Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Håvard Skåra Mellbye

Gaussian Processes for long-term trajectory prediction using historical AIS data

A Bayesian Approach

Master's thesis in Cybernetics and Robotics
Supervisor: Edmund Førland Brekke
Co-supervisor: Trym Tengesdal
June 2021

NTNU
Norwegian University of
Science and Technology

Håvard Skåra Mellbye

Gaussian Processes for long-term trajectory prediction using historical AIS data

A Bayesian Approach

Master's thesis in Cybernetics and Robotics
Supervisor: Edmund Førland Brekke
Co-supervisor: Trym Tengesdal
June 2021

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Preface

I want to thank my supervisors Edmund Førland Brekke and Trym Tengesdal for all your guidance and especially for allowing me the freedom to focus on the topics I find the most interesting. As a result, I have had the opportunity to dive deep into the world of probabilistic methods and Bayesian Inference. I also want to thank Ravinder Praveen Kumar Jain for your openness and your insight into long-term trajectory prediction.

I want to thank my family for all your support throughout my studies and growing up. The math homework I had to do as a child did pay off eventually. My grandmother, Anne Marie Mellbye, has also given me tons of support, and I will be forever grateful!

My partner in crime, Kristine Stray, deserves a tremendous amount of credit. You have patiently listened to me blabbering about statistics and probability theory, and even though I know you are not as amazed as me, you still encourage me to continue! So for this thesis, I especially want to thank you for both helping me keep motivated as well as giving me valuable input.

This work marks the end of my graduate studies in Cybernetics and Robotics Engineering at the Norwegian University of Science and Technology. I want to dedicate this thesis to my grandfather, Hans Jørgen Mellbye, who graduated from NTH in 1963 as an electrical engineer. He introduced me to the amazing world of engineering at an early age, which has become a crucial part of who I am today.

Abstract

An essential aspect of safe operations of *Autonomous Surface Vehicles (ASV)* is a robust Collision Avoidance (COLAV) system. In addition to the ability to react to dangerous situations, it is also highly beneficial for the COLAV to be able to proactively avoid high-risk scenarios. In order to do so, the ASV requires solid situational awareness and the ability to understand how the future might unfold given a current scenario.

Predicting the future behavior of surrounding vessels is the topic of this thesis. By utilizing historical data from the Automatic Identification System (AIS), the goal is to predict the trajectories of vessels into the future.

Two methods based on a Gaussian Process (GP) framework are proposed. The GP's intuitive interpretation as a statistical distribution over functions allows the predictions to also incorporate uncertainty as a first-class citizen of the model. A Bayesian statistical framework is applied to always explicitly consider the underlying uncertainty when performing predictions.

The first proposed method directly applies the GP framework to model the trajectories as a function of time. This approach works reasonably well, except for in the presence of branching traffic lanes. This formulation makes strict assumptions about unimodality and is unable to represent any form of multimodal uncertainty.

As a more indirect approach, the second method attempts to use a GP to describe a latent motion model and use it to simulate trajectories numerically. This formulation is far more flexible and is, in theory, able to express multimodal trajectory distributions. Combining this approach with an Extended Kalman Filter (EKF)-based prediction scheme to simulate trajectories works well as long as the trajectories are sufficiently smooth, such that a Taylor approximation of the motion model serves as a reasonable approximation. These assumptions do, however, make this method more fragile than the first method.

Both methods are tested extensively on a real AIS dataset collected from the Trondheim fjord over the course of one year, and the statistical performance of both methods are compared. The consistency of the uncertainty estimates is also tested to investigate whether the methods are able to accurately represent the true underlying uncertainty.

Sammendrag

For at autonome overflate-fartøyer (ASVer) skal kunne operere trygt er det essensielt med robuste antikollisjonssystemer. Slike systemer innebærer ikke bare at et fartøy må kunne reagere i det det oppstår farlige situasjoner, men også evnen til å proaktivt unngå situasjoner med høy risiko. Fartøyene er dermed nødt til å gjenkjenne ulike scenarioer og kunne planlegge for potensielle hendelser frem i tid. Denne fremtidsforståelsen er temaet i denne oppgaven, og målet er å utforske hvordan historisk data fra Automatisk Identifikasjonssystem (AIS) kan brukes til å predikere skips fremtidige bevegelser.

Mer spesifikt foreslår denne oppgaven to metoder som begge benytter Gaussiske Prosesser til å lære bevegelsesmønsteret til skip i ulike scenarier basert på historiske data. Motivasjonen bak bruken av Gaussiske Prosesser er basert på dens intuitive tolkning som en statistisk fordeling over funksjoner. En slik representasjon kan dermed naturlig innlemme usikkerhet knyttet til prediksjonene som en sentral del av modellen. Et Bayesiansk statistisk rammeverk brukes i tråd med Gaussiske Prosesser for å eksplisitt vurdere den underliggende usikkerheten.

Den første foreslåtte metoden bruker et rammeverk basert på Gaussiske Prosesser direkte for å modellere posisjon i banen som en funksjon av tid. Denne tilnærmingen fungerer rimelig bra, bortsett fra i nærvær av forgrenede trafikkfelt. Formuleringen av metoden gjør strenge antakelser om unimodalitet og er ikke i stand til å representere noen form for multimodal usikkerhet.

Som en mer indirekte tilnærming forsøker den andre metoden å bruke en Gaussisk Prosess til å beskrive en latent bevegelsesmodell og bruke den til å simulere baner numerisk. Denne formuleringen er langt mer fleksibel og er i teorien i stand til å uttrykke multimodale fordelinger for de predikerte banene. Å kombinere denne tilnærmingen med et prediksjonssystem basert på et Utvidet Kalman Filter (EKF) for å simulere baner fungerer bra så lenge banene er tilstrekkelig glatte, slik at en Taylor-approksimasjon av bevegelsesmodellen fungerer som en rimelig tilnærming. Disse antagelsene gjør imidlertid denne metoden mer skjør enn den første metoden.

Begge metodene testes grundig på et reelt AIS-datasett samlet fra Trondheimsfjorden i løpet av ett år, og den statistiske ytelsen til begge metodene sammenlignes. Konsistensen av usikkerhetsestimatene blir også testet for å undersøke om metodene er i stand til å nøyaktig kunne representere den underliggende usikkerheten.

Contents

Preface . . . iii
Abstract . . . v
Sammendrag . . . vii
Contents . . . ix
Figures . . . xiii
Tables . . . xv
Acronyms . . . xvii
1 Introduction . . . 1
1.1 The Need For Quantifying Uncertianty . . . 2
1.1.1 Gaussian Processes . . . 2
1.2 Thesis Structure . . . 3
2 Previous Work . . . 5
2.1 Data Driven Approaches . . . 7
3 Theory . . . 9
3.1 A short recap of necessary probability theory . . . 9
3.1.1 Probability Distributions . . . 9
3.1.2 Joint Distribution . . . 9
3.1.3 Conditional Distribution . . . 9
3.1.4 Marginal Distribution . . . 10
3.1.5 The Law of Iterated Expectations & Total Variance . . . 10
3.1.6 Central Limit Theorem . . . 10
3.1.7 Interpretation of Probability . . . 11
3.2 The Multivariate Gaussian Distribution . . . 11
3.2.1 Marginalization and conditioning . . . 11
3.3 Introduction to Gaussian Processes . . . 12
3.3.1 A quick note on vector notation . . . 13
3.3.2 Introduction to kernels . . . 13
3.3.3 Conditioning . . . 13
3.4 Vector-valued Gaussian Process . . . 15
3.5 Kernels . . . 16
3.5.1 Stationary Kernels . . . 16
3.5.2 Combining multiple kernels . . . 17
3.6 Hyperparameter selection . . . 17
3.6.1 Maximum Likelihood - The Marginal Likelihood . . . 17

3.7 Sampling from a Gaussian Process . . . 18
3.8 Standardization . . . 19
3.9 Approximate methods . . . 19
3.9.1 Reduced-rank approximation . . . 19
3.9.2 Sparse Variational Gaussian Process . . . 20
3.10 The Kalman Filter . . . 20
4 Historical AIS data . . . 23
4.1 Message Contents . . . 23
4.2 The Dataset . . . 23
4.3 Scenario . . . 24
4.3.1 Clustering . . . 26
4.4 Preprocessing . . . 26
4.4.1 From samples to trajectories . . . 27
5 Model trajectories directly using a Gaussian Process . . . 29
5.1 Method . . . 29
5.1.1 Key Assumption . . . 30
5.1.2 Choice of Kernel . . . 30
5.2 Implementation Details . . . 31
5.2.1 Approximate Gaussian Process . . . 31
6 GP-EKF: Non-parametric dynamic system using AIS tracking data . . . 33
6.1 Notation and variables . . . 34
6.2 Simulating Trajectories . . . 34
6.2.1 GP-EKF . . . 36
6.3 Incorporating vessel position . . . 37
6.3.1 Synthetic Likelihood . . . 39
6.3.2 Probabilistic Data Association Filter . . . 41
6.3.3 Tuning the parameters . . . 44
6.4 Simulating trajectories using Gaussian Process Sequential Monte Carlo . . . 46
6.5 Training Source . . . 47
6.6 Choice of kernel . . . 48
6.6.1 Optimizing the hyperparameters . . . 49
6.7 Summary . . . 49
7 Statistical Testing . . . 51
7.1 Method . . . 51
7.1.1 Trajectory Error . . . 51
7.1.2 Path Error . . . 52
7.1.3 Normalized Estimation Error Squared . . . 52
7.1.4 Comparing distributions - Boxplot . . . 52
7.1.5 Interpolation . . . 52
7.1.6 Baseline - Constant Velocity Model . . . 53
7.1.7 Selecting test cases . . . 53
7.1.8 Training Data . . . 54
7.2 Implementation . . . 54

7.2.1 Direct GP 55
7.2.2 GP-EKF 55
7.3 Results 56
7.3.1 GP-EKF: Finite Difference vs. COG/SOG from AIS 57
7.3.2 GP-EKF: Incorporating positions data 58
8 Discussion 65
8.1 Choice of kernel 65
8.2 Sensitivity to parameters 66
8.3 Incorporating position in GP-EKF predictions 66
8.4 Independent Outputs 67
8.5 Shared Kernel 67
8.6 Clustering 68
8.7 Branching Trajectories 68
8.8 Training Data 69
8.9 GP-EKF time dependency 69
8.10 Numerical issues 70
8.11 Computational Complexity 70
8.12 Using Gaussian Processes for trajectory prediction 70
9 Conclusion 73
9.1 Future Work 74
Bibliography 75
A GP-EKF example scenarios 79

Figures

3.1 Simple Gaussian Process example with zero-mean and RBF kernel with unit variance. The red line is the mean, while the red area is the 95% confidence interval. 15

4.1 Cumulative distribution of AIS messages for the N most frequent vessel's. 24
4.2 The available AIS data used in this thesis. 25

5.1 Example output from the Direct GP approach. 32

6.1 Illustrative example of the GP-EKF in practice 38
6.2 Illustrative example of the GP-EKF with the SL update procedure in practice. 42
6.3 Illustrative example of the GP-EKF with the PDAF update procedure in practice. 45
6.4 Trajectories simulated by sampling from $\vec{f}$. Notice how it is able to represent the multimodal trajectory distribution. The color represent the motion model's confidence when predicting the next step. 47

7.1 GP-EKF using finite differences and the COG and SOG from the AIS dataset on 350 trajectories. The finite difference approach performs slightly better, with lower median error and spread. 57
7.2 GP-EKF and Direct GP compared to a CVM on 350 straight-line and 350 curved trajectories. For straight-line trajectories, the CVM outperforms both the GP-EKF and the Direct GP, while the Direct GP yield slightly better performance than the GP-EKF. For curved trajectories, the CVM struggles as expected. Both the GP-EKF and the Direct GP yields far better results with lower error and lower spread, while Direct GP has the lowest overall trajectory error quartiles. Interestingly, both GP-EKF and Direct GP performs better on curved trajectories than straight-line trajectories, with lower overall error and spread. 59

7.3 GP-EKF with and without the SL and PDAF updates on 350 straight-line and 350 curved trajectories. While there are some slight differences, the PDAF and SL update does not appear to have any considerable effect on the trajectory errors. On straight-line trajectories, the results are almost identical across all three variants. On curved trajectories, the PDAF yields slightly lower error for the median and upper quartiles, though the difference is well within the margin of error. 60
7.4 A comparison of the NEES for the different methods. **(a)-(b)** compare the NEES for the Direct GP and the base GP-EKF to the theoretical quartile values for the $\mathcal{X}^2$ distribution (red dashed lines) and is intended as a base of comparison. **(c)-(d)** compare the difference in consistency between using COG/SOG and finite difference as input data. **(e)-(f)** compare the effect of the update steps for the GP-EKF. The theoretical quartiles are not included in the last row due to the large differences in scale, and the reader is advised to use the Direct GP and basic GP-EKF for comparison. 61

A.1 Case 1 80
A.2 Case 2 81
A.3 Case 3 82
A.4 Case 4 83
A.5 Case 5 84
A.6 Case 6: Notice how the PDAF and SL helps the GP-EKF select a single branch, insteading getting stuck in the middle. 85
A.7 Case 7: Notice how the training data contains faster vessels, making the GP-EKF overestimate the true velocity. 86
A.8 Case 8: Notice how the training data contains faster vessels, making the GP-EKF overestimate the true velocity. 87

Tables

4.1 Common content of AIS messages [4]. 24

5.1 Key variables 29

6.1 Key variables 34

7.2 Error summary for 350 straight-line trajectories. Mean and median summary statistics are calculated for the trajectory and path error at fixed timestamps. Linear interpolation is used between samples. Errors for short trajectories are not extrapolated, and therefore not included in the 20 and 25 minute bins. 62

7.3 Error summary for 350 curved trajectories. Mean and median summary statistics are calculated for the trajectory and path error at fixed timestamps. Linear interpolation is used between samples. Errors for short trajectories are not extrapolated, and therefore not included in the 20 and 25 minute bins. 63

7.4 NEES for 350 straight-line trajectories at fixed timestamps. Linear interpolation is used between samples. 64

7.5 NEES for 350 curved trajectories at fixed timestamps. Linear interpolation is used between samples. 64

Acronyms

AIS Automatic Identification System. v, xiii, xv, 3–7, 23–26, 29, 33, 47, 54, 55, 57, 65, 66, 69, 71

ASV Autonomous Surface Vehicles. v, 1, 2

COG Course Over Ground. xiii, xiv, 23, 24, 26, 29, 47–49, 53–55, 57, 58, 61, 69, 79–87

COLAV Collision Avoidance. v, 1, 2, 53

CVM Constant Velocity Model. xiii, 4, 53, 56, 59, 73, 74

EKF Extended Kalman Filter. v, 4, 33, 35–37, 41, 43, 49, 67

GP Gaussian Process. v, xiii, 2–6, 12, 13, 15–20, 29, 31–36, 48, 51, 53, 55, 56, 65–71, 73, 74

i.i.d Independent And Identically Distributed. 10, 13, 16, 19, 40, 41

MLE Maximum Likelihood Estimation. 18, 49, 55, 66, 69

MMSI Maritime Mobile Service Identity. 23, 24, 27, 53

NEES Normalized Estimation Error Squared. xiv, xv, 52, 56–58, 61, 64, 69

PDAF Probabilistic Data Association Filter. xiv, 41, 44, 49, 55, 56, 58, 60, 66, 67, 73, 74, 79, 85

PDF Probability Density Function. 9

PMF Probability Mass Function. 9

RBF Radial Basis Function. xiii, 5, 6, 15–17, 30, 31, 48, 55

SL Synthetic Likelihood. xiii, xiv, 39, 41, 42, 44, 49, 55, 56, 58, 60, 66, 67, 73, 74, 79–87

SOG Speed Over Ground. xiii, xiv, 23, 24, 26, 29, 47–49, 53–55, 57, 58, 61, 69, 80–87

SVGP Sparse Variational Gaussian Process. 20, 31, 70

Chapter 1

Introduction

Humans in general are lazy and have always strived to avoid repeating tasks. Time is better spent doing a more rewarding task, leaving machines to do tasks that are not worth spending human resources. Automation has typically been chiefly focused on industrial applications where there is a direct economic incentive to automate tasks. The processes are also isolated and highly repeatable, which avoids the need for human interaction as much as possible. While the idea of replacing humans in more complicated situations outside of the factory floor is not new, it is only in the last few decades that machines have become sophisticated enough to actually replace their human alternatives in more everyday tasks. Outside of the constrained environment of the factory floor, machines need to be able to understand and adapt to the wide range of different scenarios that may occur in the chaotic environment designed for and by humans.

