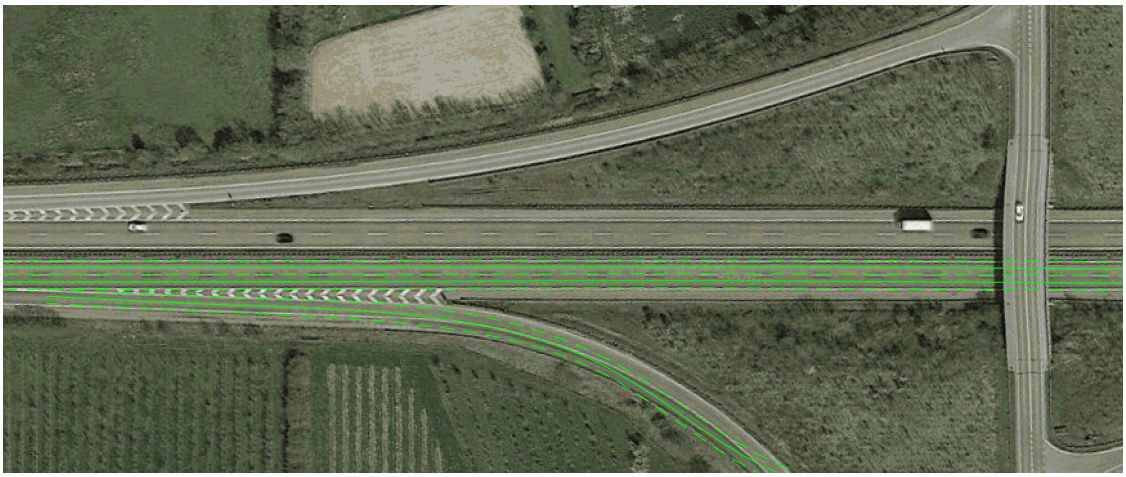

Bayesian Multi-Lane Road Geometry Estimation

Master's thesis in Systems, Control and Mechatronics

OSSIAN ERIKSSON

DEPARTMENT OF ELECTRICAL ENGINEERING

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023
www.chalmers.se

Master's thesis 2023

Bayesian Multi-Lane Road Geometry Estimation

OSSIAN ERIKSSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden 2023

Bayesian Multi-Lane Road Geometry Estimation
OSSIAN ERIKSSON

Supervisor: Lars Hammarstrand, Department of Electrical Engineering
Industry Supervisor: Junsheng Fu, Zenseact
Examiner: Lars Hammarstrand, Department of Electrical Engineering

Master's Thesis 2023
Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Estimated lane geometry overlayed on satellite image from Google Maps.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2023

Bayesian Multi-Lane Road Geometry Estimation
OSSIAN ERIKSSON
Department of Some Subject or Technology
Chalmers University of Technology

Abstract

Advanced Driver Assistance Systems (ADAS) are automotive technologies that improve driver comfort and road safety. Many such features, like adaptive cruise control and automatic lane keeping rely on knowledge about the geometry of the road around the ego vehicle. This purpose of this work is to design and evaluate a model based algorithm to estimate the center curves of highway lanes as well as on ramps and off ramps based only on internal vehicle sensors and a front camera system detecting lane markings. In particular, we design a novel road model with support for describing any number of lanes of the ego road, on ramps and off ramps within some range of the ego vehicle. To evaluate whether this road model is suitable for tracking with a Bayesian filter, a multiple hypothesis coupled multi object tracking filter is designed specifically for the chosen road model. Multiple hypothesis are used to deal with data association for detections of lane markers and coupling lanes is useful for modeling parallel lanes. The filter additionally deals with object spawning and death in a way inspired by multi-Bernoulli mixture filters by giving each lane within each hypothesis a probability of existence. The filter is evaluated on highway driving scenarios provided by the Zenseact company and is demonstrated to track the ego lane and parallel lanes well, but sometimes struggles to distinguish main road lanes from on or off ramps since the filter only knows about lane markings and not e.g. road edges or guard rails. The current filter implementation is also computationally intensive and does not reliably run in real time on consumer hardware, but demonstrates the potential of the road model used to track multiple lanes and branching roads.

Keywords: Multi-Lane Road Geometry, Sensor Fusion, Bayesian Filtering, Coupled Multi-Object Tracking.

Acknowledgements

I would like to thank my academic supervisor, Lars Hammarstrand as well as Yuxuan Xia, both at the Department of Electrical Engineering, Chalmers, for many ideas, discussions and useful feedback on my work and writings. I would also like to thank my industry supervisor, Junsheng Fu at Zenseact for always being supportive and helping in many practical aspects of my work. Additionally, I thank Sachin Madhusudhana, Valentin Kovbas, Markus Eriksson, and the rest of team Skalman at Zenseact for support and discussions.

Ossian Eriksson, Gothenburg, June 2023

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

ADAS	Advanced Driver Assistance Systems
EKF	Extended Kalman Filter
FP	False Positive
GM-PHD	Gaussian Mixture Probability Hypothesis Density
IMU	Inertial Measurement Unit
LSTM	Long Short-Term Memory
MBM	Multi-Bernoulli Mixture
MHT	Multiple Hypothesis Tracker
MOT	Multi-Object Tracking
NEES	Normalized Estimation Error Squared
PMBM	Poisson Multi-Bernoulli Mixture
PPP	Poisson Point Process
RANSAC	RANdom SAmple Consensus
RFS	Random Finite Set
RMS	Root Mean Squared
ROI	Region Of Interest
TP	True Positive
ZOD	Zenseact Open Dataset

Contents

List of Figures

List of Tables

1

Introduction

As computer hardware in cars continue to progress, it allows for an increasing amount of effort to be put into the development of Advanced Driver Assistance Systems (ADAS). Features such as driver drowsiness detection and automatic breaking are becoming ever more important to drivers and are in some cases starting to be required features in new sold cars [1].

Many active safety systems such as adaptive cruise control and automatic lane keeping are reliant on having some knowledge of the road surrounding the vehicle. Thus, increasing the accuracy and redundancy in road estimation systems is therefore crucial to improving driver safety and comfort. On the other hand, faulty road geometry estimation could have potentially fatal consequences for people on the road.

There are multiple approaches to road geometry estimation, such as using an offline [2] or crowd sourced map [3], or using on board sensors such as cameras to get a live view of the space around the vehicle [4]–[7]. Having multiple approaches to the same problem is good for redundancy. For map based descriptions of the road geometry, it is natural to include all lanes surrounding the vehicle. However, for on-board sensor systems many algorithms such as the ones discussed in [5], [8] are only concerned with estimating the ego vehicle lane, sometimes also including directly neighboring lanes [9]. The aim of this work is to develop a model based technique for estimating an arbitrary number of lanes on the ego road in highway scenarios, as well as any merges or diverges from the main road.

1.1 Structure of This Work

This works purpose is to investigate a Bayesian approach to multi-lane tracking in highway scenario. To this end, the Zenseact company has provided a closed data set of highway drives in various European countries. From this data set, the scope of this work is limited to focus on two groups of data: Internal vehicle signals such as speed and angular velocity, and lane marking detections originating from an algorithm processing images from a front facing camera. The desired output of the proposed algorithm is a representation of the centers of the lanes of the ego road and merges and diverges up to 200 m in front of the ego vehicle.

To this end, a state representation of highway road is first constructed. Also, a motion model describing the motion of the ego vehicle over the road is presented

and a measurement model describing the expected output from the lane marking detection algorithm given parameters of the road are needed. The model chosen is a coupled one, where all lanes are represented by a joint state vector and covariance matrix. Second, a multiple hypothesis multi object tracking filter is designed to track the state. This filter has multiple interesting features which arise from the specific tracking scenario, for example the way that lane markings detections are not connected to the tracked objects (lanes) directly, but to borders between these objects. Finally, the designed approach is evaluated on the data set provided by Zenseact.

This report is split into five chapters. In this first chapter the problem is introduced and related work is reviewed. In the theory chapter, relevant theory on Bayesian filters, object tracking in clutter and multi object tracking are reviewed. The methods chapter describes the design of the road model, motion and measurement models as well as the filter algorithm. Finally, evaluation results are presented in the Results chapter, and the final chapter discusses the implications of the results and areas of possible improvement.

1.1.1 Notation

Some common typesetting rules used throughout this report are as follows: Random variables are displayed in upright sans serif fonts. Vector variables are displayed as lower case boldface italic symbols, and matrix variables as uppercase upright letters. Sets are denoted by calligraphic uppercase letters. For example, x is a scalar random variable and its realization x. Additionally, $\mathbf{x}$ is a vector random variable and its realization $\boldsymbol{x}$. Furthermore, $\mathbf{X}$ is a matrix, and $\mathcal{X}$ a set. Random matrices are not used in this work, but random finite sets are and are denoted like $\mathbf{X}$. Finally, a sequence of values $(x_i)_{i=1}^{N}$ may alternatively be written as $x_{1:N}$.

1.2 Related Work

Road geometry estimation methods based on on board sensors can be roughly split into three categories: Those based fully on Bayesian filters such as the Kalman filter, those using other methods of estimation such as polynomial fitting or neural networks and those using a hybrid of Bayesian filters and other methods. Additionally, the methods can be categorized by whether multiple lanes or only the ego lane geometry is estimated, and by whether the algorithm tracks lane marking lines or lane center lines.

A purely Bayesian approach to estimating the ego lane center is presented in [5]. Here the curve of the lane is parameterized, and those parameters estimated by means of a Kalman filter based tracker. Input to the filter is left and right lane marking detections as well as static radar detections and dynamic objects detected by a camera-radar fusion system. Multiple hypothesis are introduced in order to deal with the static radar observations. This work is built upon in [4], which extends the scope of [5] by also tracking on and off ramps in highway scenarios. To this end, the

multi object tracking GM-PHD [10] and PMBM [11] filters are employed, however the filters presented do not also track lanes adjacent to the ego lane.

A novel hybrid Bayesian approach to tracking multiple lane marking lines of the ego road and also supporting on and off ramps is presented in [12]. This method is interesting and illustrative in a couple of ways. Firstly, it tracks lane marking lines which is a simpler tracking problem than tracking lane centers since the marking lines can be detected directly and are more easily defined, whereas lane centers can often only be described indirectly by multiple marking lines. It is not always easy to extract lane centers from a representation of the lane marking lines if one does not know which lines are supposed to define the left and right border respectively of each lane. Second, it identifies the concept of parallel lanes or lane markings as a topic of consideration. Lanes are often, but not always parallel, and in cases where they are this can be used to improve the road geometry estimation. In [12] each marking line is independently tracked by and extended Kalman filter, and as a post process step lane markings believed to be close to parallel are adjusted to be more in line with each other. Other methods taking alternate approaches to lane parallelism are [7] where parallelism is always assumed, and [13] where lane parallelism is not considered.

2

Theory

2.1 Bayesian State Estimation

What is the true state of a control system? This is the question which the methods of state estimation set out to answer. One may be able observe some properties of the system, like the pressure of gas in a container, without explicitly knowing other properties, like the temperature of the gas. However, knowing the pressure should let us narrow down the possible range of temperatures, even if we do not know all properties of the gas and even if we know that the pressure sensor is not perfectly reliable in that it adds noise to the measurements. Thus between some uncertainty in the inner workings of the control system and the uncertainty in the measurements, the state estimation game becomes one of probability. This section is dedicated to introducing the concept of Bayesian State Estimation as explained by [14].

There are multiple interpretations of the concept of probability, of which Bayesian probability is one. In this framework, all events have a prior probability of occurring which allows for expressing the probability of an event even when none or finitely many experiments have been done to test the ratio between positive and negative outcomes. Crucial to Bayesian probability is Bayes theorem

$$p(\mathsf{a}|\mathsf{b})p(\mathsf{b}) = p(\mathsf{b}|\mathsf{a})p(\mathsf{a}) \tag{2.1}$$

which may be derived from $p(\mathsf{a}, \mathsf{b}) = p(\mathsf{a}|\mathsf{b})p(\mathsf{b})$.

For the purpose of state estimation, we assume our control system is described by a discrete state space model. In each time step k we typically represent the state estimate by a random variable $\mathbf{x}_k$ and receive measurements also in the form of a random variable $\mathbf{y}_k$. A Bayesian network representation of a state space model with the Markov property is shown in Figure 2.1. The Markov property means that future states are influenced only by the current state, and not by any past states. The network of Figure 2.1 visualizes this fact, along with the assumption that measurements only depend on the current state. These two properties can be written as

$$p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{y}_{1:k-1}) = p(\mathbf{x}_k|\mathbf{x}_{k-1}) \tag{2.2}$$

and

$$p(\mathbf{y}_k|\mathbf{x}_k, \mathbf{y}_{1:k-1}) = p(\mathbf{y}_k|\mathbf{x}_k). \tag{2.3}$$

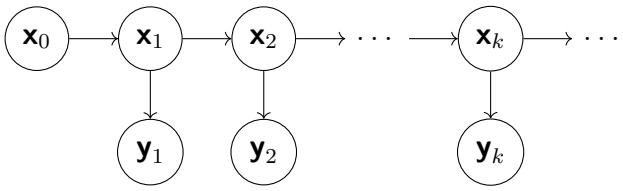

Figure 2.1: Bayesian network description of a state space model.

The goal of recursive Bayesian state estimation is to express the posterior state density $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ given a prior $p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})$ (and measurements from time 1 to k). However, to find $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ we first seek an expression for $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$. Using (2.2) and the law of total probability we arrive at what is known as the Chapman-Kolmogorov equation

$$p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) = \int p(\mathbf{x}_k|\boldsymbol{x}_{k-1})p(\boldsymbol{x}_{k-1}|\mathbf{y}_{1:k-1})\,\mathrm{d}\boldsymbol{x}_{k-1}\,, \tag{2.4}$$

where we have made use of equation (2.2).

Further, (2.3) in combination with Bayes theorem (2.1) gives

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_k|\mathbf{x}_k, \mathbf{y}_{1:k-1})p(\mathbf{x}_k|\mathbf{y}_{1:k-1})}{p(\mathbf{y}_k|\mathbf{y}_{1:k-1})} \propto p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1}). \tag{2.5}$$

Here $p(\mathbf{y}_k|\mathbf{x}_k)$ is known as the measurement likelihood. Since we are only interested in the density with respect to $\mathbf{x}_k$, we may view the denominator as a constant normalization factor and discard it. It can later be recreated by enforcing that $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ should integrate to one.

Thus, the basic building blocks of a generic Bayesian state estimator have been laid down. In time step k given a prior density $p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})$ one first performs what is known as the prediction step of the estimator by applying the Chapman-Kolmogorov equation (2.4) to obtain $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$. From a known likelihood, one next performs what is known as the update step by applying (2.5) to obtain $p(\mathbf{x}_k|\mathbf{y}_{1:k})$. This procedure is then repeated recursively for time steps $k+1$, $k+2$ and so on. For the initial time step, an expression for the initial prior $p(\mathbf{x}_0)$ needs to be provided.

