_{k-1}^h, \mathbf{Z}_L^t, \mathcal{M}_2) = \int p(\mathbf{z}_k^h|\boldsymbol{\theta}_{k-1}, \mathbf{Z}_{k-1}^h, \mathbf{Z}_L^t, \mathcal{M}_2) p(\boldsymbol{\theta}_{k-1}|\mathbf{Z}_{k-1}^h, \mathbf{Z}_L^t, \mathcal{M}_2)\, d\boldsymbol{\theta}_{k-1}, \tag{31}$$

we approximate $p(\boldsymbol{\theta}_{k-1}|\mathbf{Z}_{k-1}^h, \mathbf{Z}_L^t, \mathcal{M}_2)$ as a Gaussian density and then apply the Unscented Transform [26] to obtain a Gaussian approximation of (31). In Section 5.2 we describe how to obtain $p(\boldsymbol{\theta}_{k-1}|\mathbf{Z}_{k-1}^h, \mathbf{Z}_L^t, \mathcal{M}_2)$ using the Unscented Kalman Filter (UKF), but first we describe $\boldsymbol{\theta}_k$ and its model in more detail.

Parameterization of $\boldsymbol{\theta}_k$

The time evolution of $\boldsymbol{\theta}_k$ was described already in (13) and (14). We have previously mentioned that $\boldsymbol{\theta}_k$ defines α_{lo}, α_{in}, $\dot{\eta}_{\text{x-ref}}^h(k)$ and $\bar{t}_h(k)$, and we now wish

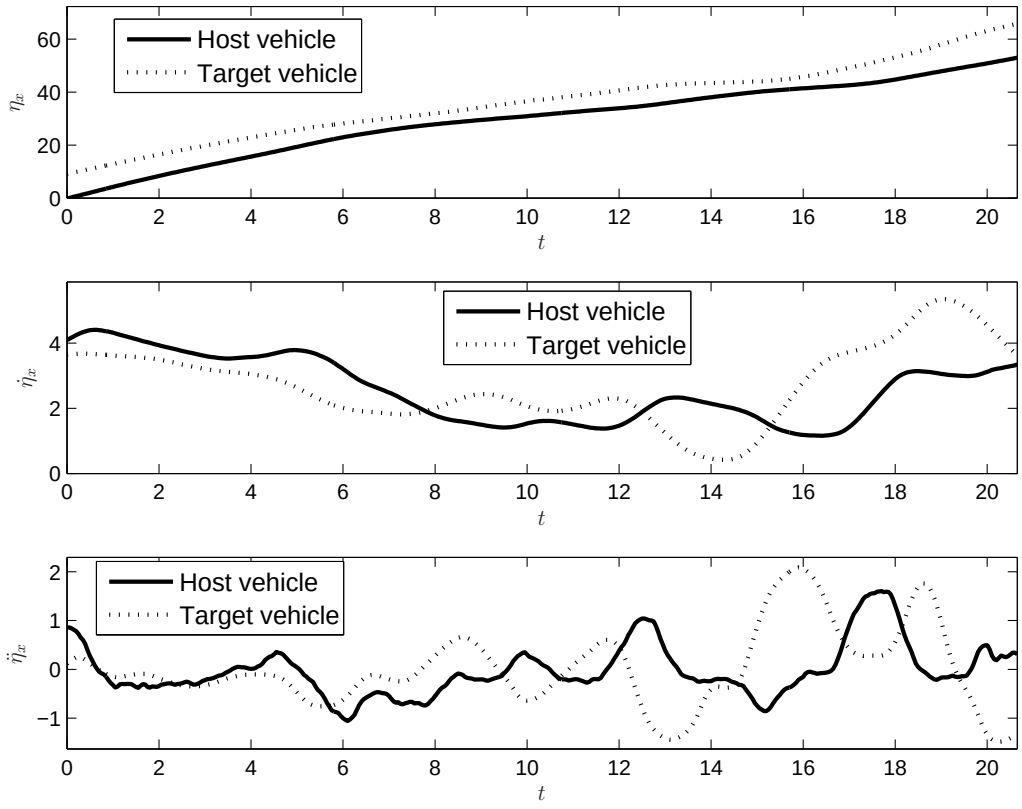


Figure 5: The positions, velocities and accelerations of the two vehicles, in the second sequence.

to specify how. For some of the parameters, it appears equally likely that they should increase by, say 30%, or to decrease by the same amount. For the process model in (13) and (14) to agree with this, we use the parameterization

$$\boldsymbol{\theta}_k = \begin{bmatrix} \log \frac{\alpha_{lo}(k)}{\tilde{\alpha}_{lo}} \\ \log \frac{\alpha_{in}(k)}{\tilde{\alpha}_{in}} \\ \log \frac{\bar{t}_h(k)}{\tilde{t}_h} \\ \dot{\eta}^h_{\text{x-ref}}(k) \end{bmatrix}, \tag{32}$$

where $\tilde{\alpha}_{lo} = 1$, $\tilde{\alpha}_{in} = 70$ and $\tilde{t}_h = 2$. The other model parameters are $\boldsymbol{\mu} = [0\ 0\ 0\ 6]^T$, $\gamma = 0.95^{T_s}$ and $\mathbf{Q}_\theta = \text{diag}[1\ \ 1\ \ 1\ \ 0.08]/80$.

The filter recursion

Here we present a recursive algorithm to compute $p(\boldsymbol{\theta}_k | \mathbf{Z}^h_k, \mathbf{Z}^t_L, \mathcal{M}_2)$ from $p(\boldsymbol{\theta}_{k-1} | \mathbf{Z}^h_{k-1}, \mathbf{Z}^t_L, \mathcal{M}_2)$. We use the Unscented Kalman Filter (UKF), which is a

Kalman-like technique to approximate the updated density[2] $p(\boldsymbol{\theta}_{k-1}|\mathbf{Z}_k^h, \mathbf{Z}_L^t, \mathcal{M}_2)$ and the predicted density $p(\boldsymbol{\theta}_k|\mathbf{Z}_k^h, \mathbf{Z}_L^t, \mathcal{M}_2)$.

In the update step, we adjust the prior density with the information obtained from the likelihood function[3]

$$p(\mathbf{z}_k^h|\boldsymbol{\theta}_{k-1}, \mathbf{z}_{k-1}^h, \mathbf{Z}_L^t) \propto \mathcal{N}\left(\mathbf{B}^\dagger\left[\mathbf{z}_k^h - \mathbf{A}\mathbf{z}_{k-1}^h\right]; \mathbf{u}_{k-1}(\boldsymbol{\theta}_{k-1}, \mathbf{z}_{k-1}^h, \mathbf{Z}_L^t), \mathbf{Q}_z\right). \quad (33)$$

In (33), we have used the singularity of $p(\mathbf{z}_k^h|\boldsymbol{\theta}_{k-1}, \mathbf{z}_{k-1}^h, \mathbf{Z}_L^t)$ (there are only uncertainties in some dimensions) in order to reduce the dimension, by multiplying with the Moore-Penrose pseudoinverse, $\mathbf{B}^\dagger$. To employ the UKF, it is more convenient to express (33) as

$$\mathbf{B}^\dagger\left[\mathbf{z}_k^h - \mathbf{A}\mathbf{z}_{k-1}^h\right] = \mathbf{u}_{k-1}(\boldsymbol{\theta}_{k-1}, \mathbf{z}_{k-1}^h, \mathbf{Z}_L^t) + \mathbf{v}_k^z, \quad (34)$$

where $\mathbf{B}^\dagger\left[\mathbf{z}_k^h - \mathbf{A}\mathbf{z}_{k-1}^h\right]$ is regarded as the measurement equation. The update step of the standard UKF algorithm can now be applied based on (34), see [26], [27]. We employ the UKF parametrization defined in [27] with $[\alpha\ \beta\ \kappa] = [1.05\ 2\ 0]$.

In the prediction step, we assume that we have a Gaussian prior density,

$$p(\boldsymbol{\theta}_{k-1}|\mathbf{Z}_k^h, \mathbf{Z}_L^t, \mathcal{M}_2) \approx \mathcal{N}(\boldsymbol{\theta}_{k-1}; \hat{\boldsymbol{\theta}}_{k-1|k}, \mathbf{P}_{k-1|k}). \quad (35)$$

Since the process model for $\boldsymbol{\theta}_k$, described in (14), is linear and Gaussian, the predicted density is also Gaussian

$$p(\boldsymbol{\theta}_k|\mathbf{Z}_k^h, \mathbf{Z}_L^t, \mathcal{M}_2) \approx \mathcal{N}(\boldsymbol{\theta}_k; \hat{\boldsymbol{\theta}}_{k|k}, \mathbf{P}_{k|k}), \quad (36)$$

where $\hat{\boldsymbol{\theta}}_{k|k} = \gamma\hat{\boldsymbol{\theta}}_{k-1|k} + (1-\gamma)\boldsymbol{\mu}$ and $\mathbf{P}_{k|k} = \gamma^2\mathbf{P}_{k-1|k} + \mathbf{Q}_z$.