This thesis will focus on one such automation task, namely *Autonomous Surface Vehicles (ASV)*. Humans are prone to loss of focus, tiredness, and limited attention span. Combined with large and powerful machines, they are a potential danger to both themselves and others. It is estimated that over 75% of maritime accidents are attributed to human errors [1], motivating the need for automated alternatives to human captains. While altogether avoiding humans at sea would significantly reduce accidents, it is simply unrealistic to achieve. Hence, autonomous vessels need to learn to understand and cooperate with other human-driven vessels. However, it is easy to take humans' remarkable ability to understand patterns for granted. While people certainly make mistakes, they can understand highly complex scenarios and infer likely outcomes given their prior experiences in similar situations. An ASV will need similar abilities in order to operate without putting humans at risk in a reliable way. A key aspect to solving this problem is a robust *Collision Avoidance (COLAV)* system, which can both reactively and proactively avoid collisions. In order to avoid high-risk situations proactively, the COLAV needs accurate situational awareness, which includes both information about the current situation as well as how the future might unfold. This thesis will focus on the latter, namely how to predict the future behavior of nearby vessels.

Knowing the future behavior of surrounding vessels, even only probabilisti-

cally, allows the COLAV system to actively avoid scenarios with increased risk of collision. A simple solution is to assume near a constant speed and course of any vessel of interest and is a commonly used assumption for obstacle models in COLAV research [2]. While certainly a reasonable solution on the open sea, where vessels spend a significant portion of the time in straight-line trajectories, it quickly fails when vessels navigate constrained areas such as fjords or close to shore where more complicated maneuvering is necessary. Considering the usual prediction horizon of $5 - 15$ minutes into the future for the typical COLAV application [3, 4], assuming near-constant speed and course is not sufficient. However, the vessels still tend to follow typical traffic lanes and behave somewhat predictably. Combined with modern machine learning, the question becomes how this data can be utilized to improve the prediction of future trajectories, and thereby allowing ASVs to avoid high-risk COLAV scenarios proactively. Ideally, the ASV should learn from the available data and use it to recognize patterns to infer likely outcomes.

1.1 The Need For Quantifying Uncertianty

Predicting the trajectory of a vessel based on the limited information available is no easy task. Even given historical trajectories of the same vessel, there is no way to guarantee that the vessel will behave identically in the future. In the end, a vessel can take an indefinite amount of different trajectories to reach a given destination.

The ability of a model to express uncertainty is therefore crucial. Ideally, the model should make accurate predictions while also quantifying the wide range of different trajectories a vessel might follow. The need for quantifying uncertainty lends itself nicely to a Bayesian approach, as the uncertainty becomes a first-class citizen in the model. Such an approach allows the COLAV to express *beliefs* about the possible outcomes and does not hide the stochastic nature of the problem it is trying to solve.

1.1.1 Gaussian Processes

The *Gaussian Process (GP)* is a stochastic process based on the multivariate Gaussian Distribution. It is an interpolation method with use cases in a wide range of fields, such as environmental science [5], medicine [6, 7], cognitive research [8] and optimization [9] to name a few. It was first used by Danie G. Krige in his master thesis [10] where he used GPs to find the most likely spatial distribution of gold based on samples from only a few boreholes. GP are therefore sometimes referred to as *kriging* in the literature, especially for geospatial statistics or related fields. The GP is a non-parametric approach, and it is well suited for approximating black-box systems. While comparable to other interpolation methods such as splines, GPs has the added benefit of expressing uncertainty based on the available data. Therefore, it is commonly interpreted as a statistical distribution over

functions. Prior beliefs can be conditioned on available data to form posterior beliefs about the true underlying function $f(x)$. This Bayesian interpretation, in combination with its flexibility as a non-parametric approach, makes GPs a very powerful tool. GPs will be covered in greater detail in Chapter 3.

The functional interpretation appears to be a good fit for the long-term prediction problem. By considering a vessel's trajectory as a function of time $f(t)$, the GP framework should, in theory, then be able to express the vessel's trajectory, including the inherent uncertainty of predicting future trajectories. In practice, however, the computational complexity of GPs may limit their usability.

This raises a few research questions which this thesis will attempt to answer:

1. How can GPs be used to model the long-term vessel trajectory of new vessels?
2. Is it computationally feasible to use GPs in the context of AIS Big Data?
3. Is a GP able to provide consistent uncertainty that reflects the true error rate?

Especially question 3 will receive emphasized attention in this thesis. If the GP cannot provide consistent uncertainty estimates, the method offers little additional value over more straightforward data-driven approaches.

This thesis will propose two alternative solutions to the long-term trajectory prediction problem, both based on the GP framework. The first method will attempt to model the trajectory directly as a function of time and use a GP to express this function. The other solution is an indirect approach attempting to learn an unknown motion model and then simulate the trajectory numerically. Both methods will be rigorously tested on a real-world AIS dataset.

1.2 Thesis Structure

The thesis will first review existing work on applying the GP framework for trajectory prediction in Chapter 2. As there is little work available on using GPs with AIS data, solutions from related fields, as well as some approaches not based on GPs, will also be presented as inspiration.

The Gaussian Process will then be introduced in greater detail in Chapter 3, along with other necessary theory. The chapter is inspired by the excellent introduction from [11], and the reasoning, intuition, and mathematics behind the Gaussian Process will be explored to lay the necessary foundation for the rest of this thesis. A basic understanding of calculus, statistics, and probability theory is assumed, but all necessary subjects are otherwise introduced.

The thesis then moves on to Chapter 4 which offers an introduction to AIS and the dataset used in this thesis. Any preprocessing and filtering of the dataset are included in this chapter.

Once the foundation for the GP and AIS have been covered, Chapter 5 will apply the GP framework to model a vessel's trajectory directly as a function of time. This method will be referred to as the *direct GP approach* throughout the rest

of this thesis. Chapter 6 then proposes a more indirect approach, using the GP framework to model an unknown motion model describing the trajectory gradients. The trajectories are simulated numerically using an EKF prediction scheme. This method is inspired by the work in [12, 13], and can be seen as an extension of the Kalman filter. Familiarity with sensor-fusion methods is beneficial but not required to follow the derivations.

Once the methods are introduced in their respective chapters, the thesis will move on to statistical testing in Chapter 7. This chapter will cover the implementation details and present the results from rigorous statistical testing to see how the methods perform on real-world AIS data. The results are compared to a simple CVM model, which is used as a baseline for comparison.

Chapter 8 will then discuss the results from statistical testing, as well as address theoretical concerns regarding the proposed methods. Finally, Chapter 9 will summarize the findings before finishing with possible extensions to this work.

Chapter 2

Previous Work

There is currently limited work available on using GPs on AIS data.

Kowalska and Peel propose a method for anomaly detection of moving vessels in [14]. The approach uses a GP to learn vessels' normal behavior in an area, which is then used to detect abnormal behavior. The work further addresses the computational challenges when using large datasets and proposes an active learning approach that iteratively adds new training samples to select the optimal training sample. A relatively small sample size is then used to represent the entire dataset. Active learning is computationally feasible by updating the Cholesky decomposition at each step instead of recomputing the entire decomposition. While the approach works well for anomaly detection, it was not intended for predicting future trajectories, only to classify existing trajectories.

Rong *et al.* address the problem of trajectory uncertainty in [15]. The paper proposes a method for probabilistic trajectory prediction using GPs, where different distributions describe the lateral and longitudinal directions of the trajectory. The parameters of the distributions are learned offline based on historical AIS data. They are then applied in real-time by adding new observations through a Cholesky update to avoid recomputing the decomposition. The model uses the common RBF kernel, and the hyperparameters are selected as the median of the maximum likelihood estimates for each unique vessel. The prediction is then conditioned on historical AIS samples for a given vessel. A case study is performed where the model is trained and tested on three months of AIS data along a smooth traffic lane. The methods yield high prediction accuracy even for long prediction horizons while still applying to real-time applications. Only a highly smooth traffic lane with little curvature is used, and the paper does not mention the performance of curved trajectories. As the model is only conditioned on previous samples of a given vessel, it is unlikely that the model can predict upcoming turns.

However, there is extensive work on using GPs for trajectory prediction outside of the maritime field. Goli *et al.* [16] utilizes GPs for long-term trajectory prediction for collision avoidance in a connected vehicle environment. The vehicles are assumed to share position through vehicle-to-vehicle communication, somewhat in a similar fashion to AIS for maritime vessels. The paper uses a GP to

learn a motion model from historical data, mapping a vehicle's current position to a trajectory derivative. The historical data is clustered into a finite number of clusters, where the trajectories in each cluster are assumed to have similar properties. The paper utilizes K-means clustering [17] to group trajectories based on the first and last position of a trajectory. For each cluster, independent GPs are then fitted for each of the two coordinate axes, using independent RBF kernels and zero mean priors. While this method should apply to AIS data, there are a few key distinctions to keep in mind:

1. In this paper, the trajectory derivatives are calculated using finite difference with a sampling interval of 0.1 seconds. The typical sampling interval for AIS data is in the range of several seconds or even minutes, so relying on numerical derivatives might be challenging.
2. The vehicles' trajectories are from an intersection, and the roads constrain the vehicles' behavior. There is therefore only a finite number of route options that a vehicle may take.

A similar approach is used by Ellis *et al.* [12] to predict the trajectory of pedestrians tracked using computer vision. The paper also learns a dynamical model using GPs, but it utilizes a Bayes Filter framework proposed by Ko and Fox [13] to simulate the trajectories. Two different approaches were used:

1. By assuming $p(\boldsymbol{x})$ is always uni-modal and Gaussian, the GP-EKF introduced in [13] was used to simulate the trajectory for multiple timesteps, using the dynamical GP model as the prediction model. The GP-EKF is based on the Extended Kalman Filter, where the prediction model is learned from data using a GP. This formulation is unable to express multimodal uncertainty.
2. To retain the inherent multimodality, a sequential Monte-Carlo approach (i.e., the prediction step of a particle filter) was used to keep track of multiple modes (i.e., branching trajectories) at the cost of computational complexity.

Of some more theoretical work, Girard *et al.*[18] discuss how GPs can be used when the function inputs $\boldsymbol{x}$ are latent variables. It is used in the context of multi-step ahead time series forecasting, where the GP is recursively evaluated using the output at one step as the input in the next. As the true posterior distribution of this recursive formulation is intractable, a Gaussian approximation is applied in combination with a Taylor approximation.

Another common approach for trajectory prediction using AIS is based on clustering methods. This approach can typically be divided into four steps [19]:

1. Cluster trajectories based on historical data
2. Classify new target vessels into the appropriate cluster
3. Generate a representative trajectory for the given cluster
4. Predict the movement using the representative trajectory

Examples of this approach can be found in [20], [21] and [22].

Traditional clustering methods, such as k-means or DBSCAN, tend to focus

on clustering point values. In the context of trajectory prediction, the trajectories would then be clustered as a whole. The trajectory clustering algorithm TRACLUS was therefore introduced by [23], where it was applied to hurricane trajectory and animal movement data. The key observation was that trajectories might have portions that share typical behavior, while entire trajectories might still differ. TRACLUS allows for clustering of trajectories based on common sub-trajectories. It works by partitioning the trajectories into smaller line segments and grouping similar segments into clusters.

2.1 Data Driven Approaches

Hexeberg *et al.* [24] introduced the *Single Point Neighborhood Search* (SPNS), a purely data-driven approach. It deviates from the clustering-based methods as it estimates the future course and speed at each prediction time based on historical AIS data. Historical AIS samples in the vicinity of the target vessel with a similar course are used to calculate the median course and velocity, which is then used to simulate the trajectory one step forward in time before the whole process is repeated.

The SPNS was later extended by Hexeberg [4] to handle two of the main shortcomings [19], mainly handling branching traffic-lanes and to better estimate the uncertainty. The result was the Neighbor Course Distribution Method (NCDM). The NCDM extends the SPNS by representing possible trajectories in a tree structure, where each trajectory is computed similarly to the SPNS. This method was further developed by Dalsnes *et al.*[19] by introducing a Gaussian Mixture Model (GMM) to represent a vessel's position in a probabilistic framework.

Chapter 3

Theory

3.1 A short recap of necessary probability theory

The reader is assumed to be already comfortable with basic probability theory. A more detailed introduction has already been covered in a previous specialization project [25], but some concepts will nevertheless be reintroduced here mainly to specify the mathematical notation used throughout this thesis.

3.1.1 Probability Distributions

The notation $p(X)$ is used to denote the probability distribution of the random variable X, regardless of X being discrete or random. Whether $p(X)$ refers to the *Probability Density Function (PDF)* for continuous random variables or *Probability Mass Function (PMF)* for discrete random variables therefore depends on the context. For the probability of a specific event occurring, the notation $\Pr\{X = x\}$ and $\Pr\{X \leq x\}$ will be used instead. The result is always a real number, i.e. $\Pr\{\cdot\} \in [0, 1]$.

3.1.2 Joint Distribution

The notation $p(X, Y) = p(X \cap Y)$ is used to denote the joint probability of X and Y. It is the probability of both events occurring at once.

3.1.3 Conditional Distribution

The notation $X|Y$ is used to express the event X occurring given the known event occurrence Y. In the context of probability distributions, $p(X|Y)$ denotes the probability of X occurring, given that the event Y has already occurred. It is perhaps easier to read as "the current belief about a quantity X given a known value of Y", as Y will in this thesis usually be some parameter rather than a discrete event.

As this thesis will use a Bayesian interpretation of probability, the notation may also be used to express the parameters of a distribution, for instance $\mathcal{N}(x \mid \mu, \sigma^2)$ is the PDF for the Gaussian distribution x with mean μ and variance σ.

3.1.4 Marginal Distribution

The integral notation will be used to denote the marginal distribution for both discrete and continuous random variables. For discrete variables, the integral is implicitly replaced by a sum.

$$p(X) = \int_Y p(X \cap \boldsymbol{Y}) d\boldsymbol{Y} = \int_Y p(X|\boldsymbol{Y}) p(\boldsymbol{Y}) d\boldsymbol{Y} \tag{3.1}$$

The last equality is commonly referred to as *The Law of total probability* and allows a complex distribution $p(X)$ to be expressed in several simpler components.

3.1.5 The Law of Iterated Expectations & Total Variance

The *Law of Iterated Expecations* states that

$$\mathbb{E}[X] = \mathbb{E}\Big[\mathbb{E}[X|Y]\Big] \tag{3.2}$$

if all the expectations exist. Similarily, the *Law of Total Variance* states that

$$\mathbb{V}[X] = \mathbb{E}\Big[\mathbb{V}[X|Y]\Big] + \mathbb{V}\Big[\mathbb{E}[X|Y]\Big] \tag{3.3}$$

These results are useful when computing the expected value and variance of complex distributions, as they allow the computations to be separated into simpler, conditional computations. Proofs are available in [26].

3.1.6 Central Limit Theorem

The *central limit theorem* should be familiar to most readers, and it is essential to the derivations in a later chapter. Consider a set of *Independent And Identically Distributed (i.i.d)* random variables X_i following any distribution with mean μ and variance σ^2. The sum of these variables

$$S_N = \sum_{i=0}^{N-1} X_i \tag{3.4}$$

can be shown to approach the Gaussian distribution

$$\lim_{N \to \infty} p(S_N = s) = \mathcal{N}(s \mid N\mu, N\sigma^2) \tag{3.5}$$

as N increases, even if the original variables are not Gaussian distributed [17]. For a finite N it can be used to approximate the sum of i.i.d random variables as a Gaussian distribution.

3.1.7 Interpretation of Probability

This thesis will heavily rely on a Bayesian interpretation of probability. Bayesian Statistics is extensively covered in the specialization project [25], and only a short introduction is included here. Without going into unnecessary philosophical details, the Bayesian view interprets probabilities as beliefs rather than relative frequencies[1]. The benefit of the Bayesian interpretation is that it can naturally be used to model uncertainty of events that cannot easily be expressed as repeated trials. Examples include one-off events and parameter estimation of fixed but unknown quantities.[2] [17].