2.2 The Kalman Filter

The Kalman filter is a Bayesian state estimator specialized for linear and Gaussian systems [15]. A linear Gaussian state space model is given by

$$\mathbf{x}_k = \mathbf{A}_k\mathbf{x}_{k-1} + \mathbf{B}_k\boldsymbol{u}_k + \mathbf{q}_k \tag{2.6}$$

$$\mathbf{y}_k = \mathbf{C}_k\mathbf{x}_k + \mathbf{D}_k\boldsymbol{u}_k + \mathbf{r}_k \tag{2.7}$$

where $\mathbf{A}_k$, $\mathbf{B}_k$, $\mathbf{C}_k$, $\mathbf{D}_k$ are constant matrices, $\boldsymbol{u}$ is a control signal and $\mathbf{q}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k)$, $\mathbf{r}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$, $\mathbf{x}_0 \sim \mathcal{N}(\hat{\boldsymbol{x}}_{0|0}, \mathbf{P}_{0|0})$ are independent random vector variables. Here, (2.6) and (2.7) are known as the motion and measurement model

respectively, while $\mathbf{q}_k$ and $\mathbf{r}_k$ are known as the process noise and measurement noise [14].

It can be shown [14] that the state estimates will also be Gaussian distributed for all time steps, i.e.

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) \sim \mathcal{N}(\hat{\boldsymbol{x}}_{k|k}, \mathbf{P}_{k|k}) \tag{2.8}$$
$$p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) \sim \mathcal{N}(\hat{\boldsymbol{x}}_{k|k-1}, \mathbf{P}_{k|k-1}), \tag{2.9}$$

where $\hat{\boldsymbol{x}}_{k|k}, \mathbf{P}_{k|k}$ and $\hat{\boldsymbol{x}}_{k|k-1}, \mathbf{P}_{k|k-1}$ are the means and covariance matrices of the prior and predicted state densities respectively. As such, the Kalman filter is a conjugate prior filter, meaning the prior and posterior densities are of the same family, Gaussian in this case. For linear and Gaussian systems, it can also be shown [14] that the Bayesian state estimation equations discussed in Section 2.1 specializes to

Prediction:

$$\hat{\boldsymbol{x}}_{k|k-1} = \mathbf{A}_k\hat{\boldsymbol{x}}_{k-1|k-1} + \mathbf{B}_k\boldsymbol{u}_k \tag{2.10a}$$
$$\mathbf{P}_{k|k-1} = \mathbf{A}_k\mathbf{P}_{k-1|k-1}\mathbf{A}_k^\top + \mathbf{Q}_k \tag{2.10b}$$

Update:

$$\hat{\boldsymbol{y}}_k = \mathbf{C}_k\hat{\boldsymbol{x}}_{k|k-1} + \mathbf{D}_k\boldsymbol{u}_k \tag{2.10c}$$
$$\mathbf{S}_k = \mathbf{C}_k\mathbf{P}_{k|k-1}\mathbf{C}_k^\top + \mathbf{R}_k \tag{2.10d}$$
$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{C}_k^\top\mathbf{S}_k^{-1} \tag{2.10e}$$
$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \mathbf{K}_k(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k) \tag{2.10f}$$
$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_{k|k-1}. \tag{2.10g}$$

2.2.1 The Extended Kalman Filter

The extended Kalman filter (EKF) is a variation of the Kalman Filter adapted for nonlinear state space models given by

$$\mathbf{x}_k = f_k(\mathbf{x}_{k-1}, \boldsymbol{u}_k) + \mathbf{q}_k \tag{2.11}$$
$$\mathbf{y}_k = h_k(\mathbf{x}_k, \boldsymbol{u}_k) + \mathbf{r}_k. \tag{2.12}$$

The process and measurements noises $\mathbf{q}_k$ and $\mathbf{r}_k$ are as in the ordinary Kalman Filter. In every time step the motion and measurement models are linearized around the mean of the prior. The ordinary Kalman filter equations are the applied to the linearization. This results in an approximate solution to the state estimation problem where all densities are assumed to be Gaussian. According to [14], the full EKF equations are

Prediction:

$$\hat{\boldsymbol{x}}_{k|k-1} = f_k(\hat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{u}_k) \tag{2.13a}$$
$$\mathbf{P}_{k|k-1} = \mathbf{F}_k\mathbf{P}_{k-1|k-1}\mathbf{F}_k^\top + \mathbf{Q}_k \tag{2.13b}$$

Update:

$$\hat{\boldsymbol{y}}_k = h_k(\boldsymbol{x}_{k|k-1}, \boldsymbol{u}_k) \tag{2.13c}$$
$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^\top + \mathbf{R}_k \tag{2.13d}$$
$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^\top \mathbf{S}_k^{-1} \tag{2.13e}$$
$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \mathbf{K}_k(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k) \tag{2.13f}$$
$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{H}_k \mathbf{P}_{k|k-1} \tag{2.13g}$$

where

$$\mathbf{F}_k = \left.\frac{\partial f_k}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{u}=\boldsymbol{u}_k}, \quad \mathbf{H}_k = \left.\frac{\partial h_k}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_{k|k-1}, \boldsymbol{u}=\boldsymbol{u}_k}. \tag{2.14}$$

2.3 Tools of Object Tracking in Clutter

Filters such as the Kalman filter and its variants are suitable for tracking the state of a single object, such as a car or other vehicle, in environments with no clutter [14], i.e. where every measurement is known to originate from the tracked object. However, many real world scenarios involve sensors which introduce clutter, or sometimes doesn't detect the real object [16].

A typical example to illustrate clutter and the possibility of not detecting the tracked object is the use of a radar sensor to track a single airplane. Every time the radar is polled it returns a variable length list of distance angle detection pairs made since the last time step. Even given that we know for sure that only one airplane is present within the field of view of the radar, the sensor when polled may yield zero, one, or multiple detections due to disturbances. This stands in contrast to simple sensors such as IMUs and encoders which always return a fixed size data structure when polled. Those radar detections which are due to disturbances and not associated to the object we desire to track are called clutter. Given a set of detections from the radar, it is possible that none of them correspond to the true object. This would obviously be the case if the radar did not detect anything at the current time step, and could for example be the result of the object holding an unfavorable angle to the radar, or that something is blocking the view from the radar to the object. In order to make use of such sensors as the radar in a tracking system, methods of handling clutter and missed detections need to be formulated.

2.3.1 Gating

One way to deal with clutter is to simply discard detections that are unlikely to originate from the plane given the prior. This is known as gating, and forms an important part of clutter rejection measures in many more complex filters [16].

For single object tracking in scenarios with low clutter intensities and high probability of detecting the object, using a simple Kalman filter in combination with gating may be enough to achieve good tracking performance [14]. To illustrate the concept of gating, we thus start from an ordinary Kalman filter. Now however, at each time

step the object we desire to track may sometimes not be detected, and sometimes we receive extra measurements due to clutter.

At the start of the time step the ordinary Kalman filter update step is performed. Then, assuming that the object was in fact detected at time k, the Kalman filter gives us the likelihood of measuring data $\boldsymbol{y}_k$ as

$$p(\boldsymbol{y}_k|\boldsymbol{x}_k) = \mathcal{N}(\boldsymbol{y}_k; \hat{\boldsymbol{y}}_k, \mathbf{S}_k) = \frac{1}{\sqrt{(2\pi)^d|\mathbf{S}_k|}} \exp\Big(-\frac{1}{2}(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k)^\top \mathbf{S}_k^{-1}(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k)\Big) \quad (2.15)$$

where $d = \dim \boldsymbol{y}_k$. One may then discard measurements where $p(\boldsymbol{y}_k|\boldsymbol{x}_k) < \Gamma$ where Γ is some threshold. However, it is often better to drop the first factor of (2.15) when performing gating such that the gate does not discard measurements because of uncertainty in the prior state [17]. We are left with the condition

$$\exp\Big(-\frac{1}{2}(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k)^\top \mathbf{S}_k^{-1}(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k)\Big) < \Gamma \quad (2.16)$$

or equivalently

$$d_{\mathrm{M}}(\boldsymbol{y}_k) = \sqrt{(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k)^\top \mathbf{S}_k^{-1}(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k)} < \Gamma \quad (2.17)$$

where $d_{\mathrm{M}}(\boldsymbol{y}_k)$ is known as the Mahalanobis distance [17]. If there are no remaining measurements after gating, no update step is performed at time k. This corresponds to the sensor not detecting the object at time k. Otherwise, one of the remaining measurements corresponds to the real object. In the case with low clutter intensity, you would expect there to only be one remaining measurement in most timesteps. In this case, it may be sufficient to perform the Kalman filter update with the most likely of the remaining measurements, discarding the rest.

2.3.2 The Poisson Point Process

Using gating on its own to deal with clutter may be feasible in scenarios with low clutter intensities, however as the amount of clutter increases it becomes more important to model the clutter in order to reject it with higher accuracy. One commonly used model of clutter is the Poisson Point Process (PPP). In this model, the clutter measurements are independent and identically distributed. At each time step, the number of clutter measurements n in the volume $\mathcal{V}$ surveillance by our sensors is Poisson distributed with expected value

$$\bar{\lambda} = \int_{\mathcal{V}} \lambda(\boldsymbol{y})\, \mathrm{d}\boldsymbol{y} \quad (2.18)$$

where $\lambda(\mathbf{y})/\bar{\lambda}$ is the probability density function of each of the clutter measurements [18]. Here $\lambda(\mathbf{y})$ is known as the intensity of the Poisson point process.

Given that we received $\mathsf{n} = n$ clutter measurements

$$\mathbf{Y}^{\mathrm{C}} = \big\{\mathbf{y}_1^{\mathrm{C}}, \ldots, \mathbf{y}_n^{\mathrm{C}}\big\}, \quad (2.19)$$

the joint probability density function of the set of those measurements would be

$$p\big(\mathbf{Y}^{\mathrm{C}}\big|n\big) = n! \prod_{j=1}^{n} p(\mathbf{y}_j^{\mathrm{C}}) = \frac{n!}{\bar{\lambda}^n} \prod_{j=1}^{n} \lambda(\mathbf{y}_j^{\mathrm{C}}), \quad (2.20)$$

where we note the addition of the $n!$ factor to account for the disregarding of the ordering of the clutter measurements. Since n itself is Poisson distributed with parameter $\bar{\lambda}$,

$$p(n) = \frac{\bar{\lambda}^n}{n!} \mathrm{e}^{-\bar{\lambda}}, \tag{2.21}$$

the complete probability density of measuring the set $\mathbf{Y}^{\mathrm{C}}$ is [18]

$$p\left(\mathbf{Y}^{\mathrm{C}}\right) = p\left(\mathbf{Y}^{\mathrm{C}} \middle| n\right) p(n) = \mathrm{e}^{-\bar{\lambda}} \prod_{j=1}^{n} \lambda(\mathbf{y}_j^{\mathrm{C}}). \tag{2.22}$$

Sets such as $\mathbf{Y}^{\mathrm{C}}$ where $\mathsf{n} = \left|\mathbf{Y}^{\mathrm{C}}\right| < \infty$ is a random variable are generally known as Random Finite Sets (RFS) [18]. In particular, $\mathbf{Y}^{\mathrm{C}}$ is a PPP RFS.

2.3.3 The Bernoulli Process

In addition to introducing clutter measurements, an imperfect sensor may also not detect the tracked object at each time step. This property is nicely modeled with a Bernoulli Process. Using the Bernoulli Process model, at time step k, the object is detected with probability ρ and undetected with probability $1 - \rho$. Given that the object is detected, it is distributed according to some distribution with probability density function $g(\mathbf{y})$ [11].

The set of detections originating from the object is also a random finite set $\mathbf{Y}^{\mathrm{D}}$ with probability density

$$p(\mathbf{Y}^{\mathrm{D}}) = \begin{cases} \rho\, g(\boldsymbol{y}^{\mathrm{D}}) & \text{if } \mathbf{Y}^{\mathrm{D}} = \left\{\boldsymbol{y}^{\mathrm{D}}\right\} \\ 1 - \rho & \text{if } \mathbf{Y}^{\mathrm{D}} = \emptyset \\ 0 & \text{otherwise.} \end{cases} \tag{2.23}$$

2.4 Gaussian Sum Filtering

The Gaussian sum filter is a filter capable of tracking a single object in scenarios with high clutter intensities. In the filter, the state densities are represented as as superposition of multiple Gaussian densities [19]. For example, the prior is parameterized by

$$\left(\langle w_{k-1|k-1}^h, \hat{\boldsymbol{x}}_{k-1|k-1}^h, \mathbf{P}_{k-1|k-1}^h \rangle\right)_{h=1}^{H_{k-1|k-1}} \tag{2.24}$$

where the $w_{k-1|k-1}^h$, $\hat{\boldsymbol{x}}_{k-1|k-1}^h$ and $\mathbf{P}_{k-1|k-1}^h$ are the weight, mean and covariance of the h:th component in the Gaussian mixture [20]. One way to interpret the prior is to view the mixture components as hypotheses, where within each hypothesis the density is Gaussian. As such, the Gaussian sum filter is an example of a multiple hypothesis tracker (MHT).

In the prediction step of the linear Gaussian sum filter, the number of hypothesis stays constant, i.e. $H_{k|k-1} = H_{k-1|k-1}$. The weights are also unchanged, $w_{k|k-1} = w_{k-1|k-1}$, while the ordinary Kalman filter prediction step is applied to each hypothesis separately to acquire $\hat{\boldsymbol{x}}_{k|k-1}^h$ and $\mathbf{P}_{k|k-1}^h$ [20].

For the update step, one is given a set of J measurement vectors,

$$\mathcal{Y}_k = \{\boldsymbol{y}_{k,1}, \ldots, \boldsymbol{y}_{k,J}\}. \tag{2.25}$$

Now, assuming point object tracking, that is there is at most one detection corresponding to the real object, there are $J+1$ different possible cases: Either one of the measurements $\boldsymbol{y}_{k,j}$ correspond to the real measurement and the rest are clutter, or all measurement vectors are clutter.

To perform the update step, for each prior hypothesis h, each of these $J+1$ so called data association hypotheses are considered. In total, $H_{k|k} = H_{k|k-1}(J+1)$ new hypothesis will be created from these combinations of possibilities. For the case where measurement $\boldsymbol{y}_{k,d}$ corresponds to the real object in hypothesis h, the weight of the resulting hypothesis is

$$w_{k|k}^{h'} \propto w_{k|k-1}^{h} P_k^{\mathrm{D}} \mathcal{N}(\boldsymbol{y}_{k,d}; \mathbf{C}_k \hat{\boldsymbol{x}}_{k|k-1}^{h}, \mathbf{S}_k^{h}) f_k^{\mathrm{C}}(\mathcal{Y}_k \setminus \{\boldsymbol{y}_{k,d}\}) \tag{2.26}$$

where P_k^{D} is the probability of detecting the object in time step k, and f_k^{C} is the clutter probability density function, and the hypothesis mean and covariance may be updated according to the standard Kalman filter update with $\boldsymbol{y}_{k,d}$ as measurement vector. For the case where no measurement correspond to the real object in hypothesis h, the weight of the resulting hypothesis is given in [20] as

$$w_{k|k}^{h'} \propto w_{k|k-1}^{h}(1 - P_k^{\mathrm{D}}) f_k^{\mathrm{C}}(\mathcal{Y}_k), \tag{2.27}$$

while the hypothesis mean and covariance remain the same. The updated weight are then normalized to sum to one in order to account for the denominator in (2.5).