From an implementation perspective, the UKF is very convenient to use but has the disadvantage that it requires us to evaluate $\mathbf{u}_{k-1}(\boldsymbol{\theta}_{k-1}, \mathbf{z}_{k-1}^h, \mathbf{Z}_L^t)$ several times (once for each sigma point; since $\boldsymbol{\theta}_{k-1}$ contains 4 elements we have $2 \times 4 + 1 = 9$ sigma points).

We have here written the densities conditional on $\mathbf{Z}_L^t$. The calculations when we instead condition on $\mathbf{Z}_k^t$ are analogous, but with a slight difference in the update step. As we have seen, the function $\mathbf{u}_{k-1}$ takes $\mathbf{Z}_L^t$ as input in order to know how to compensate for the vehicle interaction, $c_{in}(\cdot,\cdot)$. When $\mathbf{z}_k^t, \mathbf{z}_{k+1}^t, \ldots, \mathbf{z}_L^t$ are unknown, we predict these based on $\mathbf{z}_{k-1}^t$. The approach taken here is to use the predicted mean of the CA model, $\hat{\mathbf{z}}_{k-1+l}^t = \mathbf{A}^l\mathbf{z}_{k-1}^t$. (Note that t stands for target whereas $\mathbf{A}$ is raised to the power of l).

[2] As $\mathbf{z}_k^h$ is related more directly to $\boldsymbol{\theta}_{k-1}$ than to $\boldsymbol{\theta}_k$, the suggested order (first update, then predict) is more straightforward; normally one would instead first predict, compute $p(\boldsymbol{\theta}_k|\mathbf{Z}_{k-1}^h, \mathbf{Z}_L^t, \mathcal{M}_2)$, and then do the update, calculate $p(\boldsymbol{\theta}_k|\mathbf{Z}_k^h, \mathbf{Z}_L^t, \mathcal{M}_2)$.

[3] In previous sections, $\mathbf{u}_{k-1}(\cdot)$ only took $\mathbf{x}_{k-1}$ as input. However, when $\mathbf{Z}_L^t$ is known we use it as an additional input. To understand how this influences the computations, see the discussion at the end of this subsection, Section 5.2.

5.3 Results

As mentioned above, we observe two cars driving on a straight road in a city. We evaluate the criteria in Section 5.1 on two different sequences, see Fig. 4 and 5. Both sequences are interesting since there is interaction between the two vehicles; the host vehicle is forced to slow down when the target vehicle slows down. Measurement data is obtained with the frequency 20 Hz. However, to speed up the optimization algorithm we assume $T_s = 0.3$ and a prediction horizon $N = 10$ (see (19)) during our calculation of $\mathbf{u}_{k-1}(\cdot)$.

Prediction accuracy

We now wish to evaluate how well the predicted density $p(\mathbf{z}_k^h | \mathbf{Z}_{k-1}^h, \mathbf{Z}_{k-1}^t, \mathcal{M}_2)$ corresponds to the observed trajectories. The motion model in (11), is driven only by the jerk values between the sample instances. We will therefore compare the predicted jerk values (obtained from $\mathbf{u}_{k-1}(\cdot)$) with the values computed from the measurements. For the CA model we use the values $\sigma^2_{\dddot{\eta}_x} = 0.95$ and $\sigma^2_{\dddot{\eta}_x} = 1.08$ for the first and second sequences, respectively, since these values maximize the likelihood, see Section 5.2.

Fig. 6 shows the jerk predictions obtained from our model

$$E\{\dddot{\eta}_x^h(k) \big| \mathbf{Z}_{k-1}^h, \mathbf{Z}_{k-1}^t, \mathcal{M}_2\},$$

which is denoted here as modified CA, together with the observed jerk values. Note that the jerk predictions provided by the CA model are zero at all times. As we can see, the jerk predictions are fairly accurate most of the time. In the first sequence, the predictions are less accurate at around $t = 6$ when the host vehicle temporarily accelerates even though the target vehicle slows down. Similarly, for the second sequence the model incorrectly predicts an increased deceleration at $t \approx 14$. However, overall the predictions from the modified CA model are substantially better than those from the standard CA model.

For the predicted density to be accurate, it is also important that it has a reasonable variance. Let us denote the error between the predicted jerk and the observed jerk by $\Delta\dddot{\eta}_x^h(k)$. Ideally, we would like the variance of the predicted density,

$$\mathrm{Var}\{\dddot{\eta}_x^h(k) \big| \mathbf{Z}_{k-1}^h, \mathbf{Z}_{k-1}^t, \mathcal{M}_2\},$$

to be identical to $(\Delta\dddot{\eta}_x^h(k))^2$ on the average. As discussed previously, we hope that the model also has an ability to know when the predictions are reliable and when they are more uncertain, see Section 3.3.

The variance of the predictions (the uncertainties in the predictions) and the prediction errors are given in Fig. 7 and 8 for the respective data sequences. On the average, the variance is 0.65 and 0.83 in the first and second sequence, respectively, whereas the correspond errors, $(\Delta\dddot{\eta}_x^h(k))^2$, are 0.45 and 0.36. The model is thus overestimating the uncertainties, but at least the order of magnitude is correct.

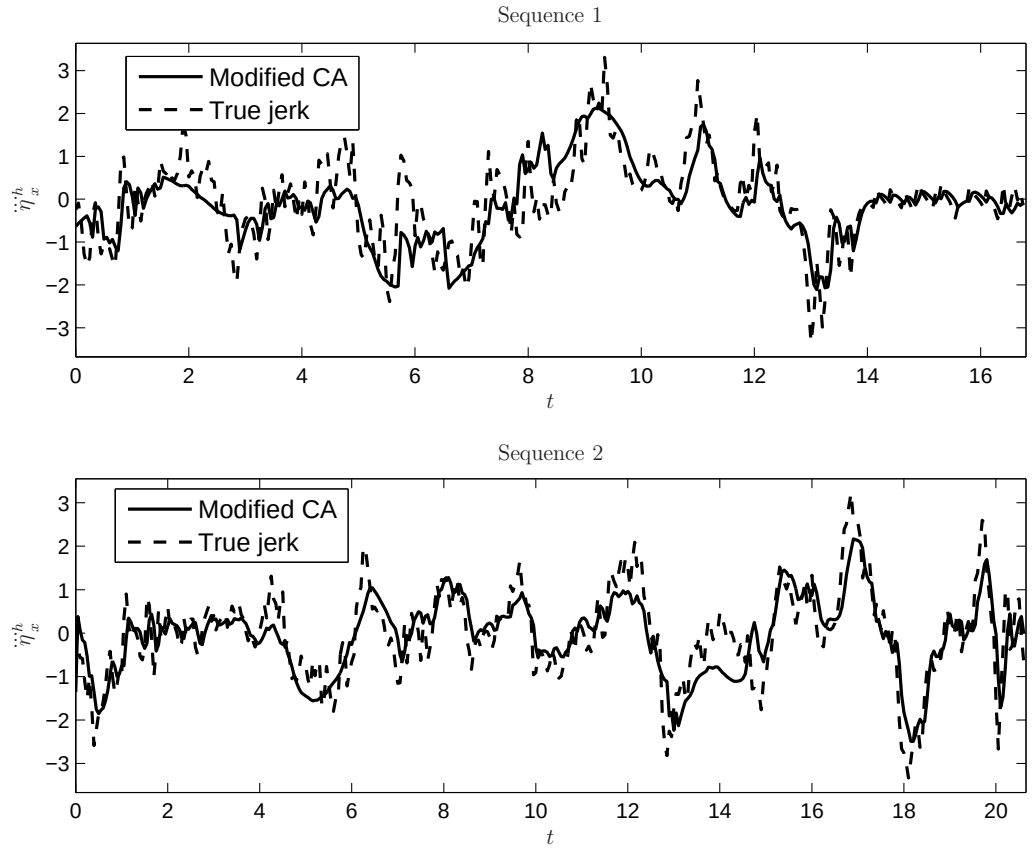


Figure 6: Predicted and true jerk for the two trajectories.