Central to the Basian view of probability is *Bayes rule*, which is used to update the prior beliefs $\boldsymbol{\theta}$ when observing new data $\mathcal{D}$. Bayes rule is given by Equation (3.6) where $p(\boldsymbol{\theta})$ is the prior belief about $\boldsymbol{\theta}$ before observing $\mathcal{D}$, $p(\mathcal{D}|\boldsymbol{\theta})$ is the likelihood of observing $\mathcal{D}$ given a known value for $\boldsymbol{\theta}$, $p(\mathcal{D})$ is a normalization constant and $p(\boldsymbol{\theta}|\mathcal{D})$ is the *posterior* beliefs about $\boldsymbol{\theta}$ after observing $\mathcal{D}$.

Bayes Rule

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\boldsymbol{\theta} \cap \mathcal{D})}{p(\mathcal{D})} = \frac{p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{D})} \tag{3.6}$$

3.2 The Multivariate Gaussian Distribution

The Gaussian Distribution is one of the most used distributions in statistics [17] and generalizes well for multivariate variables. The pdf for the D dimensional multivariate Gaussian is given by Equation (3.7) [11, 17].

$$\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) \triangleq \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\intercal \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) \right] \tag{3.7}$$

3.2.1 Marginalization and conditioning

Consider the joint multivariate Gaussian distribution for two (potentially vector-valued) variables $\boldsymbol{x}$ and $\boldsymbol{y}$.

$$p(\boldsymbol{x}, \boldsymbol{y}) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{\mu}_x \\ \boldsymbol{\mu}_y \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{xx} & \boldsymbol{\Sigma}_{xy} \\ \boldsymbol{\Sigma}_{yx} & \boldsymbol{\Sigma}_{yy} \end{bmatrix} \right) \tag{3.8}$$

The marginal and posterior conditional distribution are given by Equation (3.9) and Equation (3.10) respectively [11], and will be used extensively throughout

[1] The interpretation of probabilities as the relative frequency of outcomes from repeated trials is what is often taught in statistics courses and is called the *frequentist* interpretation.

[2] As a thought experiment, let us say you want to determine the number of trees on the planet. There is a fixed amount of trees, but the actual number is unknown as you cannot count every single one. There is nothing uncertain about the number of trees itself, only about your beliefs, which are uncertain due to incomplete knowledge.

this thesis. Note that the marginal distribution does not actually require any calculations and is found by selecting the corresponding values from $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$.

Marginal Distribution

$$p(\boldsymbol{x}) = \int_{\boldsymbol{y}} p(\boldsymbol{x}, \boldsymbol{y}) d\boldsymbol{y} = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_x, \boldsymbol{\Sigma}_{xx}) \tag{3.9}$$

Posterior Conditional Distribution

$$p(\boldsymbol{x}|\boldsymbol{y}) = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{x|y}, \boldsymbol{\Sigma}_{x|y}) \tag{3.10a}$$
$$\boldsymbol{\mu}_{x|y} = \boldsymbol{\mu}_x + \boldsymbol{\Sigma}_{xy}\boldsymbol{\Sigma}_{yy}^{-1}(\boldsymbol{y} - \boldsymbol{\mu}_y) \tag{3.10b}$$
$$\boldsymbol{\Sigma}_{x|y} = \boldsymbol{\Sigma}_{xx} - \boldsymbol{\Sigma}_{xy}\boldsymbol{\Sigma}_{yy}^{-1}\boldsymbol{\Sigma}_{yx} \tag{3.10c}$$

3.3 Introduction to Gaussian Processes

This introduction is heavily inspired by [11], where more details can be found for those interested. This introduction will only consider the scalar case, and the discussion of vector-valued functions is delayed until Section 3.4. Rest assured, this introduction easily extends to vector-valued functions!

A Gaussian Process (GP) can formally be defined as Definition 1.

Definition 1. *A Gaussian Process is a collection of random variables, any finite number of which has a joint Gaussian Distribution.*

In this thesis, a more specific definition is adopted in order to interpret GPs as a statistical distribution over functions. A GP for a random function $f \triangleq f(\boldsymbol{x})$ is fully specified by its mean function $m(\boldsymbol{x})$ and covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$.

$$f \sim GP(\, m(\boldsymbol{x}),\, k(\boldsymbol{x}, \boldsymbol{x}')\,) \tag{3.11}$$

This interpretation of a GP might seem a bit odd at first. It may help to think of functions as infinitely long vectors containing the function values for all possible inputs. The key observation is that the marginal distribution $p(\boldsymbol{x})$ of a multivariate Gaussian distribution $p(\boldsymbol{x}, \boldsymbol{y})$ is another Gaussian distribution that is completely independent of $\boldsymbol{y}$, as expressed in Equation (3.9). Any variables not of interest can therefore be easily marginalized away. Any GP by Definition 1 can therefore be viewed as the finite marginal distribution of an infinite Gaussian Distribution, jointly describing the values of f at all possible inputs $\boldsymbol{x}$. In the end, a GP is nothing more than a joint Gaussian Distribution with a fancy interpretation.

3.3.1 A quick note on vector notation

Different notations for vector values are used to differentiate between three common cases:

Bold letters are used to indicate a single, multivariate value. $f(\boldsymbol{x})$ is the scalar function f evaluated at the multivariate input $\boldsymbol{x}$. In practice, this implies that the input is a column vector $\boldsymbol{x} \in \mathcal{R}^M$ with $M \geq 2$ elements.

Capital letters such as X, are used to indicate multiple (potentially multivariate) samples. $f(X)$ is the scalar function f evaluated at each row in X. In practice this implies that the input $X \in \mathcal{R}^{NxM}$ with $N \geq 2$ rows (samples) and $M \geq 1$ columns (dimensions).

Vector Arrows such as $\vec{f}$ are used to denote vector-fields (i.e. vector-valued functions). This will be useful for vector-valued GPs as it allows the distinction between shorthand notations such as $\boldsymbol{f}_* = f(X_*)$, $\vec{f}_* = \vec{f}(\boldsymbol{x}_*)$ and $\vec{\boldsymbol{f}}_* = \vec{f}(X_*)$.

3.3.2 Introduction to kernels

The covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$ determines the similarity between two different points $\boldsymbol{x}$ and $\boldsymbol{x}'$. These covariance functions will be referred to as *kernels*, which maps the input space to a *feature-space* [11]. The output of the kernel is a value describing the similarity (i.e., covariance) between the two inputs. Kernel functions are discussed in greater detail in Section 3.5.

The kernel must be symmetric and positive definite to produce a valid covariance matrix, which requires that

$$k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}', \boldsymbol{x}) \tag{3.12}$$

The covariance matrix $K(X,X)$ is the result of calling $k(\cdot,\cdot)$ on all pairs of inputs, i.e.

$$K(X,X)_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j) \quad \forall i,j \tag{3.13}$$

3.3.3 Conditioning

So far, only the prior distribution for f has been specified. As a GP is by definition a multivariate Gaussian Distribution, the posterior conditional in Equation (3.10) can be used to condition f on observed values. A simple GP $f(\boldsymbol{x})$ with mean $m(\boldsymbol{x})$ and kernel $k(\boldsymbol{x}, \boldsymbol{x}')$ will be used as an example on conditional GPs.

Let $\boldsymbol{f}_* \triangleq f(X_*)$ denote the function evaluated at test points X_*. The prior distribution over $\boldsymbol{f}_*$ is shown in Figure 3.1a. From a Bayesian perspective, the goal is to update the prior beliefs about $\boldsymbol{f}_*$ using the (potentially) noisy observations $\boldsymbol{y} = f(X) + \epsilon$ at multiple inputs X to get the posterior belief $p(\boldsymbol{f}_* | X, \boldsymbol{y}, X_*)$. The noise term ϵ is considered i.i.d and distributed according to $\epsilon \sim \mathcal{N}(0, \sigma^2)$.

The joint distribution of $\boldsymbol{y}$ and $\boldsymbol{f}_*$ is given by

$$p(\boldsymbol{y}, \boldsymbol{f}_*) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \middle| \begin{bmatrix} m(X) \\ m(X_*) \end{bmatrix}, \begin{bmatrix} K(X,X) + \sigma^2 I & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} \right) \tag{3.14}$$

and the posterior distribution $p(\boldsymbol{f}_*|\boldsymbol{y})$ is computed using the posterior conditional distribution in Equation (3.10).

$$p(\boldsymbol{f}_*|\boldsymbol{y}) = \mathcal{N}(f \mid \boldsymbol{\mu}_{f_*|y}, \boldsymbol{\Sigma}_{f_*|y}) \tag{3.15a}$$

$$\mathbb{E}[\boldsymbol{f}_*] = \boldsymbol{\mu}_{f_*|y} = m(X_*) + K(X_*,X)\left(K(X,X) + \sigma^2 I\right)^{-1}(\boldsymbol{y} - m(X)) \tag{3.15b}$$

$$\mathbb{V}[\boldsymbol{f}_*] = \boldsymbol{\Sigma}_{f_*|y} = K(X_*,X_*) - K(X_*,X)\left(K(X,X) + \sigma^2 I\right)^{-1} K(X,X_*) \tag{3.15c}$$

As the notation quickly gets messy for the general case, the shorthand notation for evaluating $f(\boldsymbol{x}_*)$ at a single test point is introduced as well. To follow the convention used by [11], $\boldsymbol{k}_* \triangleq K(X, \boldsymbol{x}_*)$ is used to denote the vector of covariances calculated between the test point and each of the training samples. The notation $K \triangleq K(X,X)$, $\bar{f}_* \triangleq \mathbb{E}[f_*]$, and $\boldsymbol{\alpha} = (K + \sigma^2 I)^{-1}\left(\boldsymbol{y} - m(X)\right)$ is also added to simplify the equations. Using this shorthand notation for a single test case, Equation (3.15) boils down to Equation (3.16).

$$\bar{f}_* = \mathbb{E}[f_*] = m(\boldsymbol{x}_*) + \boldsymbol{k}_*^\intercal (K + \sigma^2 I)^{-1}(\boldsymbol{y} - m(X)) \tag{3.16a}$$

$$= m(\boldsymbol{x}_*) + \boldsymbol{k}_*^\intercal \boldsymbol{\alpha} \tag{3.16b}$$

$$\mathbb{V}[f_*] = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_*^\intercal K^{-1}\, \boldsymbol{k}_* \tag{3.16c}$$

In practice, computing the inverse $K(X,X)$ becomes expensive for an increasing number of samples. To avoid numerical instability, using the *Cholesky Decomposition* is usually preferred. The Cholesky Decomposition forms a new lower-triangular matrix L such that $K = LL^\intercal$, assuming K is symmetric and positive definite, and it is considered extremely numerically stable [11]. Equation (3.15) can then be computed using L. The diagonal entry $\sigma^2 I$ added to the kernel matrix is intended to model noisy observations, while in practice it also improves the numerical stability. A small value is therefore recommended even if the observations are noise-free [27].

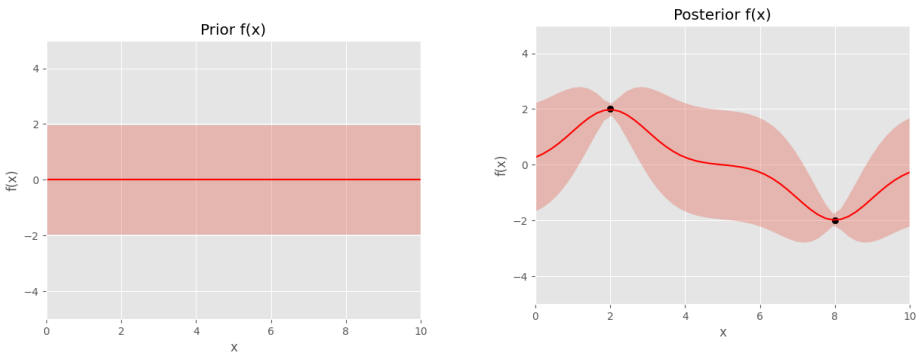
$$\begin{aligned} \mathbb{E}[\boldsymbol{f}_*] &= m(X_*) + K(X_*,X)\,(LL^\intercal)^{-1}\,(\boldsymbol{y} - m(X)) \\ &= m(X_*) + K(X_*,X)\,\underbrace{\left[(L^\intercal)^{-1}(L)^{-1}\,(\boldsymbol{y} - m(X))\right]}_{\boldsymbol{\alpha}} \end{aligned} \tag{3.17a}$$

$$\begin{aligned} \mathbb{V}[\boldsymbol{f}_*] &= K(X_*,X_*) - K(X_*,X)\,(LL^\intercal)^{-1}\,K(X,X_*) \\ &= K(X_*,X_*) - K(X_*,X)\,(L^\intercal)^{-1}(L)^{-1}\,K(X,X_*) \\ &= K(X_*,X_*) - \underbrace{\left[(L)^{-1}\,K(X,X_*)\right]^\intercal}_{v^\intercal}\,\underbrace{\left[(L)^{-1}\,K(X,X_*)\right]}_{v} \end{aligned} \tag{3.17b}$$

The whole procedure boils down to Algorithm 1 as proposed by [11]. A simple GP before and after conditioning is shown in Figure 3.1

Algorithm 1 Gaussian Process Prediction

procedure GP-PREDICT(X_*, $\boldsymbol{y}$, k, X)
  $L = cholesky\big(K(X,X) + \sigma I\big)$
  $\boldsymbol{\alpha} = L^\intercal \backslash (L \backslash \boldsymbol{y})$
  $\boldsymbol{v} = L \backslash K(X, X_*)$
  $\mathbb{E}[\boldsymbol{f}_*] = m(X_*) + K(X_*, X)\boldsymbol{\alpha}$
  $\mathbb{V}[\boldsymbol{f}_*] = K(X_*, X_*) - \boldsymbol{v}^\intercal \boldsymbol{v}$
  return $\mathbb{E}[\boldsymbol{f}_*]$, $\mathbb{V}[\boldsymbol{f}_*]$
end procedure



(a) Prior

(b) Posterior after observing the function at two different inputs (black dots).

Figure 3.1: Simple Gaussian Process example with zero-mean and RBF kernel with unit variance. The red line is the mean, while the red area is the 95% confidence interval.

3.4 Vector-valued Gaussian Process

GPs can easily be extended for vector-valued functions by simply considering the joint distribution of each function component as in Equation (3.18).

$$\vec{f}(\boldsymbol{x}) = \begin{bmatrix} f_x(\boldsymbol{x}) \\ f_y(\boldsymbol{x}) \end{bmatrix} \sim \mathrm{GP}\Big(\begin{bmatrix} m_x(\boldsymbol{x}) \\ m_y(\boldsymbol{x}) \end{bmatrix}, \begin{bmatrix} k_{xx}(\boldsymbol{x}, \boldsymbol{x}') & k_{xy}(\boldsymbol{x}, \boldsymbol{x}') \\ k_{xy}(\boldsymbol{x}, \boldsymbol{x}')^\intercal & k_{yy}(\boldsymbol{x}, \boldsymbol{x}') \end{bmatrix} \Big) \tag{3.18}$$

Only independent output dimensions, i.e. $k_{xy} = k_{yx} = 0$ will be considered in this thesis to reduce the number of complicating factors. This is, however, not without consequences as it assumes zero covariance between the two outputs. This issue is revisited in Chapter 8.

3.5 Kernels

This section will introduce some relevant kernels for this thesis. Many more kernels are available in the literature [11].

3.5.1 Stationary Kernels

Stationary kernels are kernels which only depends on $\boldsymbol{r} = \boldsymbol{x} - \boldsymbol{x}'$ and is usually specified as a function of a single variable.

Constant Kernel

As the name implies, the constant kernel is a kernel that is independent of the input. It is typically used as a scaling parameter in combination with other kernels.

$$k(\cdot) = \sigma^2 \tag{3.19}$$

White kernel

The *White Kernel* is useful for modeling whitenoise in a system as i.i.d [27]. The white kernel is given by

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \delta_{ij}\sigma^2 \tag{3.20}$$

where δ_{ij} is the Kronecker-delta which is 1 if $i = j$ and 0 otherwise. This is the same as the noise term added in Algorithm 1.