With PPP clutter, (2.26) and (2.27) become

$$w_{k|k}^{h'} \propto w_{k|k-1}^{h} P^{\mathrm{D}} \mathcal{N}(\boldsymbol{y}_{k,d}; \mathbf{C}_k \hat{\boldsymbol{x}}_{k|k-1}^{h}, \mathbf{S}_k^{h}) \mathrm{e}^{-\bar{\lambda}} \prod_{j \neq d} \lambda(\boldsymbol{y}_{k,j}). \tag{2.28}$$

and

$$w_{k|k}^{h'} \propto w_{k|k-1}^{h}(1 - P_k^{\mathrm{D}}) \mathrm{e}^{-\bar{\lambda}} \prod_{j} \lambda(\boldsymbol{y}_{k,j}). \tag{2.29}$$

respectively. When implementing the filter, it is often more efficient to divide (2.28) and (2.29) with $\mathrm{e}^{-\bar{\lambda}} \prod_j \lambda(\boldsymbol{y}_{k,j})$, since this does not affect the normalized weights.

2.5 Pruning, Merging and Capping

When making a naive implementation of a MHT such as a basic Gaussian sum filter, one quickly runs into the problem of an exploding number of hypothesis. In the update step of the filter discussed in Section 2.4 the number of hypothesis is multiplied by $J+1$ every time step where J is the number of measurements. To combat this exponential growth in computational complexity some approximations are introduced to make filter implementations more tractable. The techniques discussed here, pruning, merging and capping, are general techniques which are used

in many different MHT algorithms for reducing the number of hypothesis, often in combination with each other [21].

Pruning is the process of discarding those hypothesis which are relatively unlikely. If for example the posterior consists of $H_{k|k}$ hypotheses with weights $w_{k|k}^h$, one simply discards the hypotheses which does not fulfill

$$w_{k|k}^h > \Gamma^{\text{prune}} \max_{h'} w_{k|k}^{h'}, \tag{2.30}$$

meaning the most probable hypothesis needs to be at least $1/\Gamma^{\text{prune}}$ times more likely then hypothesis h for h to survive. After pruning, the remaining weights are renormalized.

Merging is the process of summarizing two or more similar hypothesis in a single hypothesis [22]. First, similar hypothesis are identified using some criteria. Then these hypothesis are merged into a single hypothesis. For example, with Gaussian hypothesis $\mathcal{H} = \{h_1, \dots, h_H\}$ with weights w^h, mean $\hat{\boldsymbol{x}}^h$ and covariance $\mathbf{P}^h$ merging consists of approximating this Gaussian mixture with a single Gaussian with mean $\hat{\boldsymbol{x}}^{\text{merge}}$, and covariance $\mathbf{P}^{\text{merge}}$. This can for example be done with moment matching [22], resulting in

$$\hat{\boldsymbol{x}}^{\text{merge}} = \sum_{h\in\mathcal{H}} w^h \hat{\boldsymbol{x}}^h, \quad \mathbf{P}^{\text{merge}} = \sum_{h\in\mathcal{H}} w^h \Big(\mathbf{P}^h + (\hat{\boldsymbol{x}}^h - \hat{\boldsymbol{x}}^{\text{merge}})(\hat{\boldsymbol{x}}^h - \hat{\boldsymbol{x}}^{\text{merge}})^\top\Big). \tag{2.31}$$

The weight of the merged hypothesis is the sum of the weights of the individual hypotheses,

$$w^{\text{merge}} = \sum_{h\in\mathcal{H}} w^h. \tag{2.32}$$

Finally, capping is perhaps the most straightforward way of reducing the number of hypothesis. If the number of hypotheses exceeds Γ^{cap}, we discard all but the Γ^{cap} most likely hypotheses. The remaining weights are then renormalized. Capping guarantees that the number of hypotheses will always be bounded by Γ^{cap}.

2.6 The Multi-Bernoulli Mixture Filter

The Multi-Bernoulli Mixture (MBM) filter is an algorithm for tracking multiple separate objects in the presence of clutter, a so called Multi-Object Tracking (MOT) problem. So far, the examples discussed in this work have mostly focused on single object tracking, like tracking a single airplane using radar. A natural extension is to also consider the possibility of multiple object, and an unknown number of objects. This section is dedicated to a brief explanation of the MBM filter as described by [23].

The filter represents the state as a superposition of hypothesis, where each hypothesis is a union of Bernoulli processes. Each such Bernoulli process may be interpreted as one possible object track. For Gaussian Bernoulli densities, the prior is a can be represented by

$$\left(\left\langle w_{k-1|k-1}^h, \left(\left\langle \rho_{k-1|k-1}^{h,i}, \hat{\boldsymbol{x}}_{k-1|k-1}^{h,i}, \mathbf{P}_{k-1|k-1}^{h,i} \right\rangle\right)_{i=1}^{N_{k-1|k-1}^h} \right\rangle\right)_{h=1}^{H_{k-1|k-1}}. \tag{2.33}$$

Here $\rho^{h,i}$ are the Bernoulli existence probabilities. In each hypothesis h the expected number of objects is the sum of $\rho^{h,i}$ over i. Here each hypothesis represents a Bernoulli mixture RFS, that is a union of Bernoulli RFS. Within hypothesis h each Bernoulli RFS i contains an element with probability $\rho^{h,i}$, and if the element exists it is distributed according to $\mathcal{N}(\hat{\boldsymbol{x}}^{h,i}, \mathbf{P}^{h,i})$.

The prediction step of the MBM filter is very similar to Gaussian sum filtering in that the number of hypothesis and hypothesis weights are unchanged, while the ordinary Kalman filter prediction step is applied to each object of each prior hypothesis separately. However, the MBM also includes two new features as part of the prediction step, both connected to the fact that the number of objects in each hypothesis may change. First, some of the previously tracked objects may have disappeared between the previous time step and the current one. This property is called object death. Let the probability that each object survives from one time step to the next be P^{S}. We call this quantity the probability of survival. Then we may update the existence probabilities of the previously tracked objects as

$$\rho_{k|k-1}^{h,i} = P^{\mathrm{S}} \rho_{k|k-1}^{h,i}. \tag{2.34}$$

Additionally, some new objects may have appeared from the previous time step to the current one. This is handled by adding some new components to the Bernoulli mixture of each hypothesis where we believe new objects may have appeared. This is called object birth, and the number of birth components N^{b} as well as their density and probability of existence can be arbitrarily chosen to best represent the birth density of the specific scenario one is designing a filter for. Thus the number of Bernoulli components is increased in the prediction step, $N_{k|k-1}^{h} = N_{k-1|k-1}^{h} + N^{\mathrm{b}}$.

The update step of the MBM filter is also similar to Gaussian sum filtering. In its most basic form, for every hypothesis one considers every possible valid combination of assigning measurement data to either one of the objects in the Bernoulli mixture, or to clutter. With the point object assumption, a data association is only valid if no object has multiple measurements associated to it. Consider for example the prior hypothesis h and one of the possible data associations under it. Assume also that we use a PPP clutter model with intensity λ^{C}. We are given a set of J measurement vectors

$$\mathcal{Y}_k = \{\boldsymbol{y}_{k,1}, \ldots, \boldsymbol{y}_{k,J}\}. \tag{2.35}$$

Let $i(j) = 0$ if measurement j was associated to clutter and the index of the Bernoulli mixture component associated to measurement j otherwise. Then we may add one hypothesis h' to the posterior where the mean and covariance of each object is either updated independently according to the standard Kalman filter update if it had a measurement associated to it, in which case we say it was detected, or copied straight from the prior if no measurement was associated to it. The existence probability $\rho_{k|k}^{h',i}$ is set according to

$$\rho_{k|k}^{h',i} = 1 \qquad \text{If object } i \text{ was detected} \tag{2.36}$$

$$\rho_{k|k}^{h',i} = \frac{\rho_{k|k-1}^{h,i} P^{\mathrm{D}}}{1 - \rho_{k|k-1}^{h,i} + \rho_{k|k-1}^{h,i} P^{\mathrm{D}}} \qquad \text{If object } i \text{ was not detected,} \tag{2.37}$$

where P^{D} is the probability of detecting each object as introduced for Gaussian sum filtering in section 2.4. The number of objects will be the same as in the prior, i.e. $N^{h'}_{k|k} = N^{h}_{k|k-1}$. Finally, the weight of the resulting hypothesis is set according to

$$w^{h'}_{k|k} \propto w^{h}_{k|k-1} \prod_{i \in \mathcal{I}^{\mathrm{D}}} P^{\mathrm{D}} \rho^{h,i}_{k|k-1} \mathcal{N}(\boldsymbol{y}_{k,j}; \mathbf{C}_k \hat{\boldsymbol{x}}^{h,i}_{k|k-1}, S^{h,i}_k) \times \prod_{i \in \mathcal{I} \setminus \mathcal{I}^{\mathrm{D}}} \left(1 - P^{\mathrm{D}} \rho^{h,i}_{k|k-1}\right) \prod_{j \in \mathcal{J}^{\mathrm{C}}} \lambda^{\mathrm{C}}(\boldsymbol{y}_{k,j}) \tag{2.38}$$

where $\mathcal{I} = \{1, \ldots, N^{h'}_{k|k}\}$ are the indices of all objects, $\mathcal{I}^{\mathrm{D}} = \{i(j) : j \in \mathcal{J}\}$ are the indices of the detected detected objects and $\mathcal{J}^{\mathrm{C}} = \{j \in \mathcal{J} : i(j) = 0\}$ is the set of clutter measurement indices where $\mathcal{J} = \{1, \ldots, J\}$. Finally, the weights of all posterior hypothesis are normalized.

It should be noted that when implementing the filter on a computer, one should not directly apply the methods presented here, but instead look to a more optimized formulation such as a track oriented one instead of the hypothesis oriented formulation given here. It is also often not desirable to consider all possible data associations for each hypothesis, it is enough to consider only the most likely, and these can be computed very efficiently using a cost matrix. It is also recommended to periodically remove objects with low probability of existence from each hypothesis to improve computational performance.

2.7 Normalized Estimation Error

An important property of Bayesian filters is that the state of the filter is a probability density. Thus it is possible to not only extract parameter value estimates from the state density, but also the uncertainty in those extracted values. When evaluating tracking performance of a filter, it is important not only to evaluate the accuracy of the extracted parameters, but also the accuracy of the uncertainty estimates.

Only method of testing the accuracy of uncertainty estimates of Gaussian filters is by computing a quantity known as the Normalized Estimation Error Squared (NEES). Given a sequence of means and covariance matrices $(\langle \hat{\boldsymbol{x}}_k, \mathbf{P}_k \rangle)_{k=1}^{K}$ of the estimated state density and a corresponding ground truth state sequence $(\boldsymbol{x}_k)_{k=1}^{K}$ we may compute the NEES sequence

$$\varepsilon_k = (\hat{\boldsymbol{x}}_k - \boldsymbol{x}_k)^\top \mathbf{P}_k^{-1} (\hat{\boldsymbol{x}}_k - \boldsymbol{x}_k). \tag{2.39}$$

If the the state estimate $\hat{\boldsymbol{x}}_k$ is consistent with $\mathbf{P}_k$, ε_k should be samples from a χ^2 distribution with $n = \dim(\boldsymbol{x}_k)$ degrees of freedom [24]. If this is indeed true, then the ratio of time steps such that ε_k lies within the confidence interval $[r_1, r_2]$ to all time steps is approximately $1 - \alpha$ where

$$r_1 = F^{-1}\left(\frac{\alpha}{2}, n\right), \quad r_2 = F^{-1}\left(1 - \frac{\alpha}{2}, 2\right) \tag{2.40}$$

where F is the cumulative density function of the χ^2 distribution [24].

3

Methods

The method of this work is split into three steps. First, the data sources to be used for geometry estimation is surveyed. Second, a model of the road is chosen to suit the aim of the project and the available input data. Finally, a filter is designed to estimate the parameters of the road model.

3.1 Input data

This work uses two main sources of input data provided by Zenseact, internal vehicle signals and lane marking detections. Understanding of the available input data is needed for the design of a suitable road model as well as the filtering algorithm.

3.1.1 Internal Vehicle State

A Zenseact algorithm provides a set of preprocessed internal vehicle signals, the most interesting of those being vehicle speed and angular rates. As these signals are already filtered, they are almost free from noise and can be considered very accurate. Therefore this data will be used as the control signal input for the update step of the proposed filter, thus eliminating the need for states to track the rate of change of the vehicle pose in relation to the road.

3.1.2 Lane Marking Detections

The other main data source are lane marking detection, a set of world coordinates corresponding to the position of lane markers produced by a neural network based on camera images from a front mounted camera. An illustration of a typical front camera image and the output of the lane marking algorithm are shown in Figure 3.1. In more detail, the output of the lane marking detection algorithm is a set of 3D coordinates along with uncertainty estimates of these coordinates. As uncertainty estimates are provided, this motives the choice of using lane marking detections as measurements in the update step of the proposed filter.

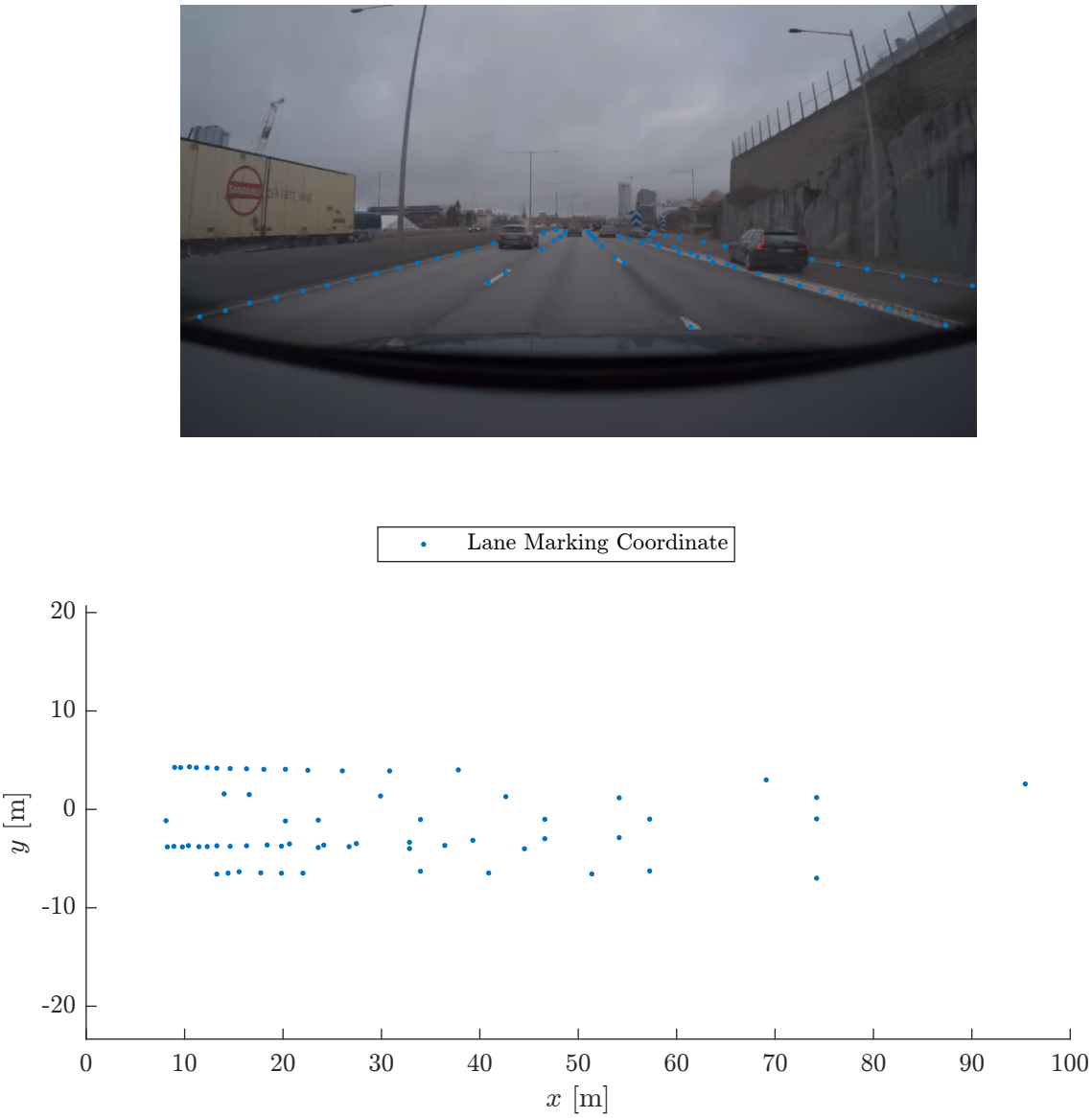

Figure 3.1: The top image show an example front camera image taken from the Zenseact Open Dataset (ZOD) [25] with the addition of reprojected manually placed lane marker detections. The detections illustrate typical output of a 3D lane marking detection algorithm. The bottom figure shows a birds eye view of the same 3D lane marking points which were reprojected into the top image.