At several occasions, like at around $t = 6$ and $t = 8$ in the first sequence and at $t \approx 13$ in the second sequence, larger prediction variances coincide with larger errors. Since an increased prediction variance can only be caused by uncertainties in $\boldsymbol{\theta}_k$, this is a good sign in favor of the filter described in Section 5.2. Though the evaluation is limited, this is still a promising indication.

Model likelihood

We stated above that we wish to compute $p(\mathbf{Z}_L^h, \mathbf{Z}_L^t | \mathcal{M}_i)$ for $\mathcal{M}_1$ and $\mathcal{M}_2$, as a measure on how well the models explain the data. However, by studying

$$\log p(\mathbf{Z}_k^h | \mathbf{Z}_L^t, \mathcal{M}_i) = \sum_{j=1}^{k} \log p(\mathbf{z}_j^h | \mathbf{Z}_{j-1}^h, \mathbf{Z}_L^t, \mathcal{M}_i), \tag{37}$$

for $k = 1, \ldots, L$ we also get information about how well the models explain different parts of the sequences (the logarithm is merely included to enable us to illustrate functions of different magnitudes in the same figure). In addition, we also include

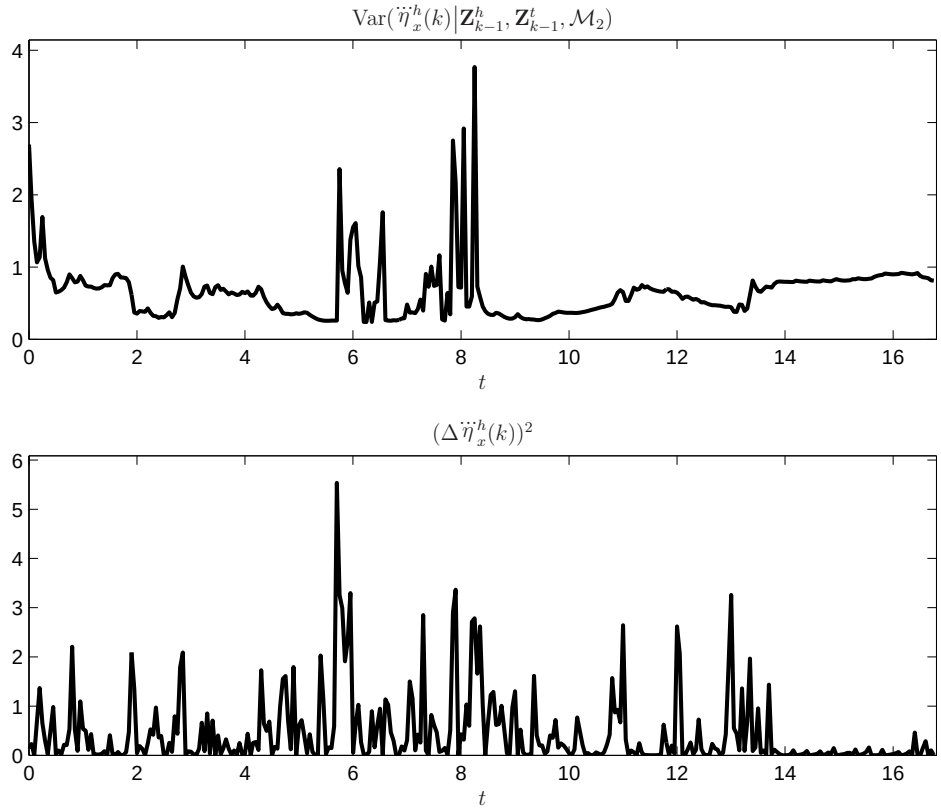


Figure 7: Prediction uncertainties and prediction errors for the first sequence.

the function

$$\sum_{j=1}^{k} \log p(\mathbf{z}_j^h \big| \mathbf{Z}_{j-1}^h, \mathbf{Z}_j^t, \mathcal{M}_2), \tag{38}$$

which is made up of the predictive densities discussed in Section 5.3. In the figures, we refer to $\log p(\mathbf{Z}_k^h \big| \mathbf{Z}_L^t, \mathcal{M}_1)$ as CA, $\log p(\mathbf{Z}_k^h \big| \mathbf{Z}_L^t, \mathcal{M}_2)$ as Conditional CA and $\sum_{j=1}^{k} \log p(\mathbf{z}_j^h \big| \mathbf{Z}_{j-1}^h, \mathbf{Z}_j^t, \mathcal{M}_2)$ as Modified CA.

The three log-likelihood functions are shown in Fig. 9 and 10 for the first and second sequences, respectively. Generally speaking, Modified CA and Conditional CA are similar and significantly larger (better) than CA. Compared to the proposed model, here represented by Modified CA and Conditional CA, CA explains the data very poorly when the acceleration changes notably; its likelihood function decreases quickly at those times whereas both Modified CA and Conditional CA are much smoother. The only clear exception is the Conditional CA which decreases substantially around $t = 6$ in the first sequence. The reason for this is that the model expects the driver to break harder when there is instead a slight acceleration; the same phenomenon was discussed previously related to Fig. 6. To

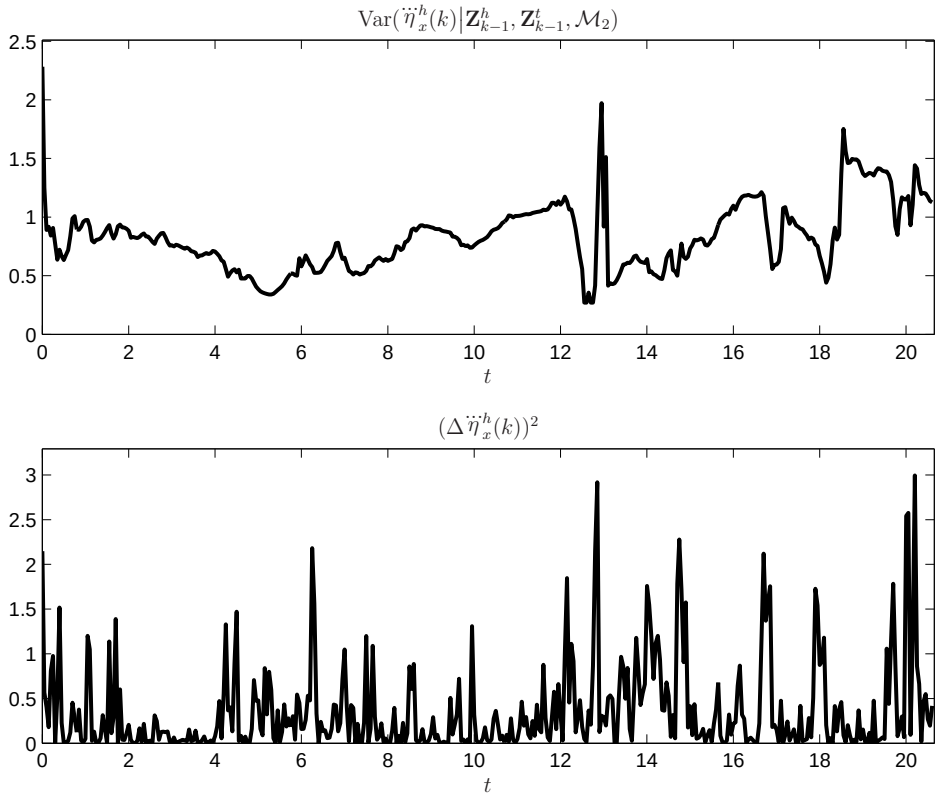


Figure 8: Prediction uncertainties and prediction errors for the second sequence.

put the numbers in Fig. 9 and Fig. 10 into perspective, Jeffreys [28] states that a log-likelihood difference greater than 2 should be viewed as decisive evidence as to what model is true.

6 Conclusion and future work

We have presented a general framework for modelling vehicle motion. By including the influence of the driver into the model, an improved prediction ability is obtained. Moreover, with a formalized treatment of uncertainties in the underlying model parameters, the description of the prediction uncertainties is improved. Specific attention was here given to adapting the model to when the driver follows the right lane of the road. Accurate data from real driving situations was collected to study the model properties. For the evaluated test scenarios, we have shown that a model developed using our framework describes reference data considerably better than the commonly used CA model.