Radial Basis Function

The *Radial Basis Function (RBF)* kernel, also referred to as *squared exponential kernel*, is one of the most frequently used kernels, and is given by the covariance function in Section 3.5.1. The scaling parameter l is the *charactheristic length scale* and can intuitively be thought of as a smoothness parameter. This kernel yields infinitely differentiable functions, meaning that any function drawn from a GP with this kernel is very smooth[11].

$$k(\boldsymbol{r}) = \exp\Big\{-\frac{||\boldsymbol{r}||^2}{2l^2}\Big\} \tag{3.21}$$

Matérn class

The *Matérn Class* of kernels is given by the covariance function in Equation (3.22), where K_ν is the *modified Bessel function*.

$$k(\boldsymbol{r}) = \frac{2^{1-\nu}}{\Gamma(\nu)}\Big(\frac{\sqrt{2\nu}||\boldsymbol{r}||}{l}\Big)^{\nu} K_\nu\Big(\frac{\sqrt{2\nu}||\boldsymbol{r}||}{l}\Big) \tag{3.22}$$

The parameter $\nu > 0$ determines the smoothness, where:

- $\nu = \frac{1}{2}$ yields the *Ornstein-Uhlenbeck Process* and functions, that when drawn from a GP, are continious, but not differentiable. The kernel is equivalent to
$$k(\boldsymbol{r}) = \exp\{-\frac{||\boldsymbol{r}||}{l}\}$$
- $\nu = \frac{3}{2}$ yields functions that, when drawn from a GP, are continous and once-differentiable. The kernel is equivalent to
$$k(\boldsymbol{r}) = \big(1 + \frac{\sqrt{3}||\boldsymbol{r}||}{l}\big)\exp\{-\frac{\sqrt{3}||\boldsymbol{r}||}{l}\}$$
- $\nu = \frac{5}{2}$ yields functions, that when drawn from a GP, are continous and twice differentiable. The kernel is equivalent to
$$k(\boldsymbol{r}) = \big(1 + \frac{\sqrt{5}||\boldsymbol{r}||}{l} + \frac{5||\boldsymbol{r}||^2}{3l^2}\big)\exp\{-\frac{\sqrt{5}||\boldsymbol{r}||}{l}\}$$

More generally, the functions drawn from a GP with a Matérn class kernel is k-times differentiable if and only if $\nu > k$[11]. The Matérn class of kernels is argued to be a better choice than the RBF kernel for many physical systems, as the infinitely smooth function generated by RBF is too smooth [11]. Further mathematical details can be found in [11, sec. 4.2] as it is outside the scope of this thesis.

3.5.2 Combining multiple kernels

Kernels can be mixed and matched through multiplication and addition, where the behavior of the individual kernels can be combined to describe more complex functions. A simple example is using the constant kernel to scale the covariance of other kernels. Different kernels can also be used for each input dimension.

3.6 Hyperparameter selection

Manually tuning the kernel hyperparameters is tedious and time-consuming, motivating the need for automatic selection of parameters. This section will discuss an automated approach for hyperparameter selection with GPs.

3.6.1 Maximum Likelihood - The Marginal Likelihood

The *marginal likelihood*, which is the likelihood of observing a set of given observations $\boldsymbol{y}$, conditioned on a GP with kernel parameters $\boldsymbol{\theta}$ and inputs X, is given by

$$p(\boldsymbol{y} \mid X, \boldsymbol{\theta}) = \mathcal{N}\big(\boldsymbol{y} \mid \boldsymbol{m}(X), K_{\boldsymbol{\theta}}(X,X)\big) \tag{3.23}$$

and can be used to obtain a *Maximum Likelihood Estimation (MLE)* estimate of the parameters $\boldsymbol{\theta}$. Defining $\tilde{\boldsymbol{y}} \triangleq \boldsymbol{y} - \boldsymbol{m}(X)$ and taking the logarithm yields Equation (3.24)

Log Marginal Likelihood

$$\log p(\boldsymbol{y} \mid X, \boldsymbol{\theta}) = -\frac{1}{2}\tilde{\boldsymbol{y}}^{\intercal} K_{\boldsymbol{\theta}}(X,X)^{-1}\tilde{\boldsymbol{y}} - \frac{1}{2}\log|K_{\boldsymbol{\theta}}(X,X)| \ldots - \frac{n}{2}\log(2\pi) \tag{3.24}$$

where the optimal hyperparameters $\boldsymbol{\theta}_{\text{ML}}$ can be found by maximizing this quantity[3], i.e.

$$\boldsymbol{\theta}_{\text{ML}} = \arg\max_{\boldsymbol{\theta}} \log p(\boldsymbol{y} \mid X, \boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \big(-\log p(\boldsymbol{y} \mid X, \boldsymbol{\theta})\big) \tag{3.25}$$

The name marginal likelihood comes from the fact that the latent function $\boldsymbol{f}$ is marginalized out, i.e. the parameters are optimized over all possible latent functions $\boldsymbol{f}$.

$$p(\boldsymbol{y} \mid X, \boldsymbol{\theta}) = \int_{f} p\big(\boldsymbol{y} \mid \boldsymbol{f}\big) p\big(\boldsymbol{f} \mid m(X), K(X,X)\big) d\boldsymbol{f} \tag{3.26}$$

The marginal likelihood is somewhat resilient to overfitting, as it naturally incorporates a trade-off between model complexity and model fit. However, the optimization step always imposes the risk of overfitting, especially if there are many hyperparameters [11].

Optimizing Equation (3.24) with gradient descent requires the inversion of the kernel matrix at each iteration. It quickly becomes a costly operation, limiting the size of the dataset that can be used realistically. A possible solution to this problem is to approximate the global GP by several local models, as proposed in work such as [28, 29]. These smaller GP models can then be evaluated in parallel and collaborate in the search for a global optimum.

3.7 Sampling from a Gaussian Process

As the GP is a joint Gaussian distribution, random samples can be drawn from it as with any other multivariate Gaussian.

$$\boldsymbol{f}_* \sim \mathcal{N}(\bar{\boldsymbol{f}}_*, \mathbb{V}[\boldsymbol{f}_*]) \tag{3.27}$$

[3]The marginal likelihood is generally not a convex function of $\boldsymbol{\theta}$ as the kernel functions are usually non-linear functions of $\boldsymbol{\theta}$, and multiple local optima may exist. The practical effects include that the observed data may be explained well by different combinations of parameters, and each combination serves as a distinct interpretation of the data. During optimization, care should be taken to avoid a bad local optimum [11].

More specifically, a vector of i.i.d standard normal samples, Z, can be turned into samples from a joint Gaussian distribution.

Once again, the Cholesky decomposition is used to decompose the GP covariance, i.e. $\mathbb{V}[\boldsymbol{f}_*] = LL^\intercal$. Using L and $\bar{\boldsymbol{f}}_*$, Equation (3.27) can be expressed as

$$\mathcal{N}(\bar{\boldsymbol{f}}_*, \mathbb{V}[\boldsymbol{f}_*]) = \bar{\boldsymbol{f}}_* + L\mathcal{N}(0, I) \tag{3.28}$$

which yields a way to convert indendent standard normal samples into samples from a GP [11]:

$$\boldsymbol{f}_* = \bar{\boldsymbol{f}}_* + LZ, \quad Z \sim \mathcal{N}(0, I) \tag{3.29}$$

3.8 Standardization

The method of *standardization* is the process of converting the data into a standard unit of measurement, which in practice typically involves transforming the data to have zero mean and unit variance [30]. This is useful when comparing measures of different scales.

The proposed GP implementation works well with and without standardized input data, but standardization has one crucial benefit. While the current implementation allows separate kernel hyperparameters for each input dimension in $\boldsymbol{x}$, the kernel is assumed to be identical for each output of $\vec{f}$. The kernel therefore represents the variability of **both** output dimensions at once. However, with standardized training outputs Y, the kernel is relative to the standard deviation of the training data. This way, the GP can express the different levels of uncertainty for each output dimension while still sharing the same kernel. An even more flexible approach would be to allow separate kernels for each dimension, though at the cost of more complex hyperparameter tuning.

3.9 Approximate methods

The standard derivation of the GP requires inverting the kernel matrix, either through the Cholesky Decomposition or directly. Unfortunately, the computational complexity is $\frac{n^3}{6}$ operations for the Cholesky Decomposition and similarly $\frac{n^2}{2}$ for solving triangular systems [11]. This may be acceptable for sparse datasets, but it makes GPs infeasible for Big Data applications.

For simple functions, it often works well to only use a representable subset of the data. However, throwing away the majority of available data is not an elegant solution to the problem.

3.9.1 Reduced-rank approximation

Several approximations are discussed in [11] and typically use a *reduced-rank approximation* of the kernel matrix $K = QQ^\intercal$ with $Q \in \mathcal{R}^{n \times q}$, and use the *matrix*

inversion lemma [11, p. 201], reducing the inversion of the $n \times n$ matrix to the inversion of a $q \times q$ matrix instead. Unfortunately, the optimal reduced-rank approximations depend on the eigenvalue decomposition, which itself is a $\mathcal{O}(n^3)$ operation. However, if a cheap approximation to the eigenvalue decomposition can be found, it can potentially be used to approximate the GP. One such low-rank approximation is the Nyström method, where a subset of the dataset is used to approximate the eigenvalues.

3.9.2 Sparse Variational Gaussian Process

Another solution is to consider the elements in the covariance matrix as the parameters to a surrogate distribution and then use *Variational Inference* to approximate the true posterior distribution [31]. The problem of inverting large matrices is then turned into an optimization problem. Variational Inference is covered in the specialization project [25].

The *Sparse Variational Gaussian Process (SVGP)* is a sparse approximation of GPs, which builds on variational inference. The idea is to summarize the data $(X, \boldsymbol{y})$ using a set of m *inducing variables* for the latent function values $\boldsymbol{f}_m$ at the inducing points X_m [32]. X_m can either be a subset of the available training inputs X or auxiliary *pseudo-points*. The goal of SVGP is then to infer the inducing points X_m as well as the hyperparameters $\boldsymbol{\theta}$ from data. Libraries, such as *GPFlow* [33], allow the SVGP to be combined with powerfull stochastic optimization techniques to learn sparse approximations for large datasets.

3.10 The Kalman Filter

The Kalman filter is an algorithm for exact Bayesian filtering for linear Gaussian state-space models [17]. The reader is assumed to already be familiar with the Kalman filter, but a short introduction is nevertheless included for completeness.

The filtering is performed in two steps. The *prediction step* computes the marginal distribution of the current state $\boldsymbol{x}_t$ given the known measurements $\boldsymbol{z}_{1:t-1}$ up to the previous timestep.

$$\begin{aligned} p(\boldsymbol{x}_t|\boldsymbol{z}_{1:t-1}) &= \int_{\boldsymbol{x}_{t-1}} p(\boldsymbol{x}_t, \boldsymbol{x}_{t-1}|\boldsymbol{z}_{1:t-1}) d\boldsymbol{x}_{t-1} \\ &= \int_{\boldsymbol{x}_{t-1}} p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) p(\boldsymbol{x}_{t-1}|\boldsymbol{z}_{1:t-1}) d\boldsymbol{x}_{t-1} \end{aligned} \tag{3.30}$$

The *update step* directly follows from Bayes Rule

$$p(\boldsymbol{x}_t|\boldsymbol{z}_{1:t}) = \frac{p(\boldsymbol{z}_t|\boldsymbol{x}_t) p(\boldsymbol{x}_t|\boldsymbol{z}_{1:t-1})}{p(\boldsymbol{x}_t|\boldsymbol{z}_{1:t-1})} \tag{3.31}$$

These two equations yield what is known as the Bayes filter, and by assuming all distributions to be linear Gaussian, there exists a closed-form solution given by Algorithm 2 [17, 34].

Algorithm 2 Kalman Filter

procedure KALMAN FILTER($\boldsymbol{x}_{t-1}, \boldsymbol{P}_{t-1}, \boldsymbol{z}_t$)
  $\hat{\boldsymbol{x}}_t = \boldsymbol{G}\boldsymbol{x}_{t-1}$ ▷ Predicted Mean
  $\hat{\boldsymbol{P}}_t = \boldsymbol{G}\boldsymbol{P}_{t-1}\boldsymbol{G}^\intercal + \boldsymbol{Q}$ ▷ Predicted Covariance
  $\hat{\boldsymbol{z}} = \boldsymbol{H}\hat{\boldsymbol{x}}_t$ ▷ Predicted measurement
  $\boldsymbol{\nu}_t = \boldsymbol{z}_t - \hat{\boldsymbol{z}}_t$ ▷ Innovation
  $\boldsymbol{S}_t = \boldsymbol{H}\hat{\boldsymbol{P}}\boldsymbol{H}^\intercal + \boldsymbol{R}$ ▷ Innovation Covariance
  $\boldsymbol{W}_t = \hat{\boldsymbol{P}}_t\boldsymbol{H}^\intercal\boldsymbol{S}_t^{-1}$ ▷ Kalman Gain
  $\boldsymbol{x}_t = \hat{\boldsymbol{x}}_t + \boldsymbol{W}_t\boldsymbol{\nu}_t$ ▷ Mean of Posterior
  $\boldsymbol{P}_t = (\boldsymbol{I} - \boldsymbol{W}_t)\hat{\boldsymbol{P}}_t$ ▷ Variance of Posterior
  return $\boldsymbol{x}_t, \boldsymbol{P}_t$
end procedure

Further details and derivations can be found in [17, 34].

Chapter 4

Historical AIS data

The *Automatic Identification System (AIS)* is a vessel-to-vessel communication system, which allows a vessel to share vital information about its current state with other ships, base stations, and satellites electronically. International voyaging ships with gross tonnage larger than 300 and all passenger ships are required to install an AIS transceiver according to the *Safety Of Life At Sea* (SOLAS) convention [35]

The typical AIS receiver has a range of about 10-20 nautical miles. However, AIS transceivers have been installed on satellites in later years, resulting in global coverage. The transceivers are divided into two types of classes [35]:

Class A typically transmits at a higher rate, ranging between 30 times per minute for high-velocity vessels to every third minute for vessels at rest.

Class B are typically smaller, cheaper, and simpler than their Class A counterparts. The position message for Class B is sent every 3 minutes when the vessel's speed is less than 2 knots and every 30 seconds for faster speeds.

4.1 Message Contents

The typical AIS message contains unique identification (MMSI / IMO), position (longitude and latitude GPS coordinates), course (COG) and speed (SOG). The relevant message content for this thesis is summarized in Table 4.1.

Several other fields may additionally be available, depending on the transceiver and which information the crew has entered.

4.2 The Dataset

The total dataset contains 2995644 AIS samples collected between 1. Jan and 31. Des 2015 from the Trondheim Fjord in Norway. There are 1555 unique MMSI values, though most of the messages come from a small subset of the vessels. More than 80% of the dataset originates from the top 250 vessels, as seen in Figure 4.1.

Parameter	***Explanation***
IMO	7 digit vessel identification number that remains unchanged when transferring a vessel's registration to a new country
MMSI	*Maritime Mobile Service Identity (MMSI)*, a 9 digit vessel identification number
Long	Degrees longitude in range $[-180°W, 180°E]$
Lat	Degrees latitude in range $[90°S, 90°N]$
COG	*Course Over Ground (COG)* is the clockwise rotation of the vessel's velocity vector relative to true north
SOG	*Speed Over Ground (SOG)* is the absolute value of the vessel's velocity vector in knots.
Heading	Direction of vessel's nose or bow relative to true north. Independent of the actual movement, so not necessarily identical to COG. NB: Not always available.
Timestamp	Number of days elapsed since 1. jan 1900, 00:00

Table 4.1: Common content of AIS messages [4].

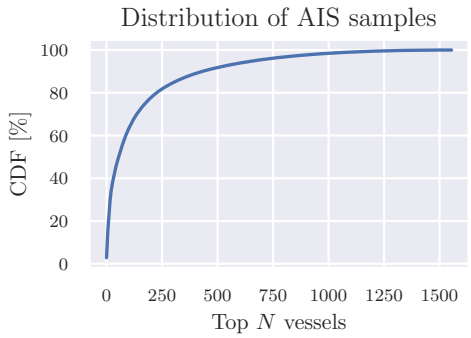
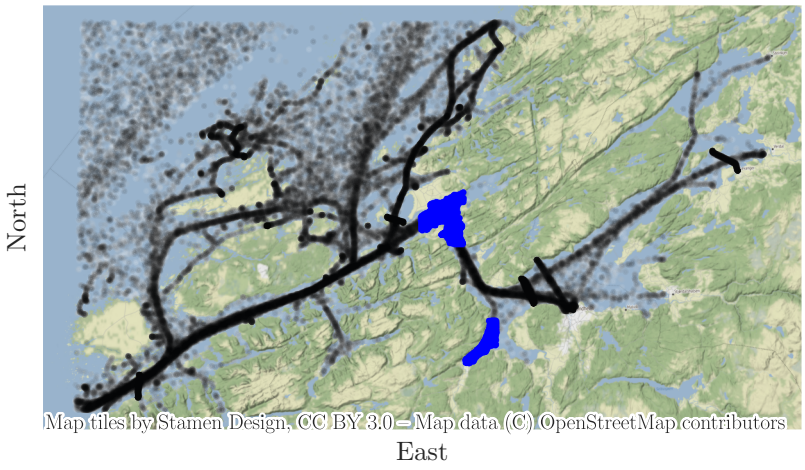


Figure 4.1: Cumulative distribution of AIS messages for the N most frequent vessel's.