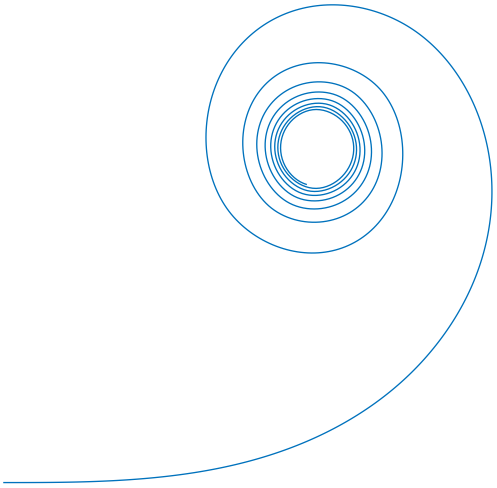

Figure 3.2: An example clothoid curve, a curve with a constant rate of curvature change.

3.2 Road Model

As the goal of this work is to estimate lane centers, the description of lane center curves is central in the choice of model. Additionally, since the lane markings are measured, it should also be possible to extract the possible locations of lane markings from the model.

3.2.1 Clothoid Splines and Their Extension to 3D

Many previous works have focused on describing only the ego lane center, or describing the lane center or marking lines of the road as independent objects. There are multiple common ways to parameterize the curve of a lane center of marker line, such as polynomials [12], lateral offsets at fixed longitudinal points [26], [27] or clothoid splines [4], [5]. In this work, clothoid splines are used as a building block in the road model. A clothoid, illustrated in Figure 3.2 is a 2D curve with constant rate of curvature change along the curve. A clothoid spline is thus a chain of multiple connected clothoid curves constrained such that the curvature is continuous. To find the geometry of the clothoid spline, the following differential equation is solved:

$$\frac{\mathrm{d}}{\mathrm{d}l}\begin{bmatrix} x(l) \\ y(l) \\ \phi(l) \end{bmatrix} = \begin{bmatrix} \cos\phi(l) \\ \sin\phi(l) \\ \kappa(l) \end{bmatrix}, \tag{3.1}$$

where $x(l)$, $y(l)$ are the Cartesian coordinates of the point a distance l along the curve, $\phi(l)$ is the heading angle of the curve and $\kappa(l)$ is a continuous piecewise linear function describing the curvature of each segment – within each segment the slope of κ, i.e the curvature rate, is constant.

In this work, after some initial testing with a 2D road model it was decided to attempt a 3D model since lane marking detection coordinates are given in 3D. To

this end, the 2D clothoid described by (3.1) was extended to 3D according to

$$\frac{\mathrm{d}}{\mathrm{d}l}\begin{bmatrix} x(l) \\ y(l) \\ z(l) \\ \phi(l) \\ \theta(l) \end{bmatrix} = \begin{bmatrix} \cos\phi(l)\cos\theta(l) \\ \sin\phi(l)\cos\theta(l) \\ \sin\theta(l) \\ \kappa(l) \\ \gamma(l) \end{bmatrix} \tag{3.2}$$

where we have introduced the elevation angle $\theta(l)$ with its corresponding piecewise linear curvature $\gamma(l)$.

3.2.2 Parameters of the Road Model

A visualization of the proposed road model and its parameters is shown in Figure 3.3. Starting from the description of the ego lane center as a 3D clothoid parameterized by curvatures κ_0^n at the edge of each segment (these curvatures are then varied linearly in between the segment edges), we then move on to describe the width of the lane in order to describe the possible locations of lane markers. The ego lane markers would be located next to the ego lane center, at a parallel offset to the lane center by half the lane width. We here notice the opportunity to also model other lanes next to the ego lane by the same method: By using their width to describe them as parallel offsets from the ego lane. Each so called main road lane m is given its own width profile as a piecewise linear function parameterized by segment edge widths w_{0m}^n. The length of both the clothoid and width segments are L_0^n, and there are N_0 main road segments in total. Finally, the number of main road lanes is M and the index of the ego lane within these m lanes is m_0.

This work also aims to estimate the center of lanes merging into or diverging from the main road. To this end, we introduce branches to our road model. A branch b can attach to any one of the main road lanes, m_b, at a parameter controlled distance α_b along and angles ϕ_b (azimuthal) and θ_b (elevation) to the lane. From the root of the branch, its curve can similarly to the ego lane be described by a 3D clothoid spline with curvatures κ_b^n. The width of the branch is also described by a piecewise linear function, parameterized by w_b^n, but constrained to have the same width as the main road lane it connect to at its root. Thus we omit w_b^0. The length of both the branch clothoid and width segments are L_b^n, and there are N_b main road segments in total. The number of branches is B.

Finally, the pose of the ego vehicle in relation to the road needs to be parameterized. This is done by describing the vehicle pose in relation to the initial point of the ego lane. The vehicle is given three spatial coordinates x, y and z as well as three Tait-Bryan angles ϕ, θ and ψ representing yaw, pitch and roll respectively. Note that while the ego vehicle thus is allowed roll in relation to the road, the road itself is entirely flat and does not change its banking angle. Neglecting road banking is done in order to limit the number of road model parameters somewhat.

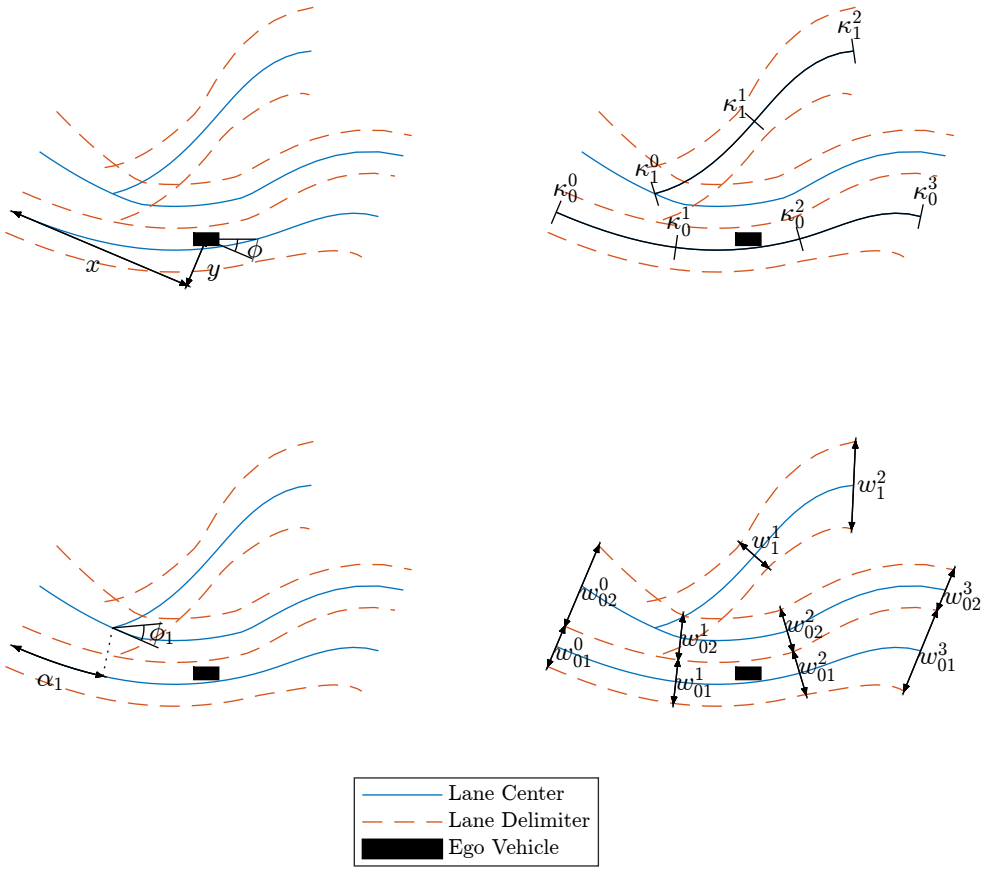

Figure 3.3: Visualization of the road model. Each subfigure shows a different subset of the parameters used. This example road has two main road lanes and one branch. The curvature and width of the ego lane and branches are described by continuous splines being linear interpolations between set values at the edge of each spline segment. Here, the main road splines have four segments and the branch two as illustrated in the top right subfigure. In this birds eye view, the parameters controlling the vertical position of the road as well as roll and pitch of the ego vehicle are omitted for visual clarity.

3.2.3 Multi-Lane Dynamic Model

To formulate the dynamic model we first define the state space representation of the road model. All continuous parameters of the road model, those being x, y, z, ϕ, θ, ψ, ϕ_b, θ_b, κ_i^n, γ_i^n, w_{0m}^n, w_b^n and α_b, are concatenated into the state vector $\boldsymbol{x}$. Furthermore, all discrete parameters of the road model, L_i^n, M, B, N_i and m_i are concatenated into a metadata vector $\boldsymbol{\xi}$. Finally, the road model is also extended with auxiliary variables $\rho_c \in [0, 1]$ which model the existence of each lane $c = 1, \ldots, M + B$, main road lanes and branches, of the road geometry. This in the sense that given the rest of the road model, lane c only exists with probability ρ_c independent of the existence of the other lanes. This is inspired by the existence probabilities introduced for the MBM filter, but note that in contrast to the MBM filter the object state of the proposed filter is not described by Bernoulli random finite sets. These existence variables are concatenated into the vector $\boldsymbol{\rho}$. The complete augmented road model state is thus $[\boldsymbol{x}^\top \quad \boldsymbol{\xi}^\top \quad \boldsymbol{\rho}^\top]^\top$. The control signal $\boldsymbol{u}_k$ at time step k consists of vehicle speed v_k, as well as roll, pitch and yaw rates $\omega_{x,k}$, $\omega_{y,k}$ and $\omega_{z,k}$.

3.2.3.1 Geometry Transition Model

The purpose of the geometry state transition model is to describe the expected change in parameter values from one time step to the next based on the control signal. Neglecting process noise, the motion model only consists of updating the ego vehicle pose,

$$x_k = x_{k-1} + \Delta t\, v_k \cos\phi_{k-1} \cos\theta_{k-1} \tag{3.3a}$$
$$y_k = y_{k-1} + \Delta t\, v_k \sin\phi_{k-1} \cos\theta_{k-1} \tag{3.3b}$$
$$z_k = z_{k-1} + \Delta t\, v_k \sin\theta_{k-1} \tag{3.3c}$$
$$\phi_{0,k} = \phi_{0,k-1} + \Delta t\, \omega_{z,k} \tag{3.3d}$$
$$\theta_{0,k} = \theta_{0,k-1} + \Delta t\, \omega_{y,k} \tag{3.3e}$$
$$\psi_{0,k} = \psi_{0,k-1} + \Delta t\, \omega_{x,k} \tag{3.3f}$$

where Δt is the time step, and with all the other components of $\boldsymbol{x}$ remaining constant except for process noise.

To compensate for motion of the ego vehicle along the main road, whenever the vehicle has progressed far enough along the main road a segment is removed from the back of the road and simultaneously one is added to the front. When this happens, all the state variables connected to segments of the main road are shifted backwards in the state vector and α_b are reduced by the segment length to reflect the new segment layout. In order to aid with merging, branches do not change their length or number of segments, and instead always retain the length they were initialized with.

3.2.3.2 Lane Spawning Model

To account for that the number of lanes in the road may increase from one time step to the next, the state transition adds some new lanes to the state every time

step. As we do not expect the number of lanes to grow very rapidly, these new lanes are given very low probabilities of existence ρ_c. In this work, we choose to spawn one additional main road lane at the very left and very right of the main road. The state parameters controlling the width of the main road lanes as well as the indices m_i are shifted over accordingly. Furthermore, we spawn one diverging and one merging branch starting from the rightmost main road lane with sufficient width and existence probability. For computational performance reasons we neglect spawning of branches to the left of the main road. Such on ramps or off ramps are not common in countries with right hand traffic.

3.2.3.3 Lane Death Model

Inspired by the MBM filter we introduce a probability of survival P^{S} representing the probability that each individual lane survives from one time step to the next. The probability of survival is then used to update the probabilities of existence of each lane according to

$$\rho_{c,k} = P^{\mathrm{S}}\rho_{c,k-1} \quad c = 1, \ldots, M_{k-1} + B_{k-1}. \tag{3.4}$$

Additionally, lanes whose widths are entirely less than some threshold, as well as branches where α_b has become negative are removed from the state.

3.2.3.4 Dynamic Model Summarized

Finally, we summarize the geometry transition model of Section 3.2.3.1, the spawning model of Section 3.2.3.2 and the death model of Section 3.2.3.3 into three transition equations

$$\boldsymbol{x}_k = f_{\boldsymbol{x}}(\boldsymbol{x}_{k-1}, \boldsymbol{\xi}_{k-1}, \boldsymbol{\rho}_{k-1}, \boldsymbol{u}_k) \tag{3.5a}$$
$$\boldsymbol{\xi}_k = f_{\boldsymbol{\xi}}(\boldsymbol{x}_{k-1}, \boldsymbol{\xi}_{k-1}, \boldsymbol{\rho}_{k-1}, \boldsymbol{u}_k) \tag{3.5b}$$
$$\boldsymbol{\rho}_k = f_{\boldsymbol{\rho}}(\boldsymbol{x}_{k-1}, \boldsymbol{\xi}_{k-1}, \boldsymbol{\rho}_{k-1}, \boldsymbol{u}_k) \tag{3.5c}$$

where process noise have been excluded. In particular, $f_{\boldsymbol{x}}$ is mostly given by (3.3), disregarding spawning and cases where the road segments are shifted to avoid negative L_0^n and when the number of segments in a branch is increased. The complete dynamic model including noise is

$$\mathbf{x}_k = f_{\boldsymbol{x}}(\mathbf{x}_{k-1}, \boldsymbol{\xi}_{k-1}, \boldsymbol{\rho}_{k-1}, \boldsymbol{u}_k) + \mathbf{q}_k \tag{3.6a}$$
$$\boldsymbol{\xi}_k = f_{\boldsymbol{\xi}}(\mathbf{x}_{k-1}, \boldsymbol{\xi}_{k-1}, \boldsymbol{\rho}_{k-1}, \boldsymbol{u}_k) \tag{3.6b}$$
$$\boldsymbol{\rho}_k = f_{\boldsymbol{\rho}}(\mathbf{x}_{k-1}, \boldsymbol{\xi}_{k-1}, \boldsymbol{\rho}_{k-1}, \boldsymbol{u}_k) \tag{3.6c}$$

where $\mathbf{q}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k)$ is the process noise.