There are several possibilities to further develop the motion model. First of all, more driver intentions should be added to capture interesting scenarios such as lane changes, overtakings and distracted drivers. For these scenarios, the

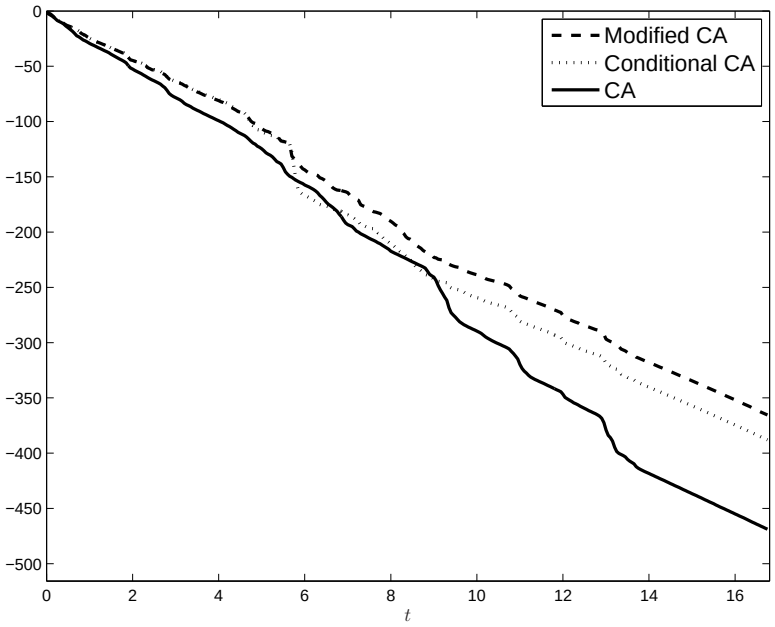


Figure 9: The log-likelihood functions for the first data sequence.

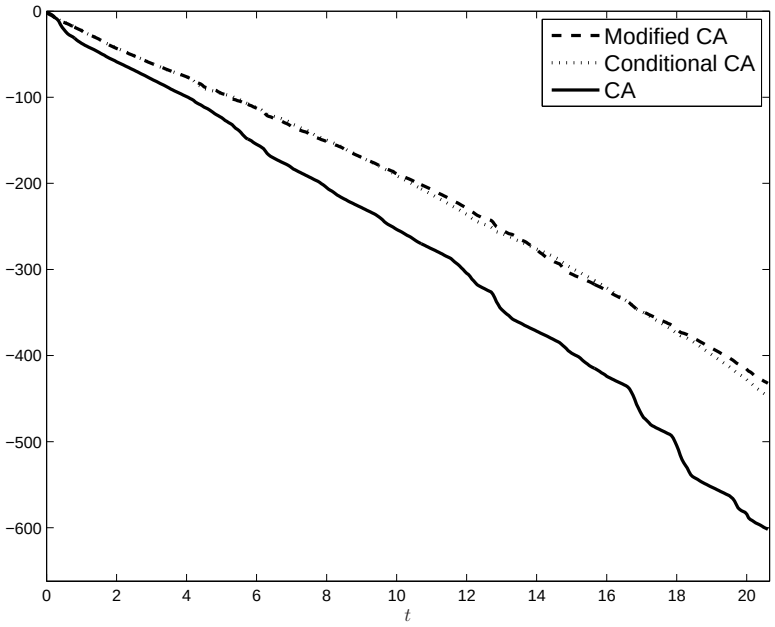


Figure 10: The log-likelihood functions for the second data sequence.

main challenge is to design suitable cost functions which accurately represent the driver behavior and still result in a manageable optimization problem. Further, a more extensive validation is needed using more test scenarios and several different drivers. The validation data should be used to investigate the model's capability to adapt to different driving styles, and also be used to fine tune the cost functions presented in this paper. Finally, it would be interesting to explore the usefulness of the prediction capability of the model in a decision making algorithm. For instance in a collision avoidance application or for run off road detection.

References

[1] "Requirements for preventive safety applications," PReVENT IP Deliverable 4 - 16.04.2005, www.ip-prevent.org.

[2] C. P. Robert, *The Bayesian choice: a decision-theoretic motivation*. Springer, 1994.

[3] X. R. Li and V. Jilkov, "Survey of maneuvering target tracking. part I. dynamic models," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 39, no. 4, pp. 1333–1364, 2003.

[4] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applicarions to tracking and navigation*. New York, NY: Wiley, 2001.

[5] S. Blackman and R. Popoli, *Design and analysis of modern tracking systems*. Norwood, MA: Artech House, 1999.

[6] R. Mobus, M. Baotic, and M. Morari, "Multi-object adaptive cruise control," in *Hybrid Systems: Computation and Control. 6th International Workshop, HSCC 2003*, Prague, Czech Republic, September 2003.

[7] A. Broadhurst, S. Baker, and T. Kanade, "Monte Carlo road safety reasoning," in *Intelligent Vehicles Symposium, Proceedings IEEE*, Las Vegas, NV, USA, 2005.

[8] P. Gipps, "A behavioural car-following model for computer simulation," *Transportation Research Part B*, vol. 15, no. B, pp. 105–111, 1981.

[9] http://www.isi-padas.eu/.

[10] http://www.iterate-project.eu/.

[11] S. M. LaValle, *Planning algorithms*. Cambridge University press, 2006.

[12] L. Svensson and J. Gunnarsson, "A new motion model for tracking of vehicles," in *14th IFAC Symposium on System Identification*, Newcastle, Australia, March 2006.

[13] J. Gunnarsson, L. Svensson, F. Bengtsson, and L. Danielsson, "Joint driver intention classification and tracking of vehicles," in *Nonlinear Statistical Signal Processing Workshop 2006*, Cambridge, UK, September 2006.

[14] R. Kalman, "A new approach to linear filtering and prediction problems," *Trans. ASME, Ser. D. Journal Basic Eng.*, vol. 82, pp. 24–45, 1960.

[15] J. Meriam and L. Kraige, *Engineering mechanics*. New York: Wiley, 1993, vol. 2.

[16] U. Kiencke and L. Nielsen, *Automotive Control Systems For Engine, Driveline and Vehicle*. Springer, 2000.

[17] N. Kaempchen, K. Weiss, M. Schaefer, and K. Dietmayer, "IMM object tracking for high dynamic driving maneuvers," in *Intelligent Vehicles Symposium, 2004 IEEE*, June 2004.

[18] A. Eidehall, "An automotive lane guidance system," Licentiate Thesis no. 1122, Department of Electrical Engineering, Linköping University, Linköping Sweden, 2004.

[19] E. Dickmanns and A. Zapp, "A curvature-based scheme for improving road vehicle guidance by computer vision," in *Proceedings of the SPIE Conference on Mobile Robots*, 1986.

[20] G. Leitmann, *The calculus of variations and optimal control*. New York: Plenum Press, 1981.

[21] J. Betts, *Practical methods for optimal control using nonlinear programming*. Philadelphia, PA: SIAM, 2001.

[22] W. Murray, P. E. Gill, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM J. Optim.*, vol. 12, pp. 979 – 1006, 2002.

[23] "VGU, VV publikation 2004:80, grundvärden," June 2004, iSSN: 1401-9612.

[24] E. Bertolazzi, F. Biral, and M. D. Lio, "Future advanced driver assistance systems based on optimal control: the influence of 'risk functions' on overall system behavior and on prediction of dangerous situations," in *Intelligent Vehicles Symposium*, Parma, Italy, 2004.

[25] E. Bertolazzi, F. Biral, M. D. Lio, A. Saroldi, and F. Tango, "Supporting drivers in keeping safe speed and safe distance: The saspence subproject within the european framework programme 6 integrating project prevent," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 525 – 538, 2010.

[26] S. Julier and J. Uhlmann, "A new extension of the Kalman filter to nonlinear systems," in *The Proceedings of AeroSense: The 11th International Symposium on Aerospace/Defense Sensing, Simulation and Controls, Multi Sensor Fusion, Tracking and Resource Management II*, 1997.

[27] E. Wan and R. V. D. Merwe, "The unscented Kalman filter for nonlinear estimation," *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 2000.

[28] H. Jeffreys, *Theory of Probability*, 3rd ed. Oxford Univiversity. Pres, 1961.

Paper V

A design architecture for sensor data fusion systems with application to automotive safety

F. Bengtsson[1] and L. Danielsson

In *Proceedings of Intelligent Transport Systems World Congress* 2008, New-York, November 2008.