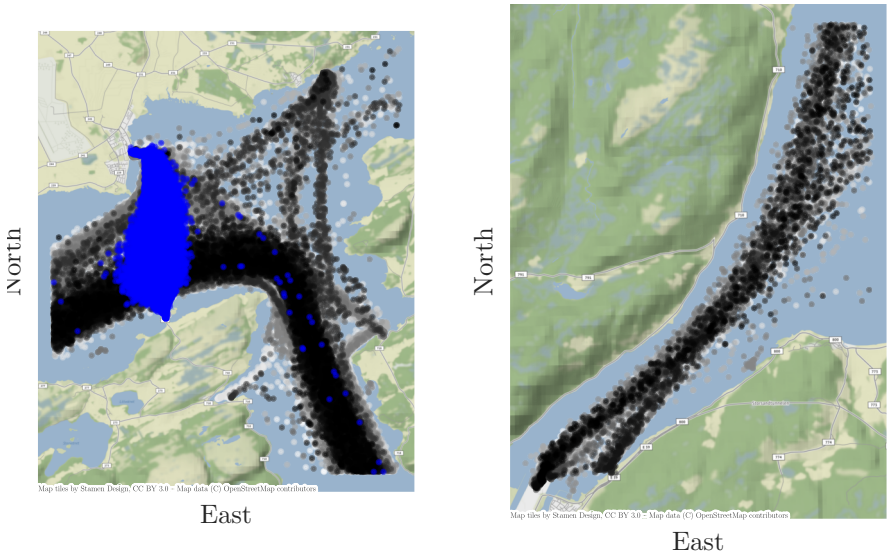
However, the dataset is currently too large to conveniently work with on a single computer. Two local regions are instead used in this thesis, and they are shown in Figure 4.2. The first subset is from a curved section of the Trondheim fjord with a lot of traffic. The subset also includes a ferry crossing, which crosses the traffic lane. The other subset is from deeper into the fjord, from a relatively straight section. The dataset therefore contains both straight-line and curved trajectories, as well as crossing trajectories.

4.3 Scenario

The AIS dataset contains samples from a wide variety of different trajectories, as each trajectory is the result of a vessel's intention, be it reaching a specific



(a) Full dataset. The blue areas mark the subset that will be used for testing purposes.



(b) Zoomed-in version of the subsets used in this thesis. The blue subset in the left-most map shows the AIS data from three local ferries, which make out more than half of the dataset.

Figure 4.2: The available AIS data used in this thesis.

destination, commercial fishing, or simply for leisure. As a result, there is a wide range of possible overlapping trajectories that a vessel might follow in any given area. Therefore, this thesis will narrow the problem of long-term prediction down to prediction in specific *scenarios*. A scenario is in this thesis characterized by the vessel's initial position, speed, and course, and it is assumed that vessels in a given scenario tend to follow similar trajectories. Though not used in this thesis, it would also be natural to incorporate contextual information such as vessel type, as there is likely quite a lot of variability between the behavior of different types of vessels.

When predicting the trajectory for a given target vessel, only the subset of AIS messages from similar scenarios will be used. This simplifying assumption is a critical part of all methods proposed in this thesis. It allows the AIS data to be reduced significantly for any given prediction and is what makes the methods computationally feasible. The following requirements must be satisfied for a trajectory to qualify as being in a similar scenario as the target vessel:

1. The trajectories' initial position must be close to the target vessel's position $\boldsymbol{x}_0$. A fixed threshold at $||\Delta \boldsymbol{x}_0|| \leq r$ is used, where $\Delta \boldsymbol{x}_0$ is the difference between the target vessel's current position and the initial conditions of a potential training trajectory and r is a fixed radius.
2. The trajectories' initial COG must be close to the target vessel's heading $\mathcal{X}$. A fixed threshold at $\mathcal{X} \pm \Delta\mathcal{X}$ is used.
3. The trajectories' initial SOG must be close to the queried velocity v. A fixed threshold at $v \pm \Delta v$ is used.

The notion of only using a small subset of close-by samples from the AIS data is somewhat similar to the SPNS method proposed by [24], though a key distinction is that this thesis selects entire trajectories based on the initial conditions, rather than individual points.

4.3.1 Clustering

In practical applications, it may be beneficial to use a fixed number of scenarios. For example, clustering-based methods may be used to cluster similar trajectories into a fixed number of scenarios. This approach is not implemented in this thesis but is considered a natural next step.

4.4 Preprocessing

The position (longitude and latitude) is converted from the standard World Geodetic System (WGS84) coordinates into the EUREF89 UTM32 (EPSG-25832) coordinate system [36]. As a result, the coordinates are converted from spherical coordinates to a 2D euclidian coordinate system, where the first and second axis corresponds to easting and northing, respectively. This coordinate system is one of two official coordinate systems in Norway and yields significantly better projections in this area than other alternatives, such as the Web-Mercator projector

frequently used by online maps. Zone 32 is selected as it encompasses the target region.

4.4.1 From samples to trajectories

Samples with identical MMSI and less than 15 minutes between subsequent samples are considered part of the same trajectory. The 15 minute requirement is added to ensure that samples before and after docking are considered separate trajectories. The length of the trajectories must be between 15 and 30 minutes, and the number of samples in each trajectory must be greater than 4.

Chapter 5

Model trajectories directly using a Gaussian Process

A tempting solution is to directly apply the GP framework introduced in Chapter 3 and model the vessel's trajectory as a function of time $\vec{f}(\tau)$. However, as there are no observations of the target vessel's future position, there is no actual data to condition the GP on. Instead, this chapter will attempt to use nearby trajectories and assume these historical trajectories to resemble the future trajectory of the target vessel.

5.1 Method

While a pure function of time is tempting, it is unable to differentiate between different historical trajectories. To utilize as much of the available information in a single AIS message as possible, and in order to compensate for any initial differences in position, velocity or course of nearby trajectories, a bit more complex formulation $\vec{f}(\boldsymbol{x}_0, \psi_0, v_0, \tau) : \mathcal{R}^5 \rightarrow \mathcal{R}^2$ is proposed. The key variables are explained in Table 5.1.

Variable	***Description***
$\boldsymbol{x}_0 \in \mathcal{R}^2$	Trajectory's initial position
$\mathcal{X}_0 \in [0, 360)$	Trajectory's initial COG
$v_0 \in \mathcal{R}$	Trajectory's initial SOG
$\tau \in [0, \infty)$	Prediction timestamp
$\boldsymbol{x}_\tau \in \mathcal{R}^2$	Predicted position

Table 5.1: Key variables

The method is formulated in mathematical terms in Equation (5.1). This will from now on be referred to as the *Direct GP* approach, as it directly utilizes a GP to predict the trajectory.

$$\boldsymbol{f}(\boldsymbol{\eta}) = \boldsymbol{x}_\tau, \quad \boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{x}_0 & \psi_0 & v_0 & \tau \end{bmatrix} \tag{5.1a}$$

$$\boldsymbol{f}(\boldsymbol{\eta}) \sim \mathrm{GP}(\boldsymbol{m}(\boldsymbol{\eta}), K(\boldsymbol{\eta}, \boldsymbol{\eta})) \tag{5.1b}$$

The function $\vec{f}$ can be conditioned on similar trajectories using Algorithm 1 from Chapter 3, and then be used to answer queries about the likely trajectories that the vessel might follow. An example output is available in Figure 5.1.

5.1.1 Key Assumption

This formulation builds on the key assumption that the target vessel is following the same underlying trajectory as the nearby historical trajectories. In other words, the function $\vec{f}$ does not represent the target vessel's future trajectory; it merely interpolates the past trajectories with similar initial conditions.

5.1.2 Choice of Kernel

Many different kernels may work well with this formulation. This section will cover a few alternatives that works well in practice, but there are likely many other alternatives that may work just as well, if not better.

The RBF kernel was found to work well with the formulation used in this chapter. Along the traffic lanes, vessels tend to move in a smooth trajectory, which makes the RBF kernel a good choice. As the function arguments are of different scales, it is a good idea to use different lengthscales for each input dimension, i.e.

$$k(\boldsymbol{\eta}, \boldsymbol{\eta}') = \sigma^2 \exp\left[(\boldsymbol{\eta} - \boldsymbol{\eta}')^\intercal W^{-1} (\boldsymbol{\eta} - \boldsymbol{\eta})\right] \tag{5.2}$$

where W is a diagonal matrix with a separate lengthscale for each dimension and σ^2 is a scale parameter. This kernel yields smooth functions for $\vec{f}$ and is well suited to model the general trend of the trajectory.

In some scenarios, a single RBF kernel results in an unreasonably smooth function. Adding a second RBF kernel as in Equation (5.3) works well in these scenarios, as it gives the model some additional flexibility in local regions. However, it makes hyperparameter selection significantly more complicated as the optimal solution is no longer unique.

$$k(\boldsymbol{\eta}_i, \boldsymbol{\eta}_j) = \sigma_0 k_0(\boldsymbol{\eta}_i, \boldsymbol{\eta}_j) + \sigma_1 k_1(\boldsymbol{\eta}_i, \boldsymbol{\eta}_j) \tag{5.3}$$

Each term in this kernel offers a unique interpretation:

Long term trend k_0 is an RBF kernel intended to cover the long-term behavior of the trajectories, i.e., a smooth component describing the overall trend of the trajectories. The length scales of this kernel are expected to be relatively large.

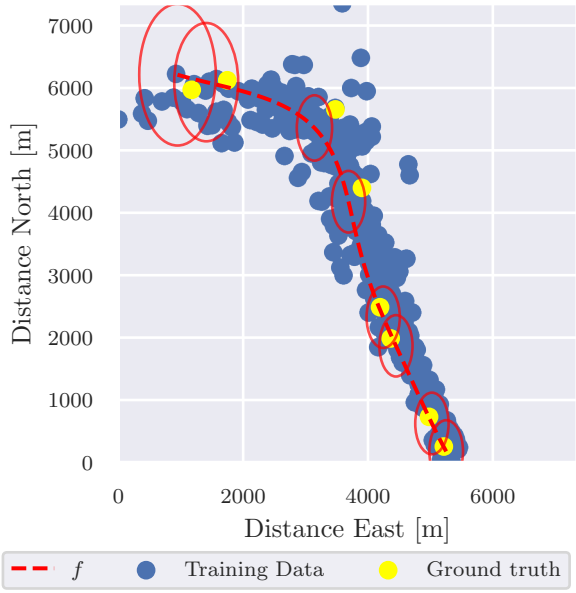
Dependent noise k_1 is an RBF kernel intended to model the local variations between different trajectories which is not well explained by k_0. The length scales are therefore expected to be short.

5.2 Implementation Details

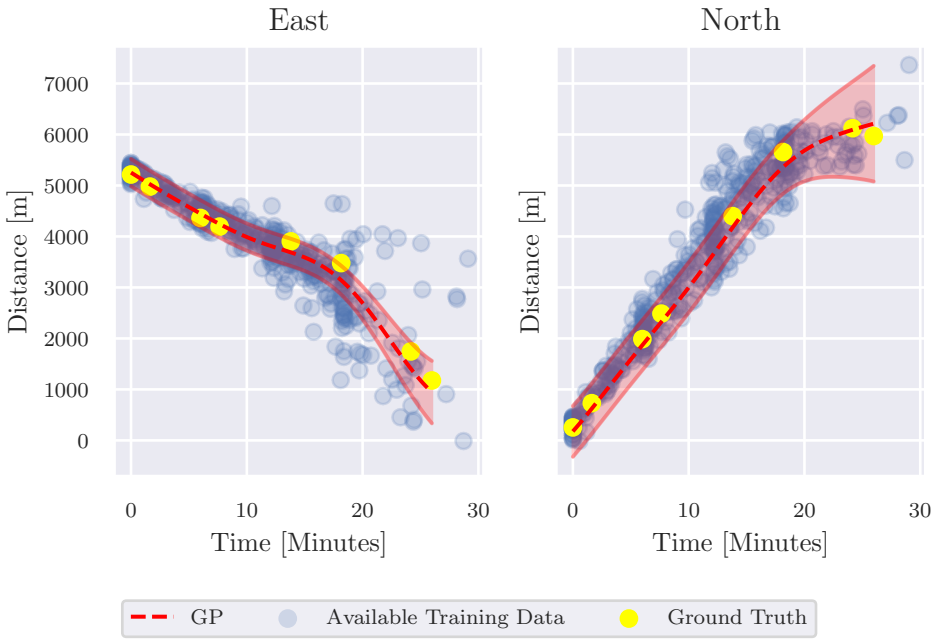
This formulation builds directly on the GP framework introduced in Chapter 3 and the trajectory can be conditioned on available training data using Algorithm 1.

5.2.1 Approximate Gaussian Process

Another promising approach is to utilize an approximation technique to avoid the scalability issue of GPs. While this thesis focuses on specific scenarios, as discussed in Chapter 4, the formulation in this chapter lends itself nicely to using more data. Though not implemented in this thesis, the SVGP introduced in Chapter 3 is one such method that attempts to fit a GP while simultaneously finding a good subset of the data to summarize the full dataset.



(a) The predicted trajectory using the direct GP formulation. The ellipses are the 95% CI for the prediction at the ground truth timestamps.



(b) The East and North components plotted against time with a 95% CI.

Figure 5.1: Example output from the Direct GP approach.

Chapter 6

GP-EKF: Non-parametric dynamic system using AIS tracking data

One of the significant issues with the direct approach is the unimodal assumption of using a GP. It works well as long as vessels agree on a specific trajectory, but fails as soon as there are multiple branching trajectories. In search of a better solution, a non-parametric dynamical model is proposed in this chapter and is inspired by [12, 13, 16, 18].

The vessel trajectory $\mathcal{T}$ can be expressed using the dynamical system

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \vec{f}(\boldsymbol{x}_t, \tau_t) \tag{6.1a}$$

$$\mathcal{T}_t = \boldsymbol{x}_t + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2) \tag{6.1b}$$

The function $\vec{f}(\cdot) : \mathcal{R}^3 \to \mathcal{R}^2$ denotes the vector field describing the expected velocity. In the case of long-term prediction, the dynamics $\vec{f}(\cdot)$ are unknown and unlikely to be stationary. Instead of using the usual parametric approaches to ODE models, the goal of this chapter is to use a GP to create a non-parametric representation of the dynamics $\vec{f}(\cdot)$ by learning from historical trajectories of other vessels. This way, arbitrary complex dynamics can be learned without being limited by a fixed parametrization. The GP considered in this chapter is a vector-valued GP with zero mean and identical kernel for each output dimension, as expressed in Equation (6.2). The output dimensions are assumed to be independent.

$$\vec{f}(\boldsymbol{x}, t) = \vec{f}(\boldsymbol{\eta}) = \begin{bmatrix} f_E(\boldsymbol{\eta}) \\ f_N(\boldsymbol{\eta}) \end{bmatrix} \sim \mathrm{GP}\big(0,\ k(\boldsymbol{\eta}, \boldsymbol{\eta}')\big) \tag{6.2}$$

The pipeline for making predictions using available AIS data will now be introduced in greater detail, but can be summarized as:

1. Calculate trajectory gradients $\boldsymbol{y}$ for inputs $\boldsymbol{\eta}$ from available AIS data.
2. Fit a GP to the time-varying vector-field $\vec{f} : \mathcal{R}^3 \to \mathcal{R}^2$
3. Simulate the vessel as it is moving through the vector field $\vec{f}$, using either EKF-based prediction or Sequential Monte Carlo.

6.1 Notation and variables

The key variables for this chapter are summarized in Table 6.1. All other variables will be introduced as needed.