3.2.4 Multi-Lane Measurement model

The measurement model describes the measurements $\boldsymbol{y}$, here the coordinates of points on the lane markings, one would expect to receive given a value of the state vector. In the proposed model, except for clutter, lane marking measurement may

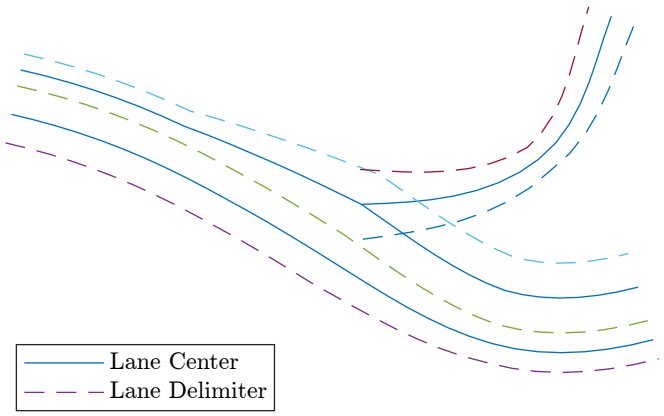

Figure 3.4: Illustration of lane delimiters. Each differently colored dashed line is considered a separate delimiter.

only occur on either edge of a lane, separated from the lane center by half the width of the lane, i.e. on the dashed lines indicated in Figure 3.4. These lines will be known as delimiters. For the measurement model, we will describe the expected position of one lane marking coordinate given that it is known which delimiter it is associated to. The process of associating real measurement to delimiters of the model will be handled by the filter discussed further on in this work. For notational brevity, we drop the time step index k in this section.

First, a discussion on which delimiters are detected. We begin by noticing that only existing delimiters can be detected. From $\boldsymbol{\rho}$ we can only directly get the probability that each lane center exists, but here we assume that given that a lane exists, both its right and left delimiter also exists. Of the existing delimiters, each has probability P^{D} of being detected. A detected delimiter produces at least one measurement, while an undetected delimiter gives rise to no measurements. We next need to describe the measurements produced by each detected delimiter.

We start by describing the geometry of the delimiters in the local coordinate frame of the ego road. As the delimiters of the main road are described in relation to the ego lane center, we first need the geometry of this center line. This is done by integrating (3.2) for the lane center parameters using the Euler method with step size Δl, with the initial conditions for coordinates and angles set to zero as we are working in the local coordinate frame of the ego road. We get

$$\begin{bmatrix} x_{0,i} \\ y_{0,i} \\ z_{0.i} \\ \phi_{0,i} \\ \theta_{0,i} \\ l_{0,i} \end{bmatrix} = \begin{bmatrix} x_{0,i-1} \\ y_{0,i-1} \\ z_{0.i-1} \\ \phi_{0,i-1} \\ \theta_{0,i-1} \\ l_{0,i-1} \end{bmatrix} + \Delta l \begin{bmatrix} \cos\phi_{0,i-1}\cos\theta_{0,i-1} \\ \sin\phi_{0,i-1}\cos\theta_{0,i-1} \\ \sin\theta_{0,i-1} \\ \kappa_0(l_{0,i-1}) \\ \gamma_0(l_{0,i-1}) \\ 1 \end{bmatrix}, \quad i = 1, 2, \ldots \tag{3.7}$$

where $\kappa_0(l)$ and $\gamma_0(l)$ represent linear interpolation over the κ_0^n and γ_0^n parameters respectively.

In order to get the sequences of widths $(w_{0m,i})_i$ corresponding to the previously com-

puted $(\langle x_{0,i}, y_{0,i}, z_{0.i}, \phi_{0,i}, \theta_{0,i}, l_{0,i}\rangle)_i$ of the different main road lanes, the parameters w_{0m}^n are simply interpolated at distances $(l_{0,i})_i$ similar to how $\kappa_0(l)$ is an interpolation of κ_0^n. We are now able to give the coordinates of points on the main road delimiters as

$$\boldsymbol{r}_{d,i} = \begin{bmatrix} x_{0,i} \\ y_{0,i} \\ z_{0,i} \end{bmatrix} + \mathbf{R}_z(\phi_{0,i}) \begin{bmatrix} \mathbf{0} \\ \mathbf{A}_d \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} w_{01,i} \\ \vdots \\ w_{0M,i} \end{bmatrix} \tag{3.8}$$

where $\mathbf{R}_z(\phi) \in \mathbb{R}^{3\times 3}$ is the rotation matrix representing rotation an angle of ϕ around the z axis, and $\mathbf{A}_d \in \mathbb{R}^{1\times M}$ is a matrix used for calculating the offset of delimiter d from the ego lane. Since we have M main road lanes, we have $M+1$ main road delimiters. For $M = 2$ and the ego lane index $m_0 = 2$ we would for example have

$$\mathbf{A}_1 = \begin{bmatrix} -1 & -0.5 \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} 0 & -0.5 \end{bmatrix}, \quad \mathbf{A}_3 = \begin{bmatrix} 0 & 0.5 \end{bmatrix}. \tag{3.9}$$

The coordinates of point on the branch lanes are computed in a very similar fashion by first solving the differential equations of the center line clothoid and lane width using the Euler method. As initial condition for the clothoid, the position is calculated by linear interpolation of the center polyline of the parent main road lane (which is computed similarly to (3.8) but with the $\mathbf{A}$ matrix modified accordingly) a distance α_b along the ego lane center (see the bottom left subplot of Figure 3.3). The initial angle is similarly computed by interpolating the main road lane angles, and then adding the initial angles ϕ_b and θ_b of the branch. The initial curvatures are directly given as κ_b^0 and γ_b^0. For the initial width of the branch, this is also interpolated from the width of the parent main road lane. In total, counting both the main road and branch delimiters there will be $D = M + 1 + 2B$ delimiters with coordinate sequences $(\boldsymbol{r}_{d,i})_i$.

So far, the description of the road geometry has all been in relation to the road frame. Our measurements, however, are given in the ego vehicle frame. This frame is aligned with the vehicle with its origin on the center of the rear axle, the x axis pointing forward, the y axis left along the axle, and z upwards. However, the parameters of the ego vehicle pose, x, y, z and ϕ, θ, ψ it is relatively easy to transform the road-aligned lane delimiter geometry $(\boldsymbol{r}_{d,i})_i$ into the ego vehicle frame, giving us the sequences $(\boldsymbol{r}'_{d,i})_i$.

Finally, some additional notes about the measurement model. Lane marking detections are assumed to be produced by the points along the polylines defined by $\left(\boldsymbol{r}'_{d,i}\right)_i$. Throughout this work, we will assume that given a lane marking measurement coordinate $\boldsymbol{y}_j$ originating from delimiter d, it was produced by the point on the delimiter polyline closest to $\boldsymbol{y}_j$. Also, as previously mentioned, the raw measurements come in the form of 3D coordinates. In order to reduce the dimension of the measurement vector, we seek to find a more compact representation of the innovation than the difference between 3D points. The chosen representation of the innovation is the vector of distances between the measured lane marking coordinates and their corresponding delimiter polylines. The measurement function $h(\boldsymbol{x}, \boldsymbol{\xi})$ is thus always the zero vector. The measurement model does, however, also include

additive measurement noise $\boldsymbol{r} \sim \mathcal{N}(\mathbf{0}, \mathbf{T}\mathbf{R}\mathbf{T}^\top)$, where $\mathbf{R}$ is the covariance matrix of the 3D coordinates $\boldsymbol{y}_j$ and $\mathbf{T}$ is an association dependent matrix which is used to transform $\mathbf{R}$ into the covariance of the distance between $\boldsymbol{y}_j$ and the delimiters, further defined in Section 3.3.2.

3.3 Filter Design

With the road model defined we turn to the design of a filter to estimate the parameters of the model. The proposed filter is a multiple hypothesis tracker. In each hypothesis h, the $\mathbf{x}$ portion of the state is represented by a single Gaussian density with mean $\hat{\boldsymbol{x}}^h$ and covariance matrix $\mathbf{P}^h$, drawing inspiration from Gaussian sum filtering. Thus, all the lanes of the road are coupled. The rest of the state, $\boldsymbol{\xi}$ and $\boldsymbol{\rho}$ are assumed to be known and equal to $\hat{\boldsymbol{\xi}}$ and $\hat{\boldsymbol{\rho}}$ respectively within each hypothesis. The uncertainty in these variables is instead represented by the ensemble of hypotheses.

The prior state density of the filter is described by

$$\left(\left\langle w^h_{k-1|k-1}, \hat{\boldsymbol{\rho}}^h_{k-1|k-1}, \hat{\boldsymbol{\xi}}^h_{k-1|k-1}, \hat{\boldsymbol{x}}^h_{k-1|k-1}, \mathbf{P}^h_{k-1|k-1} \right\rangle\right)_{h=1}^{H_{k-1|k-1}} \tag{3.10}$$

where $w^h_{k-1|k-1}$ are hypothesis weights.

For simplicity, we assume that the probability of survival P^{S} is constant and lane independent, and a PPP clutter model is used with constant intensity λ^{C}. The filter algorithm itself does however support state dependent probabilities of survival and other clutter models than the PPP. For the probability of detection, we let it be state dependent such that it is greater for lanes in front of, and near, the ego vehicle and trails off outside this region. Again for simplicity, we do not bother evaluating the probability of detection for the entire state density, but only evaluate it for the estimation mean $\hat{\boldsymbol{x}}$.

3.3.1 Prediction Step

The prediction step is relatively simple as it simply consists of for all hypotheses, making an EKF prediction of $(\hat{\boldsymbol{x}}, \mathbf{P})$ according to the motion model (3.6a), while $\boldsymbol{\xi}$ and $\boldsymbol{\rho}$ are updated according to (3.5b) and (3.5c) respectively. The weight of each hypothesis is unchanged.

3.3.2 Update Step

The input to the update step is a set of prior hypotheses, coordinates of lane marking detections and their corresponding covariances. Here, the update step for a single hypothesis is described. The same procedure gets applied in all hypotheses, and the resulting posterior hypotheses are concatenated at the end of the update step. For the update, we are supplied J lane marking coordinate measurements $\boldsymbol{y}_j \in \mathbb{R}^3$ and their joint covariance matrix $\mathbf{R} \in \mathbb{R}^{3J \times 3J}$.

The first half of the update step is to produce a number of data association hypotheses. To simplify this work, the measurement points $\boldsymbol{y}_j$ are first gated. This is done first by considering each point individually and comparing it to all available delimiters d. The point is only allowed to be associated to the delimiters where the Mahalanobis distance is below a threshold, and only to a capped number of delimiters determined by maximizing the likelihood of association as approximated by $\mathcal{N}(\boldsymbol{\Delta}_{j,d}; \mathbf{0}; \mathbf{S}_j^d)$, where $\boldsymbol{\Delta}_{j,d}$ is the closest distance between $\boldsymbol{y}_j$ and delimiter d and $\mathbf{S}_j^d$ the corresponding innovation covariance. Next, the points are clustered using a confidential Zenseact provided algorithm, and a similar gating process takes place for each cluster. Clusters are however also allowed to be associated to both a branch delimiter, and the main road delimiter of the parent lane of the branch at the same time. In this work, we do not make the point object assumption. Thus it is possible to associate multiple lane marking detections to each delimiter. After performing these step, each cluster has a set of possible associations of points within the cluster, either to delimiters or to clutter. Next, all possible combinations of cluster associations are produced to create a first set of data association candidates. Finally, these candidates are then restricted further by not allowing two likely incompatible lane marking points to be associated to the same delimiter. Compatibility between lane marking clusters is checked by fitting a third degree polynomial to the markers of the two clusters and checking the fraction of inliers from each cluster.

Now, focus on a single data association hypothesis. Assign a unique index to all the delimiters of the road model and let $d(j)$ be the index of the delimiter associated to measurement $\boldsymbol{y}_j$ in this hypothesis, $d(j) = 0$ if $\boldsymbol{y}_j$ is associated to clutter. First, the goal is to perform an EKF update step. In order to reach this point, we construct the innovation vector $\boldsymbol{\Delta}$ by concatenation of $\boldsymbol{\Delta}_{j,d(j)}$ for $d(j) \neq 0$. The matrix $\mathbf{H}$ (see (2.14)) is constructed by finding the gradient of the point on delimiter $d(j)$ closest to $\boldsymbol{y}_j$ in the direction $\boldsymbol{a}_{j,d(j)}$ of $\boldsymbol{y}_j$. Finally, the measurement noise is $\mathbf{T}\mathbf{R}\mathbf{T}^\top$ where $\mathbf{T}$ is the subset of rows j such that $d(j) \neq 0$ of the block diagonal matrix of $\boldsymbol{a}_{j,d(j)}^\top$. Using these vectors and matrices, we can compute the innovation covariance $\mathbf{S}$ and the posterior mean and covariance $\hat{\boldsymbol{x}}_{k|k}^{h'}$, $\mathbf{P}_{k|k}^{h'}$. Here h' is the index of the posterior hypothesis which originated from h and the data association considered in this section. Note that because of the coupledness of the state, it is not possible to reuse $\mathbf{S}_j^d$ from the gating stage when computing $\mathbf{S}$, however it is possible to reuse the $\mathbf{H}_j^d$ matrices also computed in the gating stage in order to find $\mathbf{S}_j^d$.