[1]Fredrik changed his name from Bengtsson to Sandblom in 2009

A DESIGN ARCHITECTURE FOR SENSOR DATA FUSION SYSTEMS WITH APPLICATION TO AUTOMOTIVE SAFETY

Fredrik Bengtsson, Lars Danielsson

In this paper we present a modular sensor data fusion functional architecture, tailored for development of automotive active safety systems. The purpose of the fusion system is to provide active safety applications with accurate knowledge regarding the environment surrounding the vehicle. Our proposed functional architecture is designed in such way that the fusion system is easy to maintain, upgrade and re-use. These aspects are assessed by the use of a reference implementation which is evaluated in terms of tracking performance and scalability. Furthermore, the reference implementation demonstrates that a system can be implemented using rapid prototyping tools, from which we can automatically generate c-code.

INTRODUCTION

In order for automotive active safety applications to make decisions about when to warn or intervene in dangerous traffic situations, reliable information about the traffic environment surrounding the vehicle is needed. Measurements on the environment are typically supplied by sensors mounted on the vehicle, for example radar sensors and vision systems. In order to meet the challenging requirements posed by safety critical applications, it is increasingly common to use information from several on-vehicle sensors in a *data fusion* framework. The task of fusing sensor data is performed by a *tracking system* or *perception layer* (1).

Tracking systems have been researched extensively and there is a significant amount of results available regarding system design, e.g. (2, 3). In this paper we have regarded the task of system design from an automotive safety system research perspective, which in some aspects differ from many other tracking applications; sensors or hardware are subject to change and multiple applications pose different requirements. At the same time, hardware and software should to a high degree be shared components. The result is a functional architecture that allows for robust, versatile implementations using known tracking strategies.

The proposed architecture is used in a reference implementation to evaluate the design principles. Algorithms are implemented using Mathworks Embedded MATLAB, from which it is possible to generate c-code and demonstrate the system in real-time on prototype PC hardware. The work is supported by Swedish *Intelligent Vehicle Safety Systems* (IVSS) program and is a part of the *Sensor Fusion for Safety* (SEFS) project.

PROBLEM FORMULATION

The main objective for this paper is to describe a functional architecture that can be used when designing a fusion system, all the way from a research platform to the vehicle production system. In this section we discuss aspects that need to be considered in order to solve this task.

Fusion Architecture

Active safety system research is aided by a perception layer where sensors can be exchanged and different tasks can be developed and evaluated separately. It is also desirable to have an architecture that allows code to be re-used through different development steps, *i.e.* it should be straightforward to go from the research system to an in-vehicle system. Embodiments intended for in-vehicle programs need to be developed in parallel with safety applications, while as long as possible keeping the door open for introducing new sensors or achievements regarding fusion methods. For these reasons, an architecture suitable for both research and embedded implementations, as illustrated in Figure 1, would be very useful.

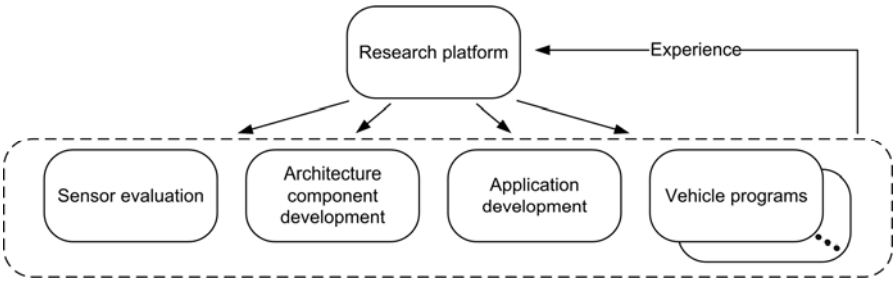


Figure 1: Fusion system components are used in multiple applications, ranging from sensor evaluation and research to embedded implementations in a vehicle program. A common architecture makes it possible for every application to make use of recent developments.

Therefore we aim to present a modular architecture that facilitates fusion system development jointly with sensor evaluation and in-vehicle studies.

Practical considerations

In the most natural multi-sensor tracking scenario all sensors are synchronized and the complete state vector is updated using all available information simultaneously. However, in practice sensor data is delayed due to internal signal processing algorithms or limited communication bandwidth and will arrive to the fusion in an asynchronous manner. If this is not considered, performance will suffer, e.g. as shown in Figure 2.

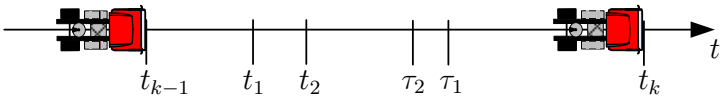


Figure 2: During t_{k-1} to t_k a truck moves from left to right in the figure. Sensor one reports the position at time t_1 and sensor two reports the position at t_2. However, the detections are delayed until τ_1 and τ_2 respectively. In this figure $\tau_1 > \tau_2$, which implies that the vehicle moved forward in the time interval $(t_{k-1}, \tau_2]$ and then backwards during $(\tau_2, \tau_1]$.

The arrival order of measurements is generally unknown in advance, and in a system with delays it may happen that we receive a so called out-of-sequence measurement. As shown in Figure 3, it is possible for a measurement to arrive after the state vector has been updated with information from a newer measurement, *i.e.* out-of-sequence, in practice a problem which requires special treatment.

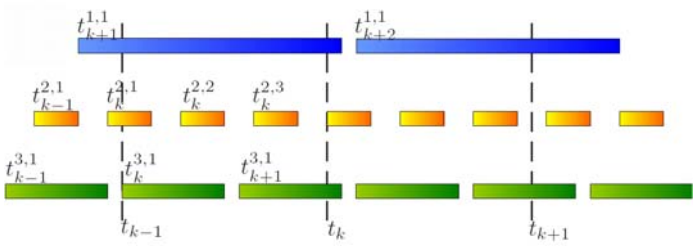


Figure 3: Sensor data measured at $t_k^{(i,j)}$, arrive to the fusion at time t_k, where i define the sensor and j is a measurement counter for that sensor. The fusion system receives data at times $(t_0,\ t_1,\ \ldots, t_k,\ \ldots)$ and must handle asynchronous measurements, some of which take longer time to reach the fusion system than others. One reason for such behaviour is that certain sensors transmit accumulated measurements in a burst, *i.e.* a list of measurements. Note that $t_k^{(i,j)}$ may be smaller than t_{k-1}.

Filter technique and data association

Algorithms typically make use of a Kalman filter (KF) framework, while sensor specific methods are used in pre-processing steps such as data association and track initialization. It is important that the architecture supports filtering techniques such as Extended Kalman filter, Unscented Kalman filter, multiple model frameworks and, to some extent, Monte Carlo methods. At the same time, sensor specific adaptations must be allowed.

Modularity

There is often an architectural conflict between a modular and re-usable system and the optimal signal processing algorithm, fully exploiting all the information in signals and models. To achieve the flexibility described above, processing should be performed in a certain order with limited information exchange between adjacent functional blocks. In other words it needs to be modular, which may lead to suboptimal processing.

TRACKING SYSTEM

The task of a tracking system is to, at a time t, describe the measured environment as good as possible given all available data, including uncertainties. Here we introduce the discrete-time state vector $\mathbf{x}_k$ to contain everything we want to know at time $t_k = kT_s(k)$, where $T_s(k)$ is the system sample time and $k \in \mathbb{N}$ a counter. Similarly, we introduce $\mathbf{y}_k$ to be the vector of measurements on the surroundings received in the time period $(t_{k-1}\ ,\ t_k]$. For a system with N sensors we form

$$\mathbf{y}_k = [(\mathbf{y}_k^1)^T\ (\mathbf{y}_k^2)^T\ \ldots\ (\mathbf{y}_k^N)^T]^T \tag{1}$$

The task can now be formulated as to calculate the posterior density $p(\mathbf{x}_k|\mathbf{y}_1, \ldots \mathbf{y}_k)$. In order to do this, two statistical models are needed, a *process model* and a *measurement model*. Textbooks which thoroughly explain estimation and modelling are *e.g. (4, 5).*

In an automotive context, a tracking system is responsible for refining the information supplied by onboard sensors to supply safety applications with information about the surrounding traffic situation. In Figure 4 this relation is depicted together with components

that are necessary in order to calculate $p(\mathbf{x}_k|\mathbf{y}_1, \ldots \mathbf{y}_k)$. This includes *gating and data association*, *track management*, *measurement update* and *state prediction*. More information about these different components can be found in *e.g.* (3, 6, 7).