Variable	*Description*
$\boldsymbol{x}_t \in \mathcal{R}^2$	Vessel position at step t
$\tau_t \in \mathcal{R}$	Timestamp at step t, number of seconds since start of trajectory
$\mathcal{X}_t \in [0, 360)$	Vessel's course over ground in degrees at step t
$v_t \in \mathcal{R}$	Vessel's speed over ground in knots at step t
$\boldsymbol{P}_t \in \mathcal{R}^{2x2}$	State Covariance at step t

Table 6.1: Key variables

Note that throughout this chapter, some of the notation used in Section 3.3 will be relaxed in order to reduce the notational complexity. More specifically, the GPs in this chapter are always assumed to be conditioned on available data, i.e. $p\big(\vec{f}(\boldsymbol{x})\big) = p\big(\vec{f}(\boldsymbol{x}) \mid \boldsymbol{y}\big)$.

6.2 Simulating Trajectories

The vector-field $\vec{f}$ can be expressed using the GP framework from Chapter 3 to get the prediction model

$$p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t) = \mathcal{N}\big(\boldsymbol{x}_{t+1} \mid (\boldsymbol{x}_t + \mathbb{E}[\vec{f}(\boldsymbol{x}_t)]), \mathbb{V}[\vec{f}(\boldsymbol{x}_t, \tau_t)]\big) \tag{6.3}$$

Given a current estimate $p(\boldsymbol{x}_t)$, the marginal next state distribution for the predicted state then becomes

$$p(\boldsymbol{x}_{t+1}) = \int_{\boldsymbol{x}_t} p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t) p(\boldsymbol{x}_t) d\boldsymbol{x}_t \tag{6.4}$$

However, this integral is intractable, as both the mean and variance depend on the current state $\boldsymbol{x}_t$ with non-trivial relationships [12, 18]. A solution to this problem is to use sampling-based methods, and one such approach is discussed later in Section 6.4. However, the main method proposed in this chapter will approximate this integral by making a few simplifying assumptions. The integral can be approximated by assuming that the posterior distribution is Gaussian, and the mean and variance can be calculated using the law of iterated expectations and the law of total variance [18]. Assuming the current estimate to be distributed according to the Gaussian distribution $p(\boldsymbol{x}_t) = \mathcal{N}(\bar{\boldsymbol{x}}_t, \boldsymbol{P}_t)$, the mean and variance of $\boldsymbol{x}_{t+1}$ is given by

$$\begin{aligned}\mathbb{E}[\boldsymbol{x}_{t+1}] &= \mathbb{E}\big[\mathbb{E}[\boldsymbol{x}_{t+1}|\boldsymbol{x}_t]\big] \\ &= \mathbb{E}\big[\boldsymbol{x}_t + \mathbb{E}[\vec{f}(\boldsymbol{x}_t,\tau_t)|\boldsymbol{x}_t]\big] \\ &= \bar{\boldsymbol{x}}_t + \mathbb{E}\big[\mathbb{E}[\vec{f}(\boldsymbol{x}_t,\tau_t)|\boldsymbol{x}_t]\big]\end{aligned} \tag{6.5a}$$

$$\begin{aligned}\mathbb{V}[\boldsymbol{x}_{t+1}] &= \mathbb{E}\big[\mathbb{V}[\boldsymbol{x}_{t+1}|\boldsymbol{x}_t]\big] + \mathbb{V}\big[\mathbb{E}[\boldsymbol{x}_{t+1}|\boldsymbol{x}_t]\big] \\ &= \mathbb{E}\big[\mathbb{V}[\vec{f}(\boldsymbol{x}_t,\tau_t)|\boldsymbol{x}_t]\big] + \mathbb{V}\big[\boldsymbol{x}_t + \mathbb{E}[\vec{f}(\boldsymbol{x}_t,\tau_t)|\boldsymbol{x}_t]\big]\end{aligned} \tag{6.5b}$$

The mean and variance of the GP are the same conditional mean and variance as in Equation (3.16) from Chapter 3, which both are non-linear functions of $\boldsymbol{x}_t$. A first-order Taylor approximation around the current best estimate $\bar{\boldsymbol{x}}_t$ is then applied to linearize the GP mean and variance, with respect to $\boldsymbol{x}_t$.

$$\mathbb{E}[\vec{f}(\boldsymbol{x}_t)|\boldsymbol{x}_t)] \approx \mathbb{E}[\vec{f}(\bar{\boldsymbol{x}}_t,\tau_t)] + \left.\frac{\partial \mathbb{E}[\vec{f}(\boldsymbol{x}_*,\tau_t)]}{\partial \boldsymbol{x}_*}\right|_{\boldsymbol{x}_*=\bar{\boldsymbol{x}}_t} (\boldsymbol{x}_t - \bar{\boldsymbol{x}}_t) \tag{6.6a}$$

$$\mathbb{V}[f(\boldsymbol{x}_t,\tau_t)|\boldsymbol{x}_t] \approx \mathbb{V}[\vec{f}(\bar{\boldsymbol{x}}_t,\tau_t)] + \left.\frac{\partial \mathbb{V}[\vec{f}(\boldsymbol{x}_*,\tau_t)]}{\partial \boldsymbol{x}_*}\right|_{\boldsymbol{x}_*=\bar{\boldsymbol{x}}_t} (\boldsymbol{x}_t - \bar{\boldsymbol{x}}_t) \tag{6.6b}$$

Defining temporary variables for the Jacobians

$$\boldsymbol{U}_t \triangleq \left.\frac{\partial \mathbb{E}[\vec{f}(\boldsymbol{x}_*,\tau_t)]}{\partial \boldsymbol{x}_*}\right|_{\boldsymbol{x}_*=\bar{\boldsymbol{x}}_t} \qquad \boldsymbol{V}_t = \left.\frac{\partial \mathbb{V}[\vec{f}(\boldsymbol{x}_*,\tau_t)]}{\partial \boldsymbol{x}_*}\right|_{\boldsymbol{x}_*=\bar{\boldsymbol{x}}_t} \tag{6.7}$$

and inserting Equation (6.6) into Equation (6.5) then yields

$$\begin{aligned}\mathbb{E}[\boldsymbol{x}_{t+1}] &\approx \bar{\boldsymbol{x}}_t + \mathbb{E}\big[\mathbb{E}[\vec{f}(\bar{\boldsymbol{x}}_t,\tau_t)] + \boldsymbol{U}_t(\boldsymbol{x}_t - \bar{\boldsymbol{x}}_t)\big] \\ &= \bar{\boldsymbol{x}}_t + \mathbb{E}[\vec{f}(\bar{\boldsymbol{x}}_t,\tau_t)] + \boldsymbol{U}_t(\mathbb{E}[\boldsymbol{x}_t] - \bar{\boldsymbol{x}}_t) \\ &= \bar{\boldsymbol{x}}_t + \mathbb{E}[\vec{f}(\bar{\boldsymbol{x}}_t,\tau_t)]\end{aligned} \tag{6.8a}$$

$$\begin{aligned}\mathbb{V}[\boldsymbol{x}_{t+1}] &\approx \mathbb{E}\big[\mathbb{V}[\vec{f}(\bar{\boldsymbol{x}}_t,\tau_t)] + \boldsymbol{V}_t(\boldsymbol{x}_t - \bar{\boldsymbol{x}}_t)\big] \\ &\quad + \mathbb{V}\big[\boldsymbol{x}_t + \mathbb{E}[\vec{f}(\bar{\boldsymbol{x}}_t,\tau_t)] + \boldsymbol{U}_t(\boldsymbol{x}_t - \bar{\boldsymbol{x}}_t)\big] \\ &= \mathbb{V}[\vec{f}(\bar{\boldsymbol{x}}_t,\tau_t)] + \boldsymbol{V}_t(\mathbb{E}[\boldsymbol{x}_t] - \bar{\boldsymbol{x}}_t) + \mathbb{V}[\boldsymbol{x}_t + \boldsymbol{U}_t\boldsymbol{x}_t] \\ &= \mathbb{V}[\vec{f}(\bar{\boldsymbol{x}}_t,\tau_t)] + (\boldsymbol{I} + \boldsymbol{U}_t)\mathbb{V}[\boldsymbol{x}_t](\boldsymbol{I} + \boldsymbol{U}_t)^\intercal \\ &= \mathbb{V}[\vec{f}(\bar{\boldsymbol{x}}_t,\tau_t)] + \boldsymbol{G}_t\boldsymbol{P}_t\boldsymbol{G}_t^\intercal\end{aligned} \tag{6.8b}$$

which those familiar with sensor fusion may recognize as the Extended Kalman Filter (EKF) prediction [13]. Notice that only the Jacobian of the expected value $\left.\frac{\partial \mathbb{E}[\vec{f}(\boldsymbol{x}_*,\tau_t)]}{\partial \boldsymbol{x}_*}\right|_{\boldsymbol{x}_*=\bar{\boldsymbol{x}}_t}$ is actually neccessary in the end. This Jacobian is easy to compute and only relies on the covariance between the test point $\boldsymbol{v}_*$ and the training inputs as expressed in

$$\begin{aligned}\frac{\partial \vec{f}(\boldsymbol{x}_*, \tau_*)}{\partial \boldsymbol{x}_*} = \frac{\partial \vec{f}(\boldsymbol{\eta})}{\partial \boldsymbol{x}_*} &= \frac{\partial}{\partial \boldsymbol{x}_*}\left(\boldsymbol{k}_*^\intercal \boldsymbol{\alpha}\right) \\ &= \frac{\partial \boldsymbol{k}_*^\intercal}{\partial \boldsymbol{x}_*}\boldsymbol{\alpha} \\ &= \begin{bmatrix} \frac{\partial k(\boldsymbol{\eta}_*, \boldsymbol{\eta}_1)}{\partial \boldsymbol{x}_*[1]} & \frac{\partial k(\boldsymbol{\eta}_*, \boldsymbol{\eta}_1)}{\partial \boldsymbol{x}_*[2]} \\ \frac{\partial k(\boldsymbol{\eta}_*, \boldsymbol{\eta}_2)}{\partial \boldsymbol{x}_*[1]} & \frac{\partial k(\boldsymbol{\eta}_*, \boldsymbol{\eta}_2)}{\partial \boldsymbol{x}_*[2]} \\ \vdots & \vdots \\ \frac{\partial k(\boldsymbol{\eta}_*, \boldsymbol{\eta}_N)}{\partial \boldsymbol{x}_*[1]} & \frac{\partial k(\boldsymbol{\eta}_*, \boldsymbol{\eta}_N)}{\partial \boldsymbol{x}_*[2]} \end{bmatrix}^\intercal \boldsymbol{\alpha}\end{aligned} \tag{6.9}$$

where the notation $\boldsymbol{x}_*[i]$ refers to the i-th dimension of $\boldsymbol{x}_*$ and $\boldsymbol{\eta}_k$ is the k-th training input.

Note that this approximation to Equation (6.4) implicitly makes a few assumptions about the motion model $\vec{f}$, namely that the dynamics are continuous and highly smooth. It is believed to be a reasonable assumption for vessels in transit, but makes this approximation unsuitable for complicated situations. Including higher-order derivatives in the Taylor approximation may yield even better results for more complex dynamics, and as proposed by [18] it is natural to extend the method by using a second-order approximation for the variance.

6.2.1 GP-EKF

The joint distribution of all states $\boldsymbol{x}_{0:t}$ up to timestep t is given by

$$p(\boldsymbol{x}_{0:t}) = p(\boldsymbol{x}_0)\prod_{i=0}^{t-1} p(\boldsymbol{x}_{i+1}|\boldsymbol{x}_{0:i}) \tag{6.10}$$

where the initial distribution $p(\boldsymbol{x}_0) = \mathcal{N}(\bar{\boldsymbol{x}}_0, \boldsymbol{P}_0)$ is assumed to be a known Gaussian.

Applying the Markov assumption allows this joint distribution to be expressed in terms of the predictive distribution from Equation (6.3), i.e. $p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_{0:t}) = p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t)$.

$$p(\boldsymbol{x}_{0:t}) = p(\boldsymbol{x}_0)\prod_{i=0}^{t-1} p(\boldsymbol{x}_{i+1}|\boldsymbol{x}_i) \tag{6.11}$$

At each step, the true posterior distribution from Equation (6.4) is approximated using a Gaussian distribution, as derived in the previous section. The full trajectory can then be simulated by recursively applying Equation (6.8).

This solution is heavily inspired by the *Extended Kalman Filter (EKF)*, which is why the rest of this thesis will refer to this method as the GP-EKF. The combination of GPs and EKF was proposed by [13] and is summarized here. The reader is

assumed to already be familiar with the Kalman filter and, by extension, the EKF. The proposed method will now be restated in terms of the EKF.

During the prediction procedure, the state is updated incrementally by adding $\vec{f}$ to the current state. In other words, the EKF prediction model, $g_t(\boldsymbol{x})$, is given by

$$\hat{\boldsymbol{x}}_t = \boldsymbol{g}_t(\boldsymbol{x}_{t-1}) = \boldsymbol{x}_{t-1} + \vec{f}(\boldsymbol{x}_{t-1}, \tau_{t-1})\Delta\tau \tag{6.12}$$

where the time increment $\Delta\tau$ is factored out of $\vec{f}$ to simplify the implementation[1].

Due to the potentially non-linear dynamics of $\vec{f}$, which implies non-linearity in $\boldsymbol{g}(\cdot)$, it is neccessary to linearize the prediction in order to propagate the previous state uncertianty $\boldsymbol{P}_{t-1}$. The Jacobian of the prediction model, $\boldsymbol{G}_t$, is given by Equation (6.13), where the jacobian of $\vec{f}$ can be computed using Equation (6.9).

$$\boldsymbol{G}_t = \frac{\partial \boldsymbol{g}_t(\boldsymbol{x}_{t-1})}{\partial \boldsymbol{x}_{t-1}} = I + \frac{\partial \vec{f}(\boldsymbol{x}_{t-1}, \tau_t)}{\partial \boldsymbol{x}_{t-1}}\Delta\tau \tag{6.13}$$

The state uncertainty can then be predicted using Equation (6.14), propagating the previous uncertainty $\boldsymbol{P}_{t-1}$ using the linearized prediction model $\boldsymbol{G}_t$ and adding the prediction uncertianty $\mathbb{V}[\vec{f}]$.

$$\boldsymbol{P}_t = \boldsymbol{G}_t \boldsymbol{P}_{t-1} \boldsymbol{G}_t^\intercal + \mathbb{V}[\vec{f}(\boldsymbol{x}_{t-1}, \tau_{t-1})](\Delta\tau)^2 \tag{6.14}$$

The prediction procedure is summarized in Algorithm 3 and can be used iteratively to simulate a complete trajectory, as demonstrated in Figure 6.1.