It now only remains to compute the updated hypothesis weight $w_{k|k}^{h'}$ and existence probabilities $\rho_{c,k|k}^{h'}$. Given the data associations, it is known which subset of all delimiters $\mathcal{D}$ where detected, namely

$$\mathcal{D}^{\mathrm{D}} = \{d(j) \,:\, j \in \mathcal{J},\ d(j) \neq 0\} \tag{3.11}$$

where $\mathcal{J} = \{1, ..., J\}$. If we let $\mathcal{C}$ be the set of all lanes c, then the possible combinations of lanes which could have existed for us to get measurements from the delimiters in $\mathcal{D}^{\mathrm{D}}$ are

$$\mathcal{C}^{\mathrm{D}} = \left\{\mathcal{C}^{\mathrm{E}} \in \mathcal{P}(\mathcal{C}) \,:\, \mathcal{D}^{\mathrm{D}} \subseteq d(\mathcal{C}^{\mathrm{E}})\right\}, \tag{3.12}$$

where $\mathcal{P}(\mathcal{C})$ is the power set of set $\mathcal{C}$ and $d(\mathcal{C}^{\mathrm{E}})$ is the set of delimiters belonging to any of the lanes in $\mathcal{C}^{\mathrm{E}}$. The probability that exactly the lanes in subset $\mathcal{C}^{\mathrm{E}} \in \mathcal{C}^{\mathrm{D}}$ exist and that out of the delimiters of those lanes, exactly $\mathcal{D}^{\mathrm{D}}$ were detected is

$$\tilde{u}^{\mathcal{C}^{\mathrm{E}}} = \left(P^{\mathrm{D}}\right)^{\left|\mathcal{D}^{\mathrm{D}}\right|}\left(1-P^{\mathrm{D}}\right)^{\left|d(\mathcal{C}^{\mathrm{E}})\setminus\mathcal{D}^{\mathrm{D}}\right|}\prod_{c\in\mathcal{C}^{\mathrm{E}}}\rho_{c,k|k-1}\prod_{c\in\mathcal{C}\setminus\mathcal{C}^{\mathrm{E}}}(1-\rho_{c,k|k-1}). \tag{3.13}$$

Note that we have assumed delimiter independent probability of detection in this equation, while the evaluated filter does have different probability of detection for each delimiter. Given that we know that $\mathcal{C}^{\mathrm{E}}$ must be an element of $\mathcal{C}^{\mathrm{D}}$ we can calculate the corresponding conditional probabilities by normalizing

$$u^{\mathcal{C}^{\mathrm{E}}} = \frac{\tilde{u}^{\mathcal{C}^{\mathrm{E}}}}{\sum_{\mathcal{C}^{\mathrm{E}\prime}\in\mathcal{C}^{\mathrm{D}}}\tilde{u}^{\mathcal{C}^{\mathrm{E}\prime}}}. \tag{3.14}$$

To summarize the progress so far, $\mathcal{C}^{\mathrm{D}}$ contains the possible combinations of lanes which are compatible with our data association. Each combination has weight $u^{\mathcal{C}^{\mathrm{E}}}$. However, to obtain recursive filtering recursions we would like the existence of lanes in the posterior to be independent. We can either add new hypotheses for all the $\left|\mathcal{C}^{\mathrm{D}}\right|$ cases, but since the state density in these hypotheses would all have the same mean and covariance this seems inefficient. Therefore we summarize all the cases in a single hypothesis where we approximate the lane existences as independent. The updated existence probabilities are computed as

$$\rho^{h'}_{c,k|k} = \sum_{\mathcal{C}^{\mathrm{E}}\in\mathcal{C}^{\mathrm{D}}} u^{\mathcal{C}^{\mathrm{E}}}\mathbf{1}_{\mathcal{C}^{\mathrm{E}}}(c), \tag{3.15}$$

where $\mathbf{1}_{\mathcal{C}^{\mathrm{E}}}$ is the indicator function on $\mathcal{C}^{\mathrm{E}}$.

For the completion of the update step, it now only remains to compute the new hypothesis weight $w^{h'}_{k|k}$. To it we have four contributing factors: The prior weight, $\tilde{u}^{C}$, the probability of the data association, the clutter intensity and the measurement likelihood. Putting these together we get

$$w^{h'}_{k|k} \propto w_{k|k-1}\left(\sum_{\mathcal{C}^{\mathrm{E}}\in\mathcal{C}^{\mathrm{D}}}\tilde{u}^{\mathcal{C}^{\mathrm{E}}}\right)\left(\lambda^{\mathrm{C}}\right)^{|\{j\in\mathcal{J}\,:\,d(j)=0\}|}\mathcal{N}(\boldsymbol{\Delta};\mathbf{0},\mathbf{S}). \tag{3.16}$$

Combining all the resulting hypotheses, and normalizing the weights $w^{h'}_{k|k}$ such that they sum to one, we get our posterior density.

To keep the number of hypotheses manageable, capping and pruning and merging are employed after the update step. For the merging step, all pairs of hypothesis with compatible geometry and similar values of the continuous road model parameters are merged. Geometries are considered compatible if they contain the same number of main road lanes and branches, all lanes have the same number of segments, and whether those branches are either merges and diverges (determined by the magnitude of ϕ_b) matches. Merging of hypotheses is done according to (2.31).

3.3.3 Additional Lane Housekeeping

Some additional housekeeping is done to make the filter more stable when evaluating it's performance on real world data: Lanes with low probability of existence are removed. Lanes spawned just prior to the update step which did not get associated to any data are also immediately removed after the update step. Main road lanes that are too wide get automatically split into two lanes. Pairs of neighboring main road lanes whose common width is too small get merged. Branches that are too wide or too thin are pruned. Additionally, if at some point the ego vehicle is deemed to be traveling on a branch instead of the main road, the entire model is reinitialized to replace the branch with a the new main road. Whether the vehicle is traveling on a branch is determined by checking whether the distance between the branch center and ego vehicle is closer than between the current ego lane and ego vehicle, and whether the vehicle is a minimum distance away from the current ego lane.

4

Results

This chapter presents evaluation results for the proposed model and also compares these results to alternate methods when possible. For these evaluation, the main road is configured to always contain seven segments, each 50 m long for a total of 350 m of estimated road. Of this distance, 200 m lies in front of the ego vehicle and the rest behind. The road behind the vehicle is not observed by any sensor, but serves as attachment points for branches. Each branch is given a fixed length of 300 m split over six segments, also making each branch segment 50 m long.

4.1 Evaluation Data Sets and Methodology

The filter is evaluated on closed source data sets provided by Zenseact. For each data set there is high accuracy ground truth geometry available for the ego lane as produced by a smoothing filter applied to on-board sensor data. This ground truth data will be used for evaluating tracking performance on the ego lane. Additionally, for some data sets there is also a full ground truth map available, which is less precise but contains not only the ego lane but all lanes around the ego vehicle. This data set will be used to evaluate the estimated road topology. Since only 2D ground truth data has been provided, all evaluations will consider the estimated 3D road geometry projected onto a horizontal flat plane aligned with the ego vehicle.

4.1.1 Extracting Parameter Estimates

As the filter to be evaluated outputs is a distribution, a method of extracting single parameter values from this distribution is needed. This is done by finding the most likely hypothesis in the posterior, and selecting the lanes of this hypothesis with probability of existence greater than a threshold. This approach to parameter extractions has some problems, namely that the most likely hypothesis may not be representative of the density as a whole. But due to the difficulty of merging hypothesis with varying state lengths (and minor differences in state representation), this method was deemed acceptable.

4.1.2 Ego Lane Evaluation Methodology

In order to put the evaluation results into context, a comparison will be made two another ego lane tracking algorithms developed at Zenseact by [27]. The first method

is a simple baseline algorithm, and the second a deep learning based approach.

4.1.2.1 Baseline RANSAC Lane Estimator

This simple baseline ego lane estimation method developed by [27] uses RANSAC to fit straight lines to the lane marking detections, then applies a basic criterion to identify the ego lane left and right marker lines and finally computes the center between these lines. In case only one of the ego lane markers are detected, the other marker is assumed to be parallel to the first with the ego vehicle driving perfectly in the middle of the road. If no ego lane markers are found, the estimated lane is completely straight and aligned with the ego vehicle.

4.1.2.2 Learning Based Lane Estimator

The learning based approach to estimating the road geometry by [27] takes in ego vehicle signals and lane marking detections, the same as the method proposed in this work, as well as the median heading angle of nearby vehicles, and outputs deviations of the ego lane center from a straight line at fixed anchor points 0–100 m in front of the ego vehicle. This method is thus specialized for estimating the ego lane center, and does not also estimate other lane centers or lane delimiters. The neural network architecture mainly consists of convolutional layers which act upon a 2D occupancy grid representation of the lane marking detections from a birds-eye view, followed by LSTM layers to provide temporal information.

4.1.3 Topology Evaluation Methodology

In order to compare the estimated road topology to a ground truth map, the following method is used: First the evaluation is restricted to only concern the parts of the estimated road geometry which falls inside a rectangular region of interest centered in front of the ego vehicle as illustrated in Figure 4.1. This restriction is introduced in order for the estimation uncertainty within the region of interest to be relatively low so that the matching of estimated lanes to ground truth lanes is less ambiguous. The lateral extent of the region of interest is denoted b and its longitudinal extent a. Then, around every ground truth lane center a gate of half width g is created. For every point on the ground truth lane, if any estimated lane falls within this gate then this point is said to be detected. Otherwise it is missed. Again, refer to Figure 4.1 for an illustration of this methodology. We then compute the fraction of the length of detected ground truth lanes to the total length of the ground truth lanes within the region of interest, as well as the fraction of estimated lane length that does not fall within any ground truth lane gate. In this context we call the former quantity the true positive rate (TP) and the latter the false positive rate (FP). These quantities may then be averaged over a time sequence.

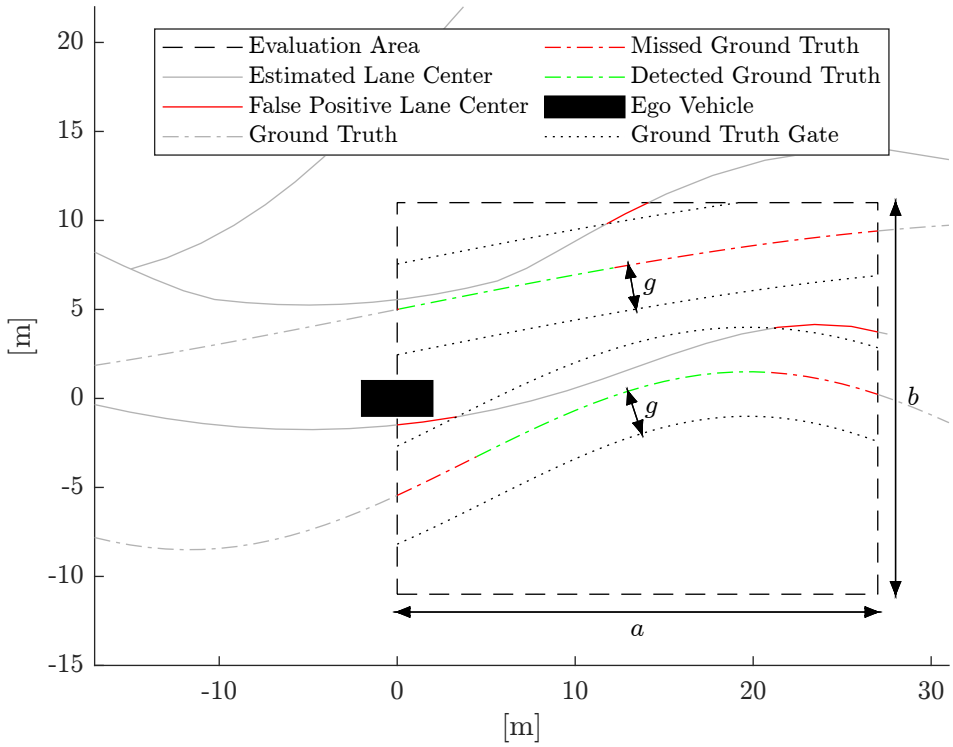

Figure 4.1: Illustration of topology evaluation methodology. Evaluation is performed within a rectangular region of interest area with side lengths a and b aligned with and centered in front of the ego vehicle. Ground truth lanes with an estimated lane center within the gate distance g are considered detected. Estimated lanes not within a distance g of any ground truth lanes are considered false positives.

4.2 Road Topology

To illustrate the tracking performance of the filter, Figure 4.2 shows the filter tracking an off ramp at an intersection. The off ramp is first modeled as a new main road lane, before this new main road lane spawns a branch which then takes over the off ramp. A merge from the same intersection is shown in Figure A.1 of Appendix A. From visual inspection in general, the filter does well with tracking many main road lanes, both in simple scenarios with a one or two lanes, as well as more complex scenarios with many (five or six) lanes. The filter also performs well even the main road is particularly curved, or when the line of sight more than just a couple of meters in front of the ego vehicle is blocked such as in queuing scenarios. However, while the filter does manage to also track merges and diverges using branching lanes, the branch tracking is not very robust. In particular, there are many false positive branch detections. This among other typical failure cases will be discussed further in Section 4.2.1.

Next, we look at the topology evaluation according to the metric defined in section 4.1.3. The evaluation is performed with the lateral extent of the region of interest (ROI) fixed to $b = 80\,\text{m}$, while the longitudinal extent a was either 50, 100 or 150 m. Data used for this evaluation is a set of drives totaling 15 minutes of driving. About 60 percent of this duration is on two lane highway, with the rest on three lane. In approximately 27 percent of the data, there is queuing on the road and poor visibility as a result of densely packed cars. In total, it contains four merges and four diverges, all to the right. The results of this evaluation are shown in Table 4.1. In this table we for example see that the true positive fraction decreases as we expand the region of interest. This is largely due to less accurate lane estimates at longer ranges, increasing the likelihood that the estimated lanes are outside the ground truth lane gates. Therefore, the smaller regions of interest may be more useful when gauging the filters ability to detect multiple lanes. However, even at shorter ranges, the fraction of true positives is very much decided by the intersections present in the evaluation data set. The filter is very good at tracking main road lanes when there are no on or off ramps present, but tends to struggle more with intersections. The false positive fraction is mainly influenced by three factors: The first is that the filter sometimes spawns "ghost" branches when there is no corresponding lane in the real world. This phenomenon will be discussed more further on in Section 4.2.1. The other factor is model mismatches. For example, when two lanes merge into one, the filter might for a while still report two lanes, only where one of the lanes has a very small width. This extra lane will be counted towards false positives. Another example is that a main road lane either exists as a whole or doesn't exist. This becomes a problem when a branch splits off from a main road lane where after in the real world the main road lane doesn't continue, but in the model it still has to. The third factor is similar to model mismatches, and is due to differing representations of the lane center by the model and ground truth. For example, exactly where a two lanes merge may be ambiguous, and depending on how you choose to represent the two lanes merging, they may either be two slowly joining lanes or one wide, slowly narrowing lane. Finally, it is worth noting that all these results also depend on the accuracy of the lane marking detections and their corresponding uncertainties as