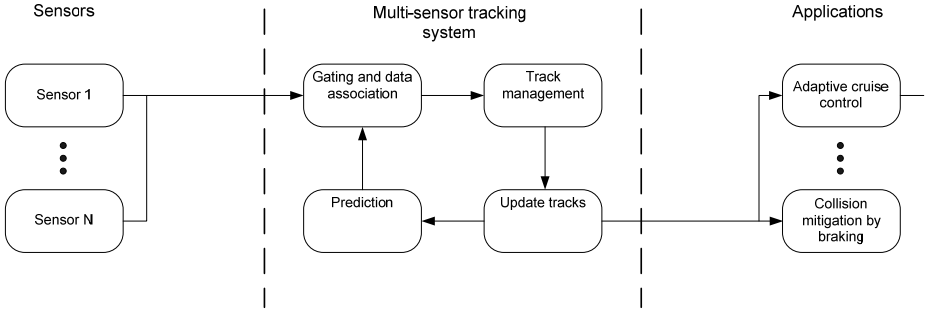


Figure 4: Schematic view of a tracking system and its components, in the context of typical automotive comfort or safety applications.

In a multi-sensor system, as in Figure 4, depending on the update rate of the tracking system, the measurement vector from each sensor, $\mathbf{y}_k^i$, may be composed of several measurements of different age. In this case $\mathbf{y}_k^i$ has the following structure

$$\mathbf{y}_k^i = [(\boldsymbol{\gamma}_k^{(i,1)})^T \; \tau_k^{(i,1)} \; (\boldsymbol{\gamma}_k^{(i,2)})^T \; \tau_k^{(i,2)} \; \ldots \; (\boldsymbol{\gamma}_k^{(i,M^i(k))})^T \; \tau_k^{(i,M^i(k))}]^T \tag{2}$$

where $M^i(k)$ are all measurement lists from sensor i received in the time period $(t_{k-1}, t_k]$, with corresponding time stamps $\tau_k^{(i,0:M^i(k))}$. That is to say, $\boldsymbol{\gamma}_k^{(i,j)}$ contains all measurements received at time $\tau_k^{(i,j)} \in [\tau_k^{i,j-1}, t_k]$. Note that $\tau_k^{i,0}$ is defined as the time sensor i last delivered data prior to t_{k-1} and that $M^i(k) \geq 0$, i.e. sensor i can deliver zero or multiple measurements during an update cycle of the fusion system. If the sensor delays are known, the time for the actual measurement $t_k^{(i,j)}$, can be derived from $\tau_k^{(i,j)}$. Thus it is possible for $t_k^{(i,j)}$ to be smaller than t_k. Figure 3 shows how measurements from three different sensors fall in different measurement vector slots, for example $\{\boldsymbol{\gamma}_k^{(2,1)}, \boldsymbol{\gamma}_k^{(2,2)}, \boldsymbol{\gamma}_k^{(2,3)}, \boldsymbol{\gamma}_k^{(3,1)}\} \in \mathbf{y}_k$.

FUSION ARCHITECTURE

A central requirement on the perception layer is a high degree of modularity, so that algorithm components can be continuously developed and sensors may be exchanged or added to the system. Further, it must be suitable for automotive safety systems regarding *e.g.* in- and output data, computational load and terms of robustness, etc. It is a challenge to design a system with these properties, but the work is aided by an appropriate high level functional design.

Architecture outline

Our goal is to treat perception layer functional tasks in a fashion that facilitate usage according to Figure 1. One important aspect is how to control the flow of information; the magnitude of data transfer easily grows during development. We suggest a one-way data flow where usage of internal variables is minimized and transferred variables are well

specified and available to modules in a predictable fashion. A well organised data flow that has grown during development can be slimmed at a later stage, depending on which methods are actually used. Functional tasks are grouped into modules, chosen to facilitate separate development and are shown in Figure 5, supporting a variety of state update methods to be used for each sensor individually. Models should be restricted to the class of models with the Markov property. However, usage of statistical linearization methods allow models to be readily exchanged – only the transformation function need to be defined – which further supports modularity.

Functional modules

A functional architecture supporting an asynchronous fusion scheme, as discussed in the problem formulation, is shown in Figure 5. The general idea is to have a common prediction and delay compensation block, run at each internal iteration in the fusion scheme, and to divide the subsequent processing modules by sensor data origin. A dedicated sensor processing module is used for each sensor, *e.g.* if internal sensor data is to be processed the filter internal sensors module is activated. After all data in a cycle has been processed, additional track handling strategies may be applied on a global level. It is recommended to treat all parameters needed in the fusion system as input and output signals, to avoid internal "global parameters".

The architecture is readily extended with new blocks, *e.g.* including data from an e-horizon system or implementing an estimator to be used for complex posterior distributions. Adaptations such as performing *e.g.* track-to-track fusion, can be assigned to existing modules.

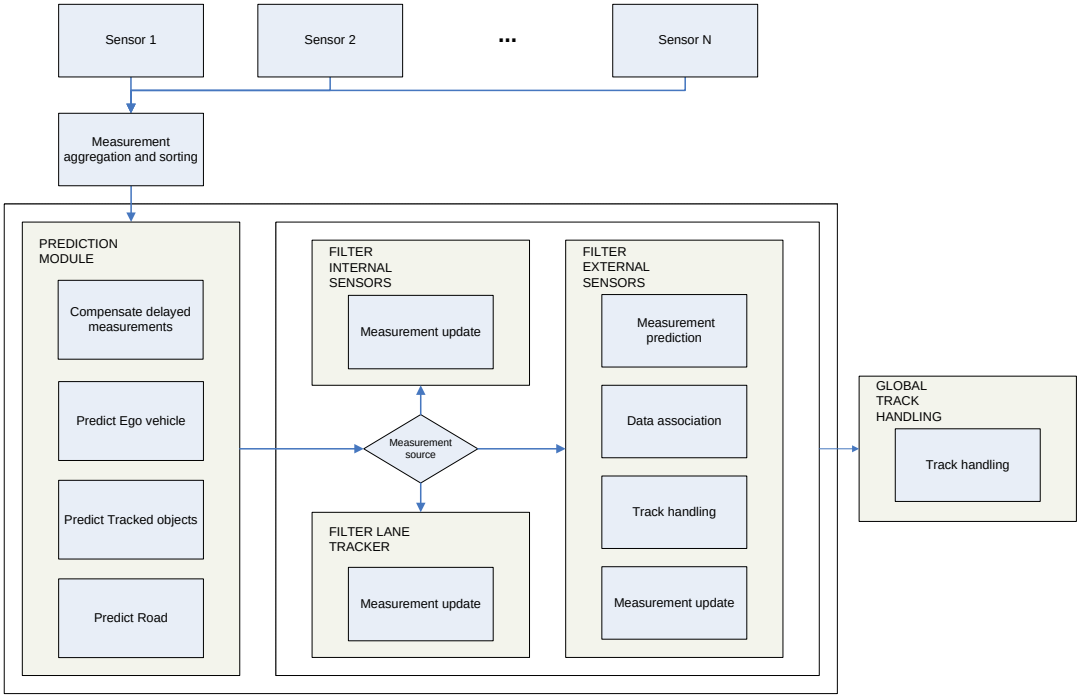


Figure 5: Fusion algorithm functional architecture. The large block is called once for each received measurement, executing necessary prediction and filter modules in the correct order.

Measurement aggregation and sorting

The information from each sensor arriving to the fusion system is stored until the beginning of the next system cycle, allowing *e.g.* platform specific conversions. The aggregated measurements are sent to the fusion system in a sorted manner; the measurement that occurred first is sent first, etc. Naturally, any knowledge regarding expected measurement delays should be incorporated in the sorting.

Prediction module

The prediction of the state is made in four basic components, *predict ego vehicle*, *predict tracked objects, predict road* and *compensate delayed measurements*. These components make sure that the measurements and state vector are aligned in time for the upcoming filter blocks. Typically this module is used to predict $\mathbf{x}_{k-1}$ to the time of the earliest measurement $t_{\kappa_1} \in (t_{k-1}, t_k]$ and subsequently with the next measurement at $t_{\kappa_2} \in [t_{\kappa_1}, t_k]$. The *compensate delayed measurements* block is called in the event of an out-of-sequence-measurement, and several strategies can be used to process such data. If models allow, a so called retrodiction step can be incorporated in the update procedure, explained *e.g.* in (8). An optimal retrodiction increases system complexity as it requires us to store previous states, a problem which can be avoided using sub-optimal alternatives. Nevertheless, blocks in the corresponding *filter external sensor module* must support retrodicted state updates which in itself add complexity. A procedure that does not affect following modules is to disregard measurements deemed to old, and to adapt the measurement distribution accordingly for measurements actually used.