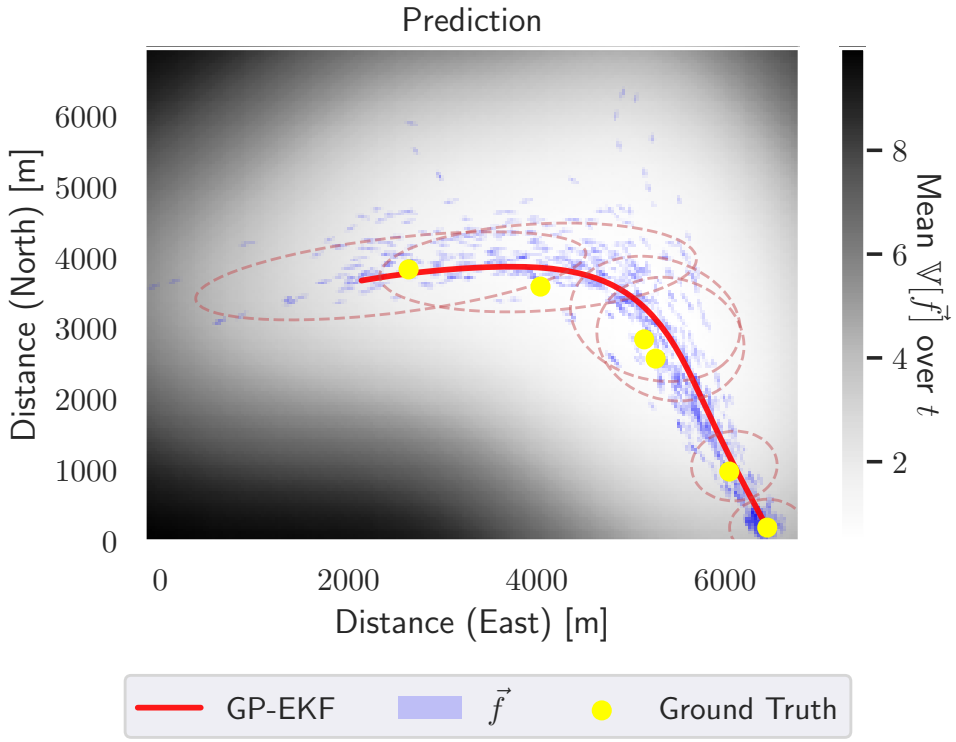
Algorithm 3 GP-EKF Trajectory Prediction

1: **procedure** GP-EKF-PREDICT($\vec{f}$, $\boldsymbol{x}_{t-1}$, $\boldsymbol{P}_{t-1}$, $\Delta\tau$)
2: $\quad \hat{\boldsymbol{x}}_t = \boldsymbol{x}_{t-1} + \mathbb{E}\big[\vec{f}(\boldsymbol{x}_{t-1}, \tau_{t-1})\big]\Delta\tau$
3: $\quad \boldsymbol{G_t} = I + \frac{\partial \vec{f}(\boldsymbol{x}_{t-1}, \tau_{t-1})}{\partial \boldsymbol{x}_{t-1}}\Delta\tau$
4: $\quad \hat{\boldsymbol{P}}_t = \boldsymbol{G_t} \boldsymbol{P}_{t-1} \boldsymbol{G_t}^\intercal + \mathbb{V}[\vec{f}(\boldsymbol{x}_{t-1}, \tau_{t-1})](\Delta\tau)^2$
5: $\quad$ **return** $\hat{\boldsymbol{x}}_t$, $\hat{\boldsymbol{P}}_t$
6: **end procedure**

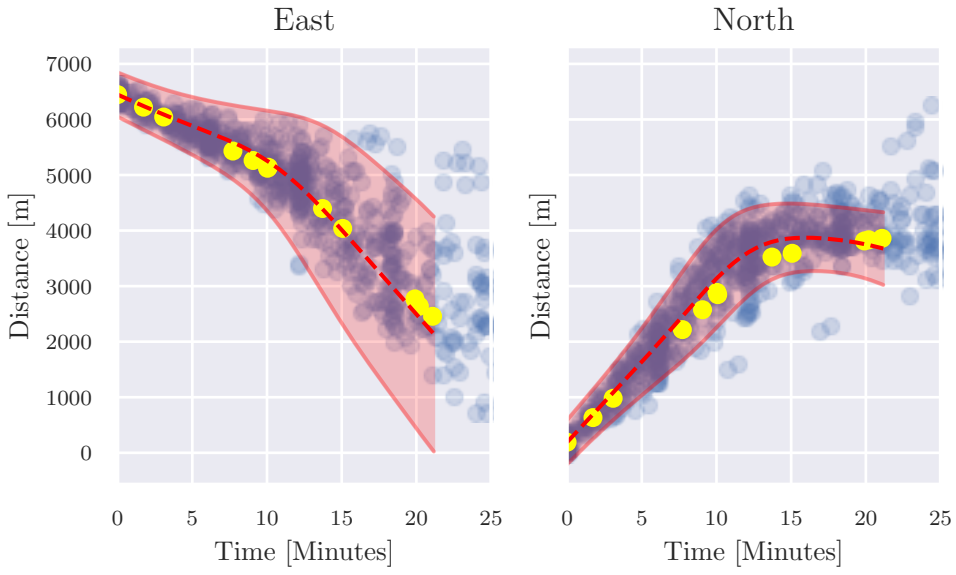
6.3 Incorporating vessel position

While the prediction procedure proposed in Algorithm 3 yields good predictions in many cases, it is inherently an open-loop prediction. Inaccurate predictions will never be corrected, propagating through any remaining iterations, potentially leading to significant errors later on. It would be desirable if the prediction converged towards available position measurements, *slowly and only if the prediction*

[1] Factoring out $\Delta\tau$ allows for the implementation of $\vec{f}$ to only focus on modelling the dynamics in meters per second. This utilizes the fact that $\vec{f}$ follows a Gaussian distribution, for which any linear combination is still Gaussian.



(a) Trajectory plotted against the vector-field $\vec{f}(\boldsymbol{x})$. The ellipses show the 95% credibility interval for the predicted trajectory at the ground-truth timestamps.



(b) 95% credibility interval for the trajectory plotted against time

Figure 6.1: Illustrative example of the GP-EKF in practice

is clearly wrong. In other words, a weak feedback compensates for minor prediction errors. Expressed using Bayes law

$$p(\boldsymbol{x}_{0:t}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{x}_{0:t})p(\boldsymbol{x}_{0:t})}{p(\mathcal{D})} \propto p(\mathcal{D}|\boldsymbol{x}_{0:t})p(\boldsymbol{x}_{0:t}) \tag{6.15}$$

where $\mathcal{D}$ here is all available training data.

Although the dataset $\mathcal{D}$ contains a lot of data from trajectories that are assumed to be similar, the problem is that it does not contain any samples from the target vessel's future trajectory. Thus, the challenge is to relate the target vessel's trajectory to the data-generating process for the dataset $\mathcal{D}$.

This is still very much considered an unsolved problem. This thesis attempts two different approaches, but the solutions are not yet satisfactory.

6.3.1 Synthetic Likelihood

This section is inspired by the work of Jain *et al.* [37].

A proper likelihood $p(\mathcal{D}|\boldsymbol{x}_{0:t})$ is unavailable, or at least considered infeasible to compute. This motivates the use of so-called *likelihood-free* methods, where the most common method is the *Approximate Bayesian Computation* (ABC). Likelihood-free methods are used when the likelihood cannot be computed, while simulation from the model is still possible [38, 39]. In this thesis, the focus will be on another likelihood-free approach, namely, the *Synthetic Likelihood (SL)*, which replaces the likelihood by one or several approximate Gaussian summary statistics, $S_{0:t} = S(\boldsymbol{x}_{0:t}) \sim \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{x}_{0:t}), \boldsymbol{\Sigma}(\boldsymbol{x}_{0:t}))$, of the data. Assuming the summary statistic contains sufficient information about $\boldsymbol{x}_{0:t}$, it should yield a good approximation of the true likelihood, i.e.

$$p(\mathcal{D}|\boldsymbol{x}_{0:t}) \approx p(S_{0:t}|\boldsymbol{x}_{0:t}) \tag{6.16}$$

The likelihood is assumed to factorize into

$$p(S_{0:T}|\boldsymbol{x}_{0:T}) = \prod_{t=0}^{T} p(S_t|\boldsymbol{x}_t) \tag{6.17}$$

and the summary statistic S_i is selected as the mean of the samples in the vicinity of $\boldsymbol{x}_i$. A simple solution is to define the vicinity as a fixed radius around the state $\boldsymbol{x}_i$

$$\mathcal{Y}_t = \mathcal{Y}_t(\boldsymbol{x}_t) = \{\boldsymbol{z}_i \in \mathcal{D} : \ ||\boldsymbol{x}_t - \boldsymbol{z}_i|| \leq r\} \tag{6.18}$$

where the threshold r is a fixed parameter. The summary statistic S_i is then given by

$$S_t = S(\boldsymbol{x}_t) = \frac{1}{N} \sum_{\boldsymbol{z}_i \in \mathcal{Y}_t} \boldsymbol{z}_i \tag{6.19}$$

where N is the number of samples in $\mathcal{Y}_t$. Assuming the measurements in $\mathcal{Y}$ are i.i.d, the summary statistic is then approximately Gaussian by the central limit theorem (see Section 3.1.6). By further assuming a simple measurement model

$$\boldsymbol{z}_t^i = \boldsymbol{x}_t + \epsilon_t^i, \quad \epsilon_t^i \sim \mathcal{N}(0, \boldsymbol{R}) \tag{6.20}$$

the expected value and variance of the summary statistic has a closed-form solution, i.e.

$$\mathbb{E}[S_t] = \frac{1}{N} \sum_{\boldsymbol{z}_t^i \in \mathcal{Y}_t} \mathbb{E}[\boldsymbol{z}_t^i] = \boldsymbol{x}_t \tag{6.21a}$$

$$\mathbb{V}[S_t] = \frac{1}{N^2} \sum_{\boldsymbol{z}_t^i \in \mathcal{Y}_t} \mathbb{V}[\boldsymbol{z}_t^i] = \frac{1}{N}\boldsymbol{R} \tag{6.21b}$$

yields the following parametric distribution for the summary statistic given the predicted state $\boldsymbol{x}_i$

$$p(S_t \mid \boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_t, \frac{1}{N}\boldsymbol{R}) \tag{6.22}$$

The posterior distribution of the trajectory prediction can then be approximated by

$$\begin{aligned} p(\boldsymbol{x}_{0:t}|\mathcal{D}) \approx p(\boldsymbol{x}_{0:t}|S_{0:t}) &= \frac{p(S_{0:t}|\boldsymbol{x}_{0:t})p(\boldsymbol{x}_{0:t})}{p(S_{0:t})} \\ &= \frac{p(S_t|\boldsymbol{x}_t)p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})p(\boldsymbol{x}_{0:t-1}|S_{0:t-1})}{p(S_t|S_{0:t-1})} \end{aligned} \tag{6.23}$$

where the second equality yields a recursive formulation for the posterior distribution. As all distributions are Gaussian, Equation (6.23) boils down to the normal Kalman update step, using the summary statistic S_i as measurement for each timestep. The procedure is summarized in Algorithm 4.

Algorithm 4 SL update for GP-EKF

procedure GP-EKF-SL($\hat{\boldsymbol{x}}_t$, $\hat{\boldsymbol{P}}_t$, | $\mathbf{R}$, r)
  $\mathcal{Y}_t = \mathcal{Y}_t(\boldsymbol{x}_t) = \{\boldsymbol{z}_i \in \mathcal{D} : \ ||\boldsymbol{x}_t - \boldsymbol{z}_i|| \leq r\}$ ▷ Relevant measurements
  $S_t = \frac{1}{N}\sum_{\boldsymbol{z}_i \in \mathcal{Y}_t} \boldsymbol{z}_i$ ▷ Compute summary
  $\boldsymbol{W}_t = \hat{\boldsymbol{P}}_t(\hat{\boldsymbol{P}}_t + \frac{1}{N}\boldsymbol{R})^{-1}$ ▷ Kalman Gain
  $\boldsymbol{x}_t = \hat{\boldsymbol{x}}_t + \boldsymbol{W}(S_t - \hat{\boldsymbol{x}}_t)$ ▷ Conditional Mean
  $\boldsymbol{P}_t = (\boldsymbol{I} - \boldsymbol{W}_t)\hat{\boldsymbol{P}}_t$ ▷ Conditional Variance
  return $\boldsymbol{x}_t$, $\boldsymbol{P}_t$
end procedure

The derivations in this section blindly make quite substantial assumptions that can seriously impact the results.

1. The derivations claim the summary statistics to be approximately Gaussian by the central limit theorem. The underlying assumption is that the data is i.i.d and that there is a large number of samples. As the samples originate from several types of vessels and from different trajectories, there is reason to suspect the samples might not be i.i.d.
2. The measurement model in Equation (6.20) assumes that the measurements are distributed around the target vessel's trajectory. In practice, however, the samples might be heavily influenced by several different trajectories at once. This makes tuning $\boldsymbol{R}$ a challenge. More complex measurement models can be used by replacing the mean and variance of $p(S_t|\boldsymbol{x})$ by estimates [39].

6.3.2 Probabilistic Data Association Filter

The SL update ends up using an overly simplistic measurement model and does not consider the possibility that some measurements may originate from different underlying trajectories. Thus, for the target vessel following a specific underlying trajectory, the dataset may actually contain substantial amounts of false measurements, which negatively affect the prediction. As the SL update has no notion of false measurements, the result is overconfident uncertainty measurements, as can be seen in Figure 6.2.

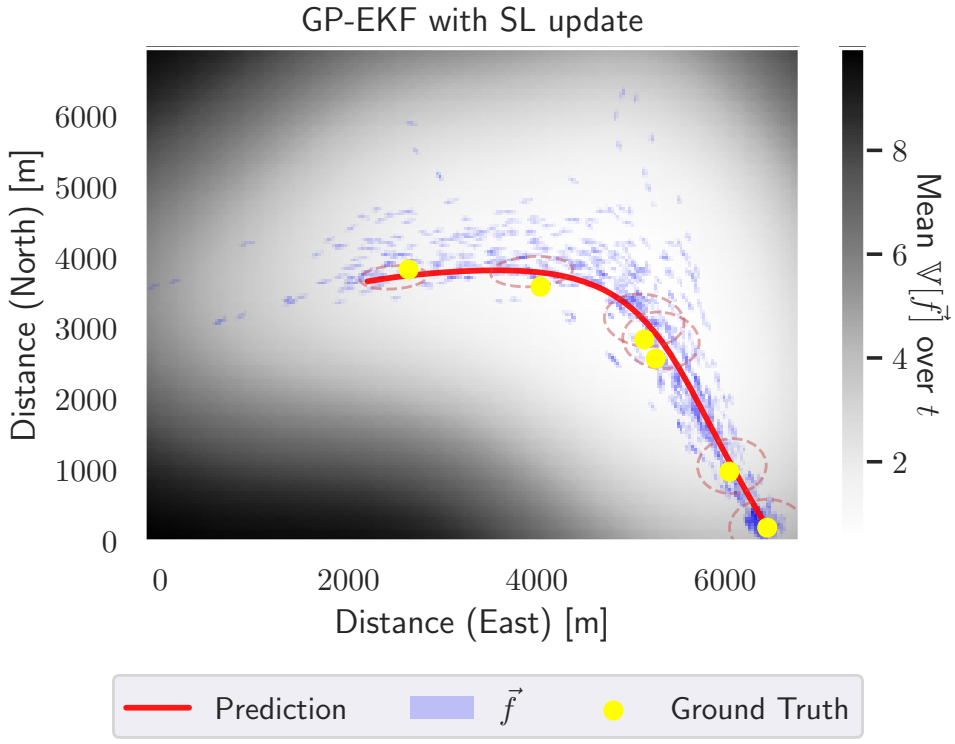
This section attempts a rather different approach by incorporating the notion of data association into the mix. More specifically, the idea is to view the entire problem as single-target tracking and then apply the *Probabilistic Data Association Filter (PDAF)* to solve the problem.

The PDAF is a method commonly used in target tracking which combines data association and filtering. The following introduction to PDAF is mostly inspired by [34], with some adaptations to better fit the current problem. As single target tracking and data association is not the topic of this thesis, only a short introduction to the PDAF will be included here. For more details, see [34, 40].

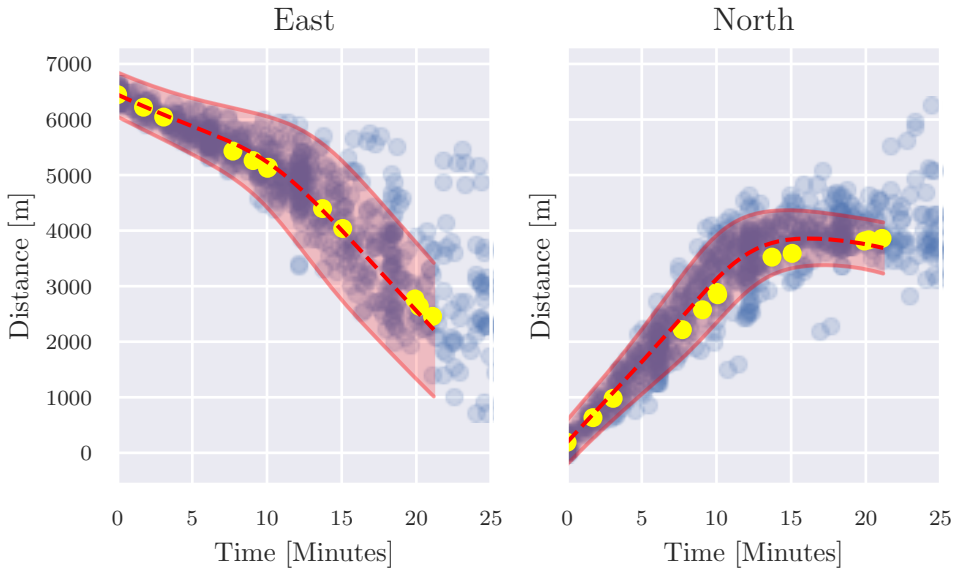
In this section, all measurements in the available training set are considered *virtual* [2] position measurements, which may or may not originate from the vessel at time t. It is assumed that **at most one** measurement may originate from the target to reduce the computational complexity significantly [34]. The rest of the measurements are assumed to be clutter.