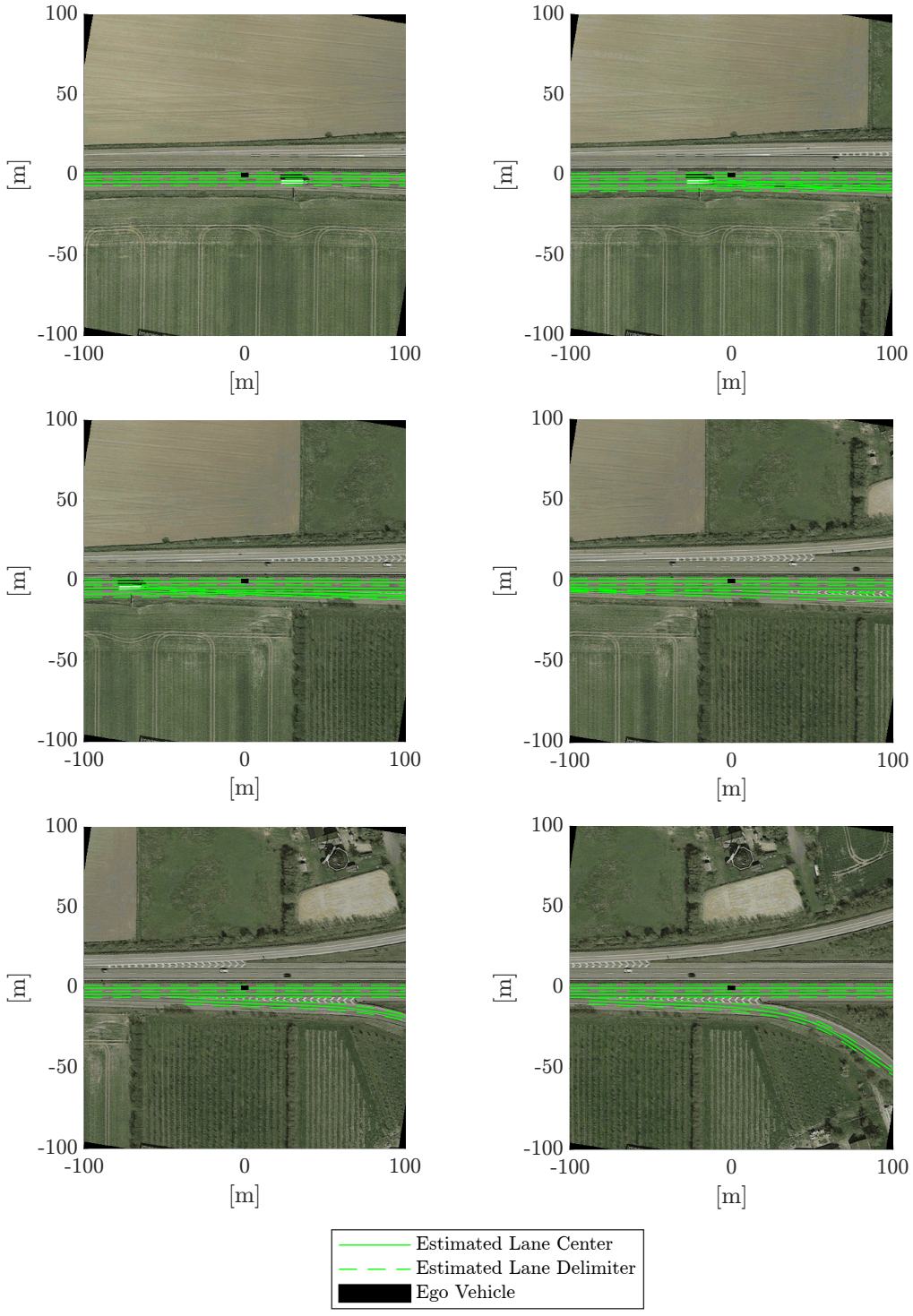

Figure 4.2: Snapshots from the filter tracking an off ramp, overlayed over satellite images of the intersection fetched from Google Maps. First, the off ramp is represented by an extra main road lane which then gets replaced by a branch. Only lanes with sufficiently high probability of existence are shown.

Table 4.1: Mean true and false positive fractions as defined in Section 4.1.3 for different region of interest longitudinal extents a and gate sizes g. The lateral extent of the region of interest is fixed at $b = 80\,\text{m}$.

Mean True Positive Fraction

		ROI extent a		
		50 m	100 m	150 m
Gate g	0.75 m	0.924	0.874	0.808
	1.00 m	0.945	0.918	0.875
	1.25 m	0.957	0.941	0.913

Mean False Positive Fraction

		ROI extent a		
		50 m	100 m	150 m
Gate g	0.75 m	0.090	0.139	0.205
	1.00 m	0.068	0.095	0.138
	1.25 m	0.056	0.071	0.010

reported by the lane marking detection algorithm.

4.2.1 Common Failure Cases

The filter does show some fairly common failure cases, exemplified in Figure 4.3. In the top image, the road is just rewidening to three lanes after a narrow underpass. The road width is however expanding too fast for the road model to accurately describe the lane width with the linear spline width profiles given to each lane. Therefore the filter spawns a branch. It is a common phenomenon that when some measurements don't fit any other delimiter according to the filter, it sometimes mistakenly spawns a branch to explain these measurements instead of attributing them to clutter. When tuning the filter there is a trade off involving the clutter intensity λ^{C} and the probability of existence of birth components which could potentially make these "ghost" branches less likely to appear, but this may cause knock on effects elsewhere, such as not spawning branches when really needed. Branches are generally more flexible in their geometry than main road lanes since they are not constrained by parallelism. This additionally gives rise to conflicts between the branches and the main road, where a human might say that one lane should be a main road lane, but a branch being more flexible is able to explain the measurement data better.

The second failure case of Figure 4.3 shows what in the real world is a merge, but which the filter tries to explain with a very twisted diverging branch. This problem is most likely caused by the choice of distribution of the parameters of the spawn branches. In order to reduce the number of spawn components for computational performance gains, the uncertainty in each spawn component is high in order to cover more possible cases. However this has the knock on effect of the spawn branches sometimes being too flexible and having the ability to bend into unnatural trajectories. This failure case also illustrates another issue which could also probably be remedied somewhat with more compute power: The filter is not punished for lanes which cross one another. One reason this is not included in the current algorithm is that computing which lanes cross is very computationally intensive.

The third failure case of Figure 4.3 is typical with merges, but also happens for a few diverges. The branching lane is mistaken for two main road lanes. This happens because the filter has no way of knowing (other than the shape of the marker lines) that the space between the main road and the branch should not contain any lane.

Table 4.2: Comparison of the ego lane center tracking performance between method proposed in this work (Our), a learning based method (NN) and a baseline method (BL). The table shows lateral offset errors to a high-resolution ground truth provided by Zenseact, both root mean squared errors (RMS) and median absolute errors (median). The horizon indicates the distance in front of the ego lane vehicle at which the errors were evaluated.

Horizon [m]	0	10	20	30	40	50	60	70	80	90	100
BL (RMS) [m]	0.281	0.401	0.630	0.888	1.112	1.379	1.624	1.936	2.310	2.731	3.176
NN (RMS) [m]	**0.157**	**0.142**	**0.137**	**0.169**	**0.243**	**0.355**	0.502	0.684	0.903	1.156	1.451
Our (RMS) [m]	0.212	0.223	0.241	0.271	0.321	0.392	**0.487**	**0.607**	**0.756**	**0.936**	**1.150**
BL (median) [m]	0.145	0.094	0.086	0.097	0.124	0.142	0.169	0.204	0.239	0.296	0.358
NN (median) [m]	0.089	0.075	0.074	0.099	0.124	0.150	0.171	0.203	0.246	0.304	0.376
Our (median) [m]	**0.049**	**0.054**	**0.063**	**0.075**	**0.092**	**0.116**	**0.146**	**0.182**	**0.227**	**0.278**	**0.334**

This failure case could be mitigated by informing the filter about which spaces are expected to not contain any lanes, such as feeding it information about barrier or road edge detections. The type of lane marking line also carries some information which in the current filter is not utilized: Typically for example, a solid line indicates the edge of the road while a dashed marker indicates that there probably is a lane on either side of it.

4.3 Ego Lane Geometry

The ego lane geometry was evaluated on two different data sets provided by Zenseact. The first consists of 823 sequences of highway driving without lane changes, each 30 s long. For every sequence, the filter was given the first 10 s to converge, after which evaluation started. After also removing some time steps with invalid ground truth, the remaining sequences totaled 3.4 hours in length. This is the test data set also used for evaluating the learning based lane estimation algorithm presented in Section 4.1.2.2. A comparison of the ego lane center tracking performance between method proposed in this work and the learning based method is presented in Table 4.2 for the range 0–100 m. From this data it is evident that the method proposed in this work performs ego lane center tracking at least on par with a neural network specialized for ego lane tracking.

The second data set used for evaluating the ego lane geometry consists of 111 sequences varying in length from 0 to 5 minutes, gathered from highway driving and including lane changes. Evaluating on the time steps with valid ground truth, the sequences sum to 4 hours and 37 minutes of driving data. In comparison with the previous data set used in Table 4.2, the lane marking detections are also less disturbed by noise in these scenarios. The high-resolution ground truth also includes the position of the left and right lane marking lines of the ego lane. Since the proposed method also estimates lane widths, it is possible to extract estimated positions of these lane marking lines from the filter. Table 4.3 thus shows the comparison of both estimated lane center and left and right markers to the ground truth for the full estimation horizon span of 0–200 m. It should be noted that as opposed to Ta-

Figure 4.3: Some typical failure cases. In the top image, the number of lanes is increasing back up from two to three after a narrow underpass. The fast increase in road width causes a "ghost" branch to spawn. In the center image, the model believes that a merge is in fact a heavily bent diverge. In the bottom image a on ramp is mistaken for two main road lanes. Satellite images have been fetched from Google Maps.

Table 4.3: Lateral estimation errors of ego lane features at various longitudinal distances, horizons, from the ego vehicle along the road given by the column headers. The evaluated ego lane features are center (Center), as well as the left (Left) and right (Right) lane marker lines. The lateral offsets are presented both as root mean squared errors (RMS) and median absolute errors (median).

Horizon [m]	0	20	40	60	80	100	120	140	160	180	200
Center (RMS) [m]	0.319	0.376	0.491	0.662	0.833	1.064	1.186	1.350	1.333	1.574	1.820
Left (RMS) [m]	0.376	0.424	0.529	0.686	0.841	1.061	1.175	1.337	1.316	1.554	1.799
Right (RMS) [m]	0.356	0.411	0.519	0.687	0.863	1.095	1.219	1.380	1.363	1.605	1.849
Center (median) [m]	0.049	0.059	0.081	0.120	0.177	0.248	0.331	0.422	0.526	0.641	0.755
Left (median) [m]	0.074	0.076	0.086	0.116	0.163	0.227	0.305	0.393	0.496	0.609	0.721
Right (median) [m]	0.112	0.119	0.138	0.178	0.235	0.303	0.381	0.470	0.566	0.678	0.790

ble 4.2, the RMS and medians shown in this table were only taken over ranges that were deemed valid at each time stamp. Data outside the valid range at each time stamp was discarded. Which ranges are considered valid is provided by Zenseact based on many factors related to visibility of the road. Because of this, the tracking performance does not deteriorate very drastically at long horizons. One notices the accuracy of the estimated left lane marker is often better than for the right. This is probably due to the real road having more splits and merges on the right side, affecting the placement of the lane markers. Generally, it is possible to have about a one meter accuracy of the ego lane road geometry at 200 m in front of the ego vehicle given that the visibility is good.

4.3.1 Estimation uncertainty

For evaluating the estimation uncertainty estimates provided by the filter, we study a representative drive from the set of 111 highway sequences. In this approximately 150 s long drive, the main road mostly consists of two lanes, with a couple of off and on ramps to the right, thus sometimes bringing the number of main road lanes up to three, and the visibility is mostly good. Plots of the NEES and 95 percent confidence interval bounds for this sequence is shown in figure 4.4. In this figure we notice a few things: Firstly, the NEES seems a lot more time correlated at shorter horizons. This may be due to slight biases between the ground truth definition of the lane center and the definition used by the proposed method, which is the center between the lane marking lines, or be the result of not directly observing the road underneath the ego vehicle and thus "locking in" whatever guess was made the last time each piece of road was seen. Secondly, the magnitude of the NEES seems to go through different phases during the drive. This is natural since both the visibility and road layout changes throughout the drive, and the filter may be better adapted for some scenarios than to others. At around time step $k = 2500$ for example, the road goes from being slightly curved to almost completely straight, perhaps as an indication to why the magnitude of the NEES seems to drop around this point. Thirdly, especially at longer horizons the filter seems under confident. This could be remedied by tuning the filter parameters, but it is likely that this underconfidence is beneficial in those scenarios where the distant road is more difficult to predict.

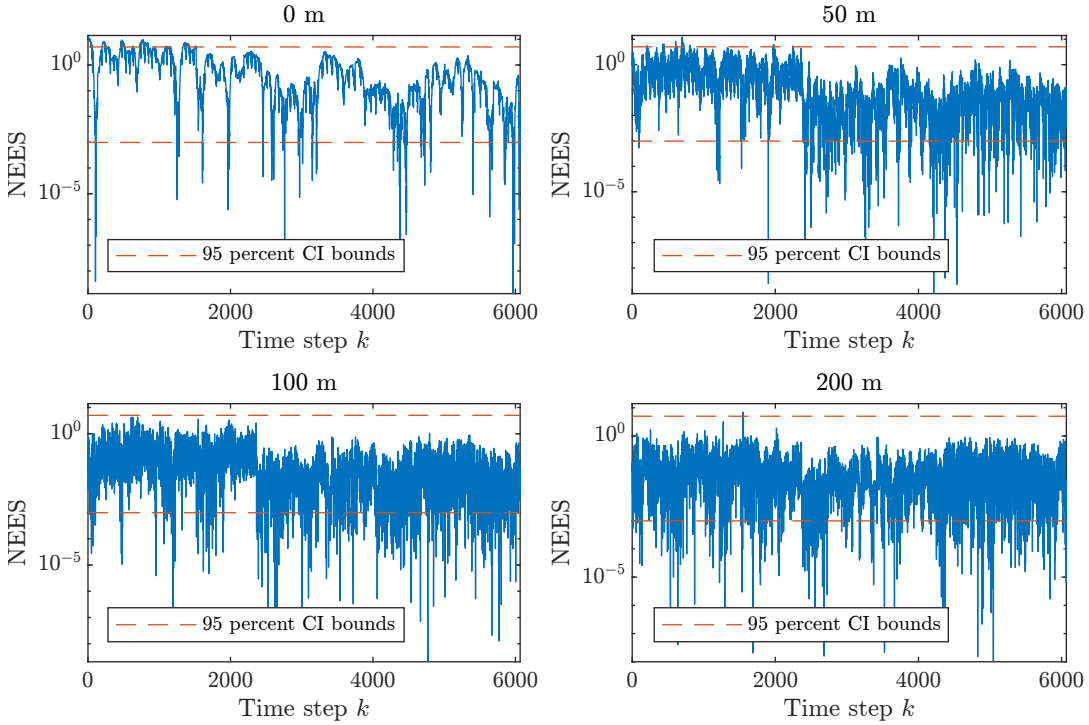

Figure 4.4: Logarithmic plots of normalized estimation error squared (NEES) for lateral error at a few different distances in front of the ego vehicle as indicated above each plot. Additionally, 95 percent confidence interval bounds calculated according to (2.40) are shown. The magnitude of the NEES varies throughout the sequence, and decreases as the distance from the ego vehicle increases, suggesting the filter is underconfident in its long range estimations, but overall the NEES is in line with expectations.

All told, the NEES as computed from this scenario seems to be reasonably in line with expectations.

5

Discussion

The discussion goes over some of the lessons learned from the choice of road model and filter design, as well as from the evaluations presented in this report. Throughout we touch on areas of future work, and the chapter is ended with a conclusion section.