Filter internal sensors

The ego vehicle is often modelled in more detail than observed vehicles and measurements not available from other vehicles are typically obtained at a high rate. For these reasons it is suitable to perform the filtering of measurements from internal sensors in a separate module.

Filter lane tracker

Lane tracker measurements are often pre-processed and should if possible be treated using knowledge regarding internal filtering models. Nevertheless, this does not prevent $\mathbf{r}_k$ from being updated with data from external sensors in another module.

Filter external sensors

Each sensor observing the environment outside the ego vehicle has a corresponding filter module. Sensors may observe the world quite differently and it is here detailed sensor models are used to perform *measurement prediction*, *data association* and *state updates*. Track management such as *track initialization* or calculating *track score* can be carried out, and in multiple model frameworks, model probabilities are updated. Note that the module input generally is an object list, which is required for efficient data association.

Global track management

Track deletion, validation and merging of tracks can be done on a global level to complement the limited set of track management tasks done locally for each sensor. Strategies for object classification, for example during track validation, can be implemented here.

IMPLEMENTATION

We have implemented a simple, but fully autonomous, tracking system to be evaluated before proceeding to develop components further. It is based on the proposed architecture and runs on a platform that can be used in real-time in vehicle demonstrators (9). Detailed information regarding this particular implementation can be found in (6, 7).

Parameterization

For a total of n_c tracked cars, the state vector $\mathbf{x}_k$ is partitioned as

$$\mathbf{x}_k = [(\mathrm{z}_k^{ego})^T \ (\mathrm{z}_k^1)^T \ (\mathrm{z}_k^2)^T \ \ldots \ (\mathrm{z}_k^{n_c})^T \ \mathrm{r}_k^T]^T \tag{3}$$

where $\mathbf{z}_k^i$ is the state vector for vehicle i and $\mathbf{r}_k$ describes the road. The choice of states in $\mathbf{z}_k^i$ is directly affected by the choice of vehicle motion model. Similarly, the parameterization of $\mathbf{r}_k$ is coupled with the choice of road process model.

Sensor setup

The evaluation sensor setup is limited to one radar, a lane tracker camera and internal sensors measuring ego vehicle states.

Radar

Delphi's ACC3.5, 77GHz Automotive Radar, with a detection range of approximately 150 meters and an opening angle of 16° is mounted in the front of the vehicle. Up to 20 unfiltered detections, each consisting of range (r [m]), range rate ($\dot{r}$ [$\frac{m}{s}$]) and azimuth (φ [rad]), grouped in an *object list* are reported every measurement cycle (100 ms). The j^{th} object list[1] is written

$$\boldsymbol{\gamma}^{(1,j)} = [r_1 \ \varphi_1 \ \dot{r}_1 \ r_2 \ \varphi_2 \ \dot{r}_2 \ \ldots \ r_{20} \ \varphi_{20} \ \dot{r}_{20}]^T, \tag{4}$$

letting index one denote the radar sensor.

Vision system

A camera based lane tracking system provides measurements on the vehicle heading relative the road Ψ_{rel} [rad], the distances to the left and right lane markings R_{off} and L_{off} [m] respectively, and the road curvature c_0 [m^{-1}]. The j^{th} measurement in a fusion cycle is

[1] for the j^{th} object list in the k^{th} iteration, but k is left out at this point for clarity reasons

$$\boldsymbol{\gamma}^{(2,j)} = [\Psi_{rel}\ R_{off}\ L_{off}\ c_0]^T, \tag{5}$$

if the vision sensor is indexed as sensor number two.

Internal sensors

Several internal sensors are available in modern vehicles. In this system the vehicle speed v $[\frac{m}{s}]$, acceleration a $[\frac{m}{s^2}]$ and yaw-rate $\dot{\Psi}$ $[\frac{\text{rad}}{s}]$ are used for ego vehicle tracking. Consequently,

$$\boldsymbol{\gamma}^{(3,j)} = [v\ a\ \dot{\Psi}]^T. \tag{6}$$

Filtering

A statistical linearization algorithm (10) is used to estimate effects of nonlinear transformations, resulting in the so called Unscented Kalman filter (UKF). Effects of linear transformations are calculated analytically. The global nearest neighbour data association method is implemented using the auction algorithm (11) and unlikely associations are ruled out using ellipsoidal gates. Unassociated measurements yield new tracks, confirmed when associated with measurements *n*-times out of *m* possible. Track deletion occurs using a similar scheme, or when uncertainties are larger than a threshold. Measurement delays are estimated and assumed known and, when using a single radar, there are no problems with out-of-sequence-measurements.

Vehicle motion model

The same parameterization and dynamic model is used for other vehicles as well as for the ego vehicle. We include the global position (ξ_x, ξ_y), heading Ψ, velocity v, yaw-rate $\dot{\Psi}$ and acceleration a in the vehicle state vector:

$$\mathbf{z}_k = [\xi_x\ \ \xi_y\ \ \psi\ \ v\ \ \dot{\psi}\ \ a\ ]^T. \tag{7}$$

The motion model is derived from the continuous-time model

$$\dot{\mathbf{z}}(t) = [v(t)cos(\psi(t))\ \ v(t)sin(\psi(t))\ \ \dot{\psi}(t)\ \ a(t)\ \ 0\ \ 0]^T + [0\ \ 0\ \ 0\ \ 0\ \ v_{\ddot{\psi}}(t)\ \ v_{\dot{a}}(t)]^T. \tag{8}$$

$v_{\ddot{\psi}}(t)$ and $v_{\dot{a}}(t)$ describe modelling errors and are continuous time Gaussian stochastic processes. Both are zero mean and white, with intensity $q_{\ddot{\psi}}$ and $q_{\dot{a}}$, respectively. A fixed step-length discrete model is used, a derivation of which is presented in (12).

Road process model

We follow suggestions from *e.g. (13)* and use a clothoid model to make a local approximation of the road curvature around the ego vehicle. The curvature c_0 changes as a linear function, so at distance η ahead of the ego vehicle, the curvature is written

$$c_0(\eta) = c_0(0) + \eta\, c_1, \tag{9}$$

The road state vector contains ego vehicle *heading angle relative the road, distances to the lane markings* and the *curvature parameters* (c_0, c_1). Apart from c_0 described in (9), all states are modelled as constants influenced by zero mean white Gaussian noise (WGN).

Measurement models

The radar sensor model is kept fairly simple, which is sufficient for this evaluation. It is assumed that the vehicle can be modelled as a point target, *i.e.* at most one measurement may originate from each vehicle. The measurement equation for one radar detection ($i = 1$), is

$$\gamma_k^{1,j} = \begin{bmatrix} r \\ \dot{r} \\ \varphi \end{bmatrix} + \mathbf{w}_k^{1,j}, \tag{10}$$

where, omitting the time dependency (k) and assuming the detection originates from vehicle l,

$$\begin{aligned} r &= \sqrt{(\xi_x^l - \xi_x^{\text{ego}} - \epsilon_x)^2 + (\xi_y^l - \xi_y^{\text{ego}} - \epsilon_y)^2} \\ \dot{r} &= v^l \cos(-(\psi^l - \psi^{\text{ego}}) + \varphi) - v^{\text{ego}} \cos(\varphi) \\ \varphi &= \tan^{-1}(\frac{\xi_y^l}{\xi_x^l}) - \psi^{\text{ego}} - \epsilon_\psi \end{aligned} \quad . \tag{11}$$

$(\epsilon_x, \epsilon_y, \epsilon_\psi)$ represent the sensor mounting position and orientation in the local ego vehicle coordinate system. The ego vehicle motion is considered a deterministic control signal and measurement noise, $\mathbf{w_k^{1,j}}$, is WGN with known covariance.