5.1 Evaluations

The evaluations of the proposed method was mainly split into two parts, the first focusing on comparisons to satellite images and the second on ego lane accuracy. It is perhaps slightly surprising that the proposed method seems to outperform the learning based approach discussed in Section 4.1.2.2 at 60–100 m, which are still quite moderate ranges. The two methods have access to the same input data, the learning based approach is specialized for the ego lane center tracking task, and finally, the lane marking measurements do not follow the typical assumptions needed for a Bayesian method to perform well, namely their uncertainty is very difficult to model. Although the measurement are provided along with estimates of their noise covariance, the reality is that these detections likely originate from some neural network with very complex internal behavior. One might expect that a learning based lane tracker may learn the intricacies of this lane marking detecting network and therefore make better use of its output.

As for evaluating the full estimated road geometry, this becomes difficult even when one has access to a ground truth map of the full road network. One then has to compare two graphs, and these graphs may not be defined in the same way. For example, how one defined the lane center may differ, especially at complex intersections. The herein used evaluation metric of false and true positive fractions as defined in Section 4.1.3 is simple but flawed. For example, it allows for one estimated lane center to be within the gate of multiple ground truth lanes. It doesn't take estimation uncertainty into account and also doesn't explicitly compare the topology of the estimated lanes to the ground truth topology, as in e.g. checking that a branch originates from the correct main road lane. For making possible broad comparisons between multi-lane tracking algorithms, it would be useful to have a standard evaluation measure for full road geometries, similar to how lateral offsets to ground truth are currently broadly used for single lane tracking. However, a good evaluation metric for full geometries would likely need to be much more

complex. As for the evaluation results themselves, the current filter does a good job at estimating the correct number of main road lanes, but is not very robust when it comes to detecting merges and diverges. This is likely partially due to not having a very clear distinction between which lanes should be considered main road lanes and which should be considered branches.

Sometimes the tracking algorithms also makes mistakes which could be remedied by feeding it more information. For example, in some scenarios where there are two parallel but disconnected lanes, the tracking algorithm also inserts a middle lane which is not present in the real world. From the perspective of the filter, this makes sense since the lane marking lines are not affected by whether the middle lane exists or not. By feeding the filter information about the type of lane marking, road edges or guard rails, the filter may be able to make more informed decisions as to which spaces should be filled with lanes and which not. This type of information would probably also be helpful to distinguish main road lanes from branches.

5.2 Choice of Road Model

This report mainly boils down to the description and evaluation of two things: A road model and a filter to estimate the parameters of this model. It is sometimes difficult to say what is a feature of the road model and what is a feature of the filter. One problem encountered when working with the road model is that there is no obvious distinction between which lanes are main road lanes and which lanes are branches. The filter therefore understandably sometimes confuses the two, and e.g. spawns another main road lane instead of a branch when it first detects the start of an off ramp. In the real world this distinction is also sometimes very much undecided. But the benefit of the choice of splitting lanes into main road lanes and branches, besides from deciding parallelism, which is an important point, is that it becomes easier to model where it is possible to change lanes. If all lanes were meddled as branch-like, one would have to make a decision for when two branches are close enough together so that a lane change between them is possible. This being said, the road model would be much simplified if there was no distinction between main road lanes and branches. There is also the question of whether one should choose to primarily describe the lane center geometry or lane marking geometry when selecting a road model. From a filtering perspective it would be easier to track lane markings since these can be directly detected, but one then has to do some heuristic post processing steps in order to arrive at the lane center geometry. All told the proposed road model is satisfactory in that it successfully models parallel lanes between which you can make lane changes as well as merges and diverges.

5.3 Computational Performance of the Filter

However, with one big coupled road model it becomes a challenge to estimate all its parameters efficiently. Firstly, the number of parameters in each hypothesis is very large, often over one hundred. It is therefore slightly surprising that the MATLAB

implementation of filter is able to run at only around one order of magnitude slower than real time on a 2.6 GHz Intel Core i9 processor. Also negatively contributing to the computational performance situation is the fact that the proposed filter is inherently hypothesis oriented. This means that filter state has to be implemented as a list of hypothesis, as opposed to track oriented where the filter state is implemented as a list of object tracks, each hypothesis then picking and choosing some subset of those tracks. Since the entire state is coupled, one also has to make sure that innovation covariance matrices are correctly computed taking the entire state into account. Inverting and finding determinants of these big matrices can also cause some numerical stability issues. For this report these were sidestepped by carefully tuning the process noise covariance matrices. Surprisingly, the main computational performance bottleneck of the current filter implementation does not seem to be the matrix inversions, but the computation of the lane delimiter lines discussed in Section 3.2.4 and corresponding gradients. It is likely that large performance optimizations can be done in this area, particularly with memory management since the matrices to store the gradients can grow quite large, and are also somewhat sparse.

5.4 Conclusion

In summary, this work presents a novel multi-lane road model which classifies lanes as either main road lanes or branches. A multiple hypothesis tracking filter has also been developed to track the road geometry. The filter estimates the main road lane geometry well and tracking performance for the ego lane center is very good, even at long ranges. The algorithm also successfully detects and tracks merges and diverges, however in its current state the filter sometimes spawns false branches. It is probable that the filter could be improved in many ways, both in terms of computational complexity and tracking performance, like adding inputs for road edge or barrier detections. Overall, this work demonstrates the potential of a road model based on a main road supplemented with branches, but more work is needed, perhaps first in defining rigorous methods of evaluating the full estimated road geometry of general multi-lane tracking algorithms.

Bibliography

[1] Council of European Union, *Regulation (eu) 2019/2144 of the european parliament and of the council*, 2019. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2019/2144/oj.

[2] Y. Xu, V. John, S. Mita, H. Tehrani, K. Ishimaru, and S. Nishino, "3d point cloud map based vehicle localization using stereo camera," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2017, pp. 487–492. DOI: 10.1109/IVS.2017.7995765.

[3] K. Kim, S. Cho, and W. Chung, "Hd map update for autonomous driving with crowdsourced data," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1895–1901, 2021. DOI: 10.1109/LRA.2021.3060406.

[4] F. Burman and A. Greger, "Automatic landmark recognition in aerial images for the autonomous navigation system of unmanned aerial vehicles," M.S. thesis, Chalmers University of Technology, 2022.

[5] L. Hammarstrand, M. Fatemi, Á. F. García-Fernández, and L. Svensson, "Long-range road geometry estimation using moving vehicles and roadside observations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 8, pp. 2144–2158, 2016. DOI: 10.1109/TITS.2016.2517701.

[6] Y.-C. Zhang, "Road geometry estimation using vehicle trails: A linear mixed model approach," *Journal of Intelligent Transportation Systems*, 2022, ISSN: 1547-2450. DOI: https://doi.org/10.1080/15472450.2021.1974858. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1547245022003462.

[7] G. Cui, J. Wang, and J. Li, "Robust multilane detection and tracking in urban scenarios based on lidar and mono-vision," *Image Processing, IET*, vol. 8, pp. 269–279, May 2014. DOI: 10.1049/iet-ipr.2013.0371.

[8] A. H. Sakr, G. Bansal, V. Vladimerou, K. Kusano, and M. Johnson, "V2v and on-board sensor fusion for road geometry estimation," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017, pp. 1–8. DOI: 10.1109/ITSC.2017.8317876.

[9] A. Abramov, C. Bayer, C. Heller, and C. Loy, "Multi-lane perception using feature fusion based on graphslam," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 3108–3115. DOI: 10.1109/IROS.2016.7759481.

[10] D. E. Clark, K. Panta, and B.-n. Vo, "The gm-phd filter multiple target tracker," in *2006 9th International Conference on Information Fusion*, 2006, pp. 1–8. DOI: 10.1109/ICIF.2006.301809.

[11] Á. F. García-Fernández, J. L. Williams, K. Granström, and L. Svensson, "Poisson multi-bernoulli mixture filter: Direct derivation and implementation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 4, pp. 1883–1901, 2018. DOI: 10.1109/TAES.2018.2805153.

[12] K. Zhao, M. Meuter, C. Nunn, D. Müller, S. Müller-Schneiders, and J. Pauli, "A novel multi-lane detection and tracking system," in *2012 IEEE Intelligent Vehicles Symposium*, 2012, pp. 1084–1089. DOI: 10.1109/IVS.2012.6232168.

[13] Y. Son, E. Lee, and D. Kum, "Robust multi-lane detection and tracking using adaptive threshold and lane classification," *Machine Vision and Applications*, vol. 30, Feb. 2019. DOI: 10.1007/s00138-018-0977-0.

[14] S. Simo, *Bayesian Filtering and Smoothing (Institute of Mathematical Statistics Textbooks, Series Number 3)*, English. Cambridge University Press, Oct. 21, 2013, ISBN: 978-1107619289.

[15] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[16] S. K. Singh, M. Premalatha, and G. Nair, "Ellipsoidal gating for an airborne track while scan radar," in *Proceedings International Radar Conference*, 1995, pp. 334–339. DOI: 10.1109/RADAR.1995.522568.

[17] J. Collins and J. Uhlmann, "Efficient gating in data association with multivariate gaussian distributed states," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 28, no. 3, pp. 909–916, 1992. DOI: 10.1109/7.256316.

[18] R. L. Streit, "The poisson point process," in *Poisson Point Processes: Imaging, Tracking, and Sensing*. Boston, MA: Springer US, 2010, pp. 11–55, ISBN: 978-1-4419-6923-1. DOI: 10.1007/978-1-4419-6923-1_2. [Online]. Available: https://doi.org/10.1007/978-1-4419-6923-1_2.

[19] D. Alspach and H. Sorenson, "Nonlinear bayesian estimation using gaussian sum approximations," *IEEE Transactions on Automatic Control*, vol. 17, no. 4, pp. 439–448, 1972. DOI: 10.1109/TAC.1972.1100034.

[20] L. Svensson, *Single object tracking in clutter*, https://chalmersuniversity.app.box.com/s/kbkmglktznkb2tjlr9pqefz3ezbiyw8p/file/1139125805961, Version January 15, 2021, 2019.

[21] A. D'Ortenzio, C. Manes, and U. Orguner, *Adaptive mixture approximation for target tracking in clutter*, 2022. arXiv: 2211.13624 [stat.AP].

[22] A. R. Runnalls, "Kullback-leibler approach to gaussian mixture reduction," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 3, pp. 989–999, 2007.

[23] K. Granström, *Multi-object conjugate priors*, https://chalmersuniversity.app.box.com/s/kbkmglktznkb2tjlr9pqefz3ezbiyw8p/file/1139152121110, Version May 3, 2019, 2019.

[24] Z. Chen, C. Heckman, S. Julier, and N. Ahmed, "Weak in the nees?: Autotuning kalman filters with bayesian optimization," in *2018 21st International Conference on Information Fusion (FUSION)*, 2018, pp. 1072–1079. DOI: 10.23919/ICIF.2018.8454982.

[25] M. Alibeigi, W. Ljungbergh, A. Tonderski, *et al.*, *Zenseact open dataset: A large-scale and diverse multimodal dataset for autonomous driving*, 2023. arXiv: 2305.02008 [cs.CV].

[26] N. Garnett, R. Cohen, T. Pe'er, R. Lahav, and D. Levi, *3d-lanenet: End-to-end 3d multiple lane detection*, 2019. arXiv: 1811.10203 [cs.CV].

[27] K. Niwong and S. Amirijoo, "Learning based road estimation," M.S. thesis, Lund University, 2023.

A

Appendix 1

This appendix contains some additional plots of filter evaluations.

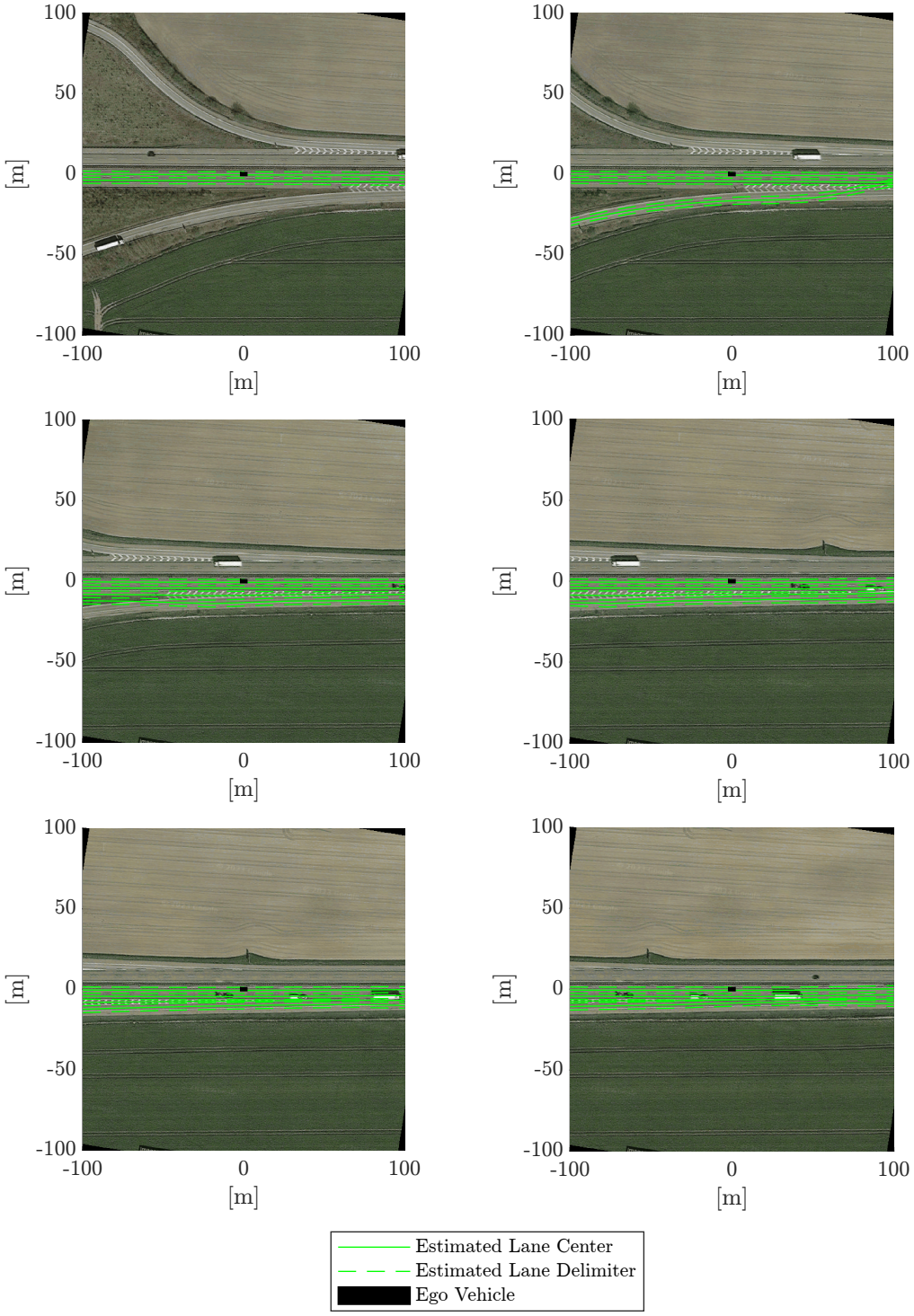

Figure A.1: Merge from the same intersection as Figure 4.2. Satellite images fetched from Google Maps.

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

CHALMERS
UNIVERSITY OF TECHNOLOGY