Measurement models for the ego vehicle sensors and the vision system are linear. The vision system sensor model ($i = 2$), using the identity matrix $\mathbf{I}$, becomes

$$\gamma_k^{2,j} = \begin{bmatrix} \mathbf{I}_{4\text{x}4} & 0 \end{bmatrix} \mathbf{r_k} + \mathbf{w_k^{2,j}}, \tag{12}$$

and the sensor model for internal sensors ($i = 3$) is written

$$\gamma_k^{3,j} = \begin{bmatrix} \mathbf{0}_{3\text{x}3} & \mathbf{I}_{3\text{x}3} \end{bmatrix} \mathbf{z_k^{\text{ego}}} + \mathbf{w_k^{3,j}}. \tag{13}$$

EVALUATION

It is hard to assess and quantify architecture performance. In this section we aim to demonstrate that the purposed fusion architecture is capable of producing a functional and processing efficient perception layer, using rapid prototyping tools. This is accomplished by evaluating the estimation accuracy of the reference implementation and by making predictions of the computational load when additional sensors are included.

Performance comparison and evaluation

We compare the estimates from the reference implementation with that of a sensor specific tracker used in active safety systems already on the market, relative ground truth

data. The aim is to show that our modular system is capable of performing on par with or better then a dedicated sensor tracker based on the same data, and that estimates are sensible relative ground truth data.

The comparison is limited to two scenarios. Scenario 1 involves the lead vehicle making a hard braking manoeuvre whereas in scenario 2, the host vehicle is accelerating towards the target vehicle. Sensor data from the described sensor setup is collected together with highly accurate *differential global positioning system* (DGPS) measurements of the position of both the host and the tracked vehicle. Using these DGPS position measurements we estimate the quantities needed to evaluate the tracking performance. Note that only the position is measured directly and that other states are calculated using the motion model derived from (8) and a UKF filter. Hence, the estimates of the other states can mainly serve as an indication of the magnitude of the true errors. The parameters in the implementation have been tuned to cover common scenarios, and the same setup is used for scenario 1 and scenario 2.

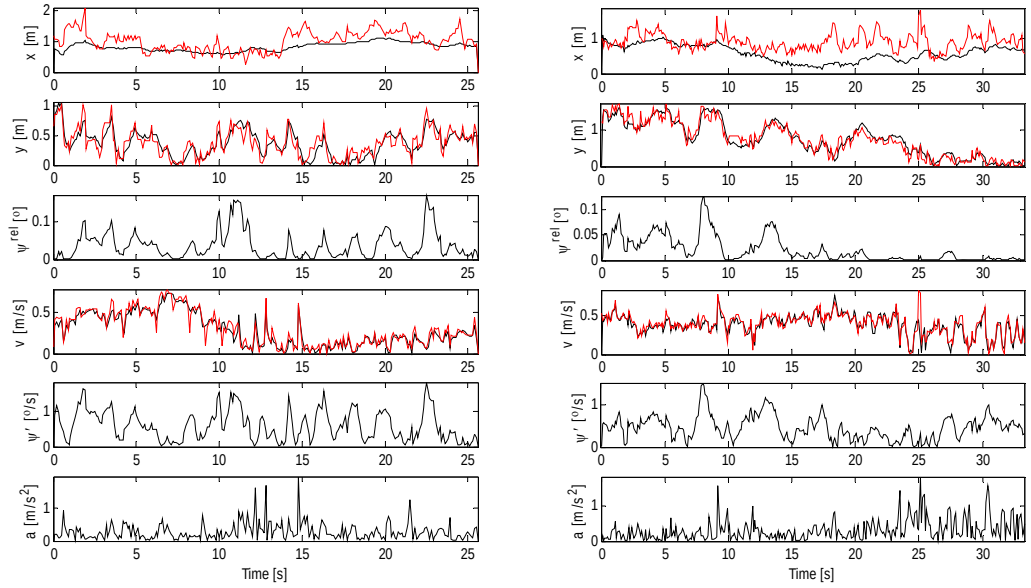


Figure 6: Evaluation of estimation accuracy for scenario 1 (left) and scenario 2 (right) using DGPS position measurements as reference. In the figures are the Euclidean error shown for all the states in the state vector for the target vehicle using the reference implementation (black) as well as for the sensor specific tracker (red), for applicable states.

Results, shown in Figure 6, indicate that a tracking system implemented using the proposed architecture is capable of delivering sensible estimates and that performance is comparable with that of a dedicated sensor tracker.

Processing time evaluation

The second aspect of our fusion architecture to be evaluated is it's scalability in terms of processing time for the multi-sensor system. As shown in Figure 5, adding a sensor does not affect the content of other modules. Hence, we can imitate a multi-sensor system by

independently duplicating both the prediction for the tracks and the filtering module for the external object sensor, i.e. simulating additional radar sensors. This way it is possible to simulate a larger number of simultaneous tracks, and multiple sensors.

In Table 1 the mean and maximum processing time for a system with 1 – 5 simulated external object sensors is shown. Simulated sensors deliver data simultaneously, which in corresponds to a "worst case" scenario. The processing times are measured running the algorithm in MATLAB on a laptop PC (Intel Pentium M 1,66 GHz). Initial tests indicate faster processing when running algorithms on a dedicated development hardware, i.e. xPC or dSPACE autobox (14, 15).

Table 1: Mean and maximum processing time for a system based on data from 1 - 5 sensors.

Processing time [s]	DF1	DF2	DF3	DF4	DF5
Mean	0,0049	0,008	0,0110	0,0141	0,0171
Maximum	0,0123	0,0191 (+55%)	0,0284 (+48%)	0,0367 (+29%)	0,0462 (+20%)

The measured processing times indicate that this implementation can run at least in 20 Hz incorporating data from 5 sensor similar to the forward looking radar.

CONCLUSIONS

We have proposed a fusion functional architecture that is modular in nature and support code and component re-use through different incarnations of the system. The architecture supports rapid prototyping tools, from which code suitable for production projects can be automatically generated. Sensor specific filter components with well defined input and output signals allow a high degree of adaptation while maintaining the general filtering framework. Several filtering techniques, such as the KF, UKF and EKF, are supported and can be used jointly in a single system. A reference implementation of the system based on standard methods run in real-time and performs on par with, or better than, a sensor specific tracker when using the same input data. We conclude that the proposed fusion architecture facilitates technology transfer and enables us to research and develop high-performing multi-sensor tracking systems for automotive safety systems in a structured, non-limiting, fashion.

REFERENCES

(1) A. Polychronopoulos, A. Amditis, U. Scheunert, and T. Tatschke, *Revisiting JDL model for automotive safety applications: the PF2 functional model,* in *The 9th International Conference on Information Fusion*. 2006: Florence.

(2) Y. Bar-Shalom and W. Dale Blair, *Multitarget-Multisensor Tracking Volume III: Applications and Advances*, Artech House, 2000.

(3) S. Blackman and R. Popoli, *Design and analysis of modern tracking systems*, Artech House, 1999.

(4) Y. Bar-Shalom, X. Rong Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation*. 2001, John Wiley & Sons, Inc.: New York, NY.

(5) B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman filter: particle filters for tracking applications*. 2004, Artech House: Norwood, MA.

(6) F. Bengtsson, *Models for tracking in automotive safety systems*. Institutionen för signaler och system, Chalmers tekniska högskola, 2008.

(7) L. Danielsson, *Tracking Theory for Preventive Safety Systems*. Institutionen för signaler och system, Signalbehandling, Chalmers tekniska högskola, 2008.

(8) Y. Bar-Shalom, "Update with out-of-sequence measurements in tracking: exact solution". *Aerospace and Electronic Systems, IEEE Transactions on*, 2002. **38**(3): p. 769-777.

(9) F. Bengtsson, L. Danielsson, and J. Gunnarsson, *D1.41 Definition of project demonstrators*. 2006: SEFS Deliverable D1.41.

(10) S.J. Julier and J.K. Uhlmann, "Unscented filtering and nonlinear estimation". *Proceedings of the IEEE*, 2004. **92**(3): p. 401 - 22.

(11) D.P. Bertsekas, "The auction algorithm for assignment and other network flow problems: a tutorial introduction ". *Interfaces*, 1990. **20**(4): p. 133 - 49.

(12) J. Gunnarsson, L. Svensson, L. Danielsson, and F. Bengtsson. *Tracking vehicles using radar detections*. in *Intelligent Vehicles Symposium, 2007 IEEE*. 2007.

(13) E.D. Dickmanns and A. Zapp, *A curvature-based scheme for improving road vehicle guidance by computer vision*, in *Proceedings of the SPIE Conference on Mobile Robots*. 1986.

(14) *dSPACE AutoBox*. Available from: http://www.dspace.com/ww/en/pub/home/products/hw/accessories/autobox.cfm.

(15) *xPC Target*. Available from: http://www.mathworks.com/products/xpctarget/.