Given the predicted state $\hat{\boldsymbol{x}}_t$, any real measurement is expected to be distributed around this state due to some measurement noise. Following the notation used by EKF, and using the measurement model $h(\boldsymbol{x}) = \boldsymbol{x} \implies H = \frac{\partial h(\boldsymbol{x})}{\partial \boldsymbol{x}} = I$, the predicted measurement distribution is expressed as in Equation (6.24) where the innovation covariance is defined as $\boldsymbol{S}_t \triangleq \hat{\boldsymbol{P}}_t + \boldsymbol{R}$ and $\boldsymbol{R}$ is the measurement noise.

[2] Virtual meaning a measurement that did not originate from the target vessel, but rather a measurement that could potentially originate from the target in the future. The word measurement is still used to keep the terminology similar to what is used by PDAF.



(a) Trajectory plotted against the vector-field $\vec{f}(\boldsymbol{x})$. The ellipses show the 95% credibility interval for the predicted trajectory at the ground-truth timestamps.



(b) 95% credibility interval for the trajectory plotted against time.

Figure 6.2: Illustrative example of the GP-EKF with the SL update procedure in practice.

$$\hat{\boldsymbol{z}}_t \sim \mathcal{N}(\hat{\boldsymbol{x}}_t, \boldsymbol{S}_t) = \mathcal{N}(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{P}}_t + \boldsymbol{R}) \tag{6.24}$$

There is also the possibility that none of the measurements originated from the target, and all observations are clutter. A good clutter model is a complicated topic, but the Poisson clutter model is used in this thesis. As the measurements are not actual measurements from the target, it is difficult to assign meaning to any clutter model. It therefore simply boils down to which parameters need to be tuned [3] and the Poisson clutter model should already be familiar to anyone with experience in target tracking. Using the Poisson clutter model, the association probabilities are given by Equation (6.25) [34], where a_t is a discrete variable following a Categorical distribution and $a_t = k > 0$ denotes that measurement k originated from the target. $a_t = 0$ is the special case when none of the measurements originated from the target, i.e. the predicted state should not be updated. Z denotes a matrix of all the measurements (positions) available in the training data and is independent of time, i.e. all measurements are always potential candidates. λ denotes the clutter rate, and P_D denotes the probability of detecting the target vessel.

$$\Pr\{a_t|Z\} \propto \begin{cases} \lambda(1-P_D) & a_t = 0 \\ P_D\mathcal{N}(\boldsymbol{z}^{a_t}|\hat{\boldsymbol{x}}_t, \boldsymbol{S}_t) & a_t > 0 \end{cases} \tag{6.25}$$

Using the likelihood for each of the possible outcomes, the association probabilities $\boldsymbol{\beta}$ can be computed by normalizing the likelihood, i.e.

$$\beta_t^{a_t=i} = \frac{\Pr\{a_t = i \mid Z\}}{\sum_{k=0}^{M} \Pr\{a_t = k \mid Z\}} \tag{6.26}$$

The predicted state can then by updated using the Kalman update procedure for each measurement individually. As the measurements $\boldsymbol{z}_t^{a_t>0}$ are known values, the state innovation $\boldsymbol{\nu}_t^{a_t>0} \triangleq \boldsymbol{z}_t^{a_t>0} - \hat{\boldsymbol{z}}_t$ is distributed according to the measurement prediction $\hat{\boldsymbol{z}}_t$.

$$\boldsymbol{\nu}_t^{a_t>0} \sim \mathcal{N}(\boldsymbol{z}_t^{a_t>0} - \hat{\boldsymbol{x}}_t, S_t) \tag{6.27}$$

The updated state for each measurement is then given by the normal EKF update step

$$\boldsymbol{x}_t^{a_t>0} = \hat{\boldsymbol{x}}_t + \boldsymbol{W}_t \boldsymbol{\nu}_t^{a_t>0} \tag{6.28a}$$

$$\boldsymbol{P}_t^{a_t>0} = (\boldsymbol{I} - \boldsymbol{W}_t)\hat{\boldsymbol{P}}_t \tag{6.28b}$$

where $\boldsymbol{W}_t \triangleq \hat{\boldsymbol{P}}_t \boldsymbol{S}_t^{-1}$ is the Kalman gain. The updated state of the vessel over all possible measurements can be described as a Gaussian Mixture Model over the

[3] The trajectory prediction is here considered to be target tracking of future position. The clutter parameters therefore need to be interpreted in the context of target tracking, not trajectory prediction.

$M+1$ different modes weighted by the association probabilites, i.e.

$$p(\boldsymbol{x}_t) = \underbrace{\beta_t^{a_t=0}\mathcal{N}(\boldsymbol{x}_t \mid \hat{\boldsymbol{x}}_t, \hat{\boldsymbol{P}}_t)}_{\text{No measurements are valid}} + \sum_{k=1}^{M} \underbrace{\beta_t^{a_t=k}\mathcal{N}\big(\boldsymbol{x}_t | \boldsymbol{x}_t^{a_t=k}, \boldsymbol{P}_t^{a_t=k}\big)}_{\text{Measurement } k \text{ is valid}} \tag{6.29}$$

Moment reduction is then used to combine the different hypotheses into a single unimodal Gaussian distribution, i.e. a Gaussian distribution is fitted to the first and second moment (mean and variance) of the Gaussian mixture. The mean and variance of the resulting distribution is given by Equation (6.30a) and Equation (6.30b) respectively, using $\boldsymbol{\nu}_t \triangleq \sum_{a_t>0} \beta_t^{a_t} \boldsymbol{\nu}_t^{a_t}$.

$$\boldsymbol{x}_t = \hat{\boldsymbol{x}}_t + \boldsymbol{W}_t \boldsymbol{\nu}_t \tag{6.30a}$$

$$\boldsymbol{P}_t = \hat{\boldsymbol{P}}_t - (1-\beta_t^0)\boldsymbol{W}_t \boldsymbol{S}_t \boldsymbol{W}_t + \underbrace{\boldsymbol{W}_t\Big[\sum_{a_t>0}^{M} \beta_t^{a_t} \boldsymbol{\nu}_t^{a_t}(\boldsymbol{\nu}_t^{a_t})^\intercal - \boldsymbol{\nu}_t\boldsymbol{\nu}_t^\intercal\Big]\boldsymbol{W}_t^\intercal}_{\text{spread of innovation}} \tag{6.30b}$$

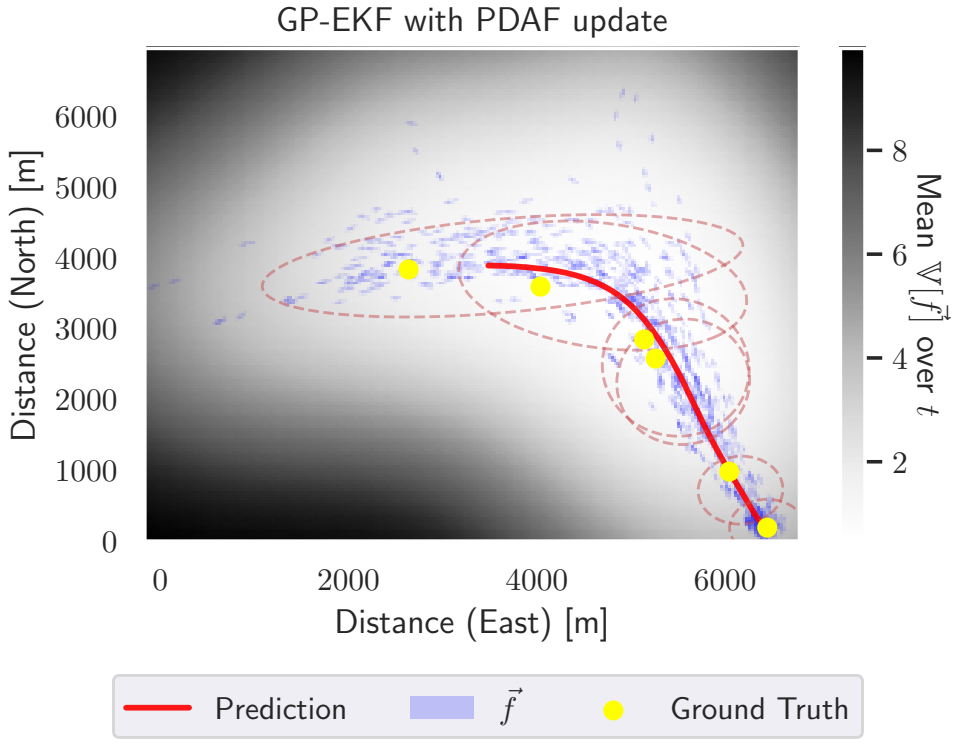
While available measurements could be used at each timestep, it is in practice more convenient to only include a subset that is close enough to the predicted state. As this measurement *gate* should scale with the uncertainty, the gated subset is selected as Equation (6.31), where g is the number of standard deviations that the method should consider.

$$\mathcal{G} = \left\{\boldsymbol{z}^{a_t>0} \mid (\boldsymbol{z}^{a_t>0} - \hat{\boldsymbol{x}}_t)^\intercal S^{-1} (\boldsymbol{z}^{a_t>0} - \hat{\boldsymbol{x}}_t) < g^2\right\} \tag{6.31}$$

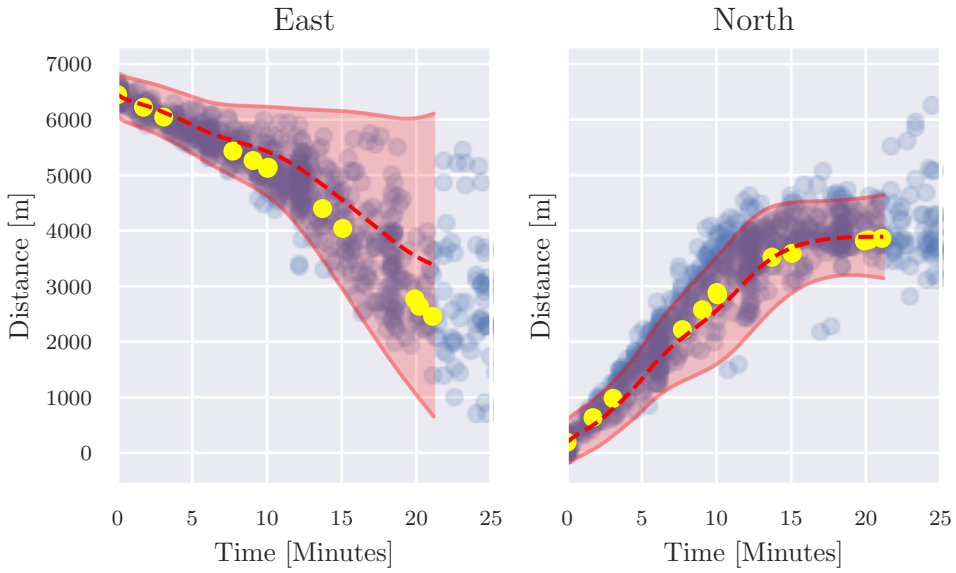
By combining the PDAF update with the GP-EKF prediction procedure in Algorithm 3, the predicted trajectory can be tuned to favor areas with a large number of samples, effectively pulling the state towards areas with available samples. Conversely, in regions with samples spread evenly around the predicted state, the PDAF's effect is then negligible (assuming proper tuning).

6.3.3 Tuning the parameters

The model should primarily trust the prediction model, since it would get stuck as the measurements do not change over time. If the SL or the PDAF parameters are tuned too aggressively, it tends to affect the velocity estimates of the prediction negatively. However, finding a set of parameters that works well across different trajectories turns out to be challenging. Parameters that improve the prediction for one trajectory may do more harm than good on another. The performance of using SL and PDAF will be explored further in Chapter 7.



(a) Trajectory plotted against the vector-field $\vec{f}(\boldsymbol{x})$. The ellipses show the 95% credibility interval for the predicted trajectory at the ground-truth timestamps.



(b) 95% credibility interval for the trajectory plotted against time.

Figure 6.3: Illustrative example of the GP-EKF with the PDAF update procedure in practice.

Algorithm 5 PDAF update for GP-EKF

procedure GP-EKF-PDAF($\hat{\boldsymbol{x}}_t$, $\hat{\boldsymbol{P}}_t$, | $\mathbf{R}$, λ, p_D, g)
  $\boldsymbol{S}_t = \hat{\boldsymbol{P}}_t + \boldsymbol{R}$ ▷ Innovation Covariance
  $\boldsymbol{W}_t = \hat{\boldsymbol{P}}_t \boldsymbol{S}_t^{-1}$ ▷ Kalman Gain
  for each measurement $\boldsymbol{z}^{a_t>0}$ **do**
    $\boldsymbol{\nu}_t^{a_t>0} = \boldsymbol{z}^{a_t>0} - \hat{\boldsymbol{x}}_t$ ▷ Innovation
    $\tilde{\beta}^{a_t>0} = \mathcal{N}(\boldsymbol{\nu}_t^{a_t>0} \mid 0, \boldsymbol{S}_t)$ ▷ Unnormalized Weight
  end for
  $\tilde{\beta}_t^{a_t=0} = \lambda(1-p_D)$ ▷ Unnormalized Clutter Probability
  $\boldsymbol{\beta} = \frac{\tilde{\boldsymbol{\beta}}}{\sum_{a_t} \tilde{\beta}^{a_t}}$ ▷ Normalize weights
  $\boldsymbol{\nu}_t = \sum_{a_t>0} \beta^{a_t} \boldsymbol{\nu}_t^{a_t}$
  $\boldsymbol{x}_t = \hat{\boldsymbol{x}}_t + \boldsymbol{W}_t \boldsymbol{\nu}_t$ ▷ Update the state mean
  $\tilde{\boldsymbol{P}}_t = \boldsymbol{W}_t \big[\sum_{a_t>0}^{M} \beta_t^{a_t} \boldsymbol{\nu}_t^{a_t} (\boldsymbol{\nu}_t^{a_t})^\intercal - \boldsymbol{\nu}_t \boldsymbol{\nu}_t^\intercal \big] \boldsymbol{W}_t^\intercal$ ▷ Spread of innovation
  $\boldsymbol{P}_t = \hat{\boldsymbol{P}}_t - (1-\beta_t^0) \boldsymbol{W}_t \boldsymbol{S}_t \boldsymbol{W}_t + \tilde{\boldsymbol{P}}_t$ ▷ Updated state uncertianty
  return $\boldsymbol{x}_t$, $\boldsymbol{P}_t$
end procedure

6.4 Simulating trajectories using Gaussian Process Sequential Monte Carlo

The Kalman-based prediction scheme proposed in the previous section works well as long as a single Gaussian distribution can sufficiently explain the uncertainty. However, in branching trajectories, minor differences in position early in the predicted trajectory might significantly affect the predictions that follow. The result is a multimodal trajectory distribution, which a Kalman-based approach is not able to express.

Inspired by the prediction step used by *particle filters* [34], the idea of sampling trajectories can be used to explore the multimodal trajectory distribution. While inspired by the particle filter, this approach will from now on be referred to as *Sequential Monte-Carlo* to avoid confusion[4], as well as to follow the same naming convention as used by Ellis *et al.* [12].

The derivation of the Sequential Monte-Carlo approach is embarrassingly simple, as this method trades high computational complexity for more straightforward mathematics. Instead of analytical propagation of uncertainty, many trajectories are simulated through random sampling and used to express the uncertainty empirically. As a result, the uncertainty for any trajectory distribution can be described, though at the cost of considerable computational complexity. N different trajectories are initialized with similar initial conditions. The trajectories are then simulated by drawing independent increments from $\vec{f}$ using the approach described in Section 3.7, conditioned on the current state of each trajectory. The

[4] A vital part of the particle filter is weighting the particles based on available measurements. As only the sampling of trajectories is performed, Sequential Monte-Carlo seems like a better fit.

result is visualized in Figure 6.4 for $N = 1000$ trajectories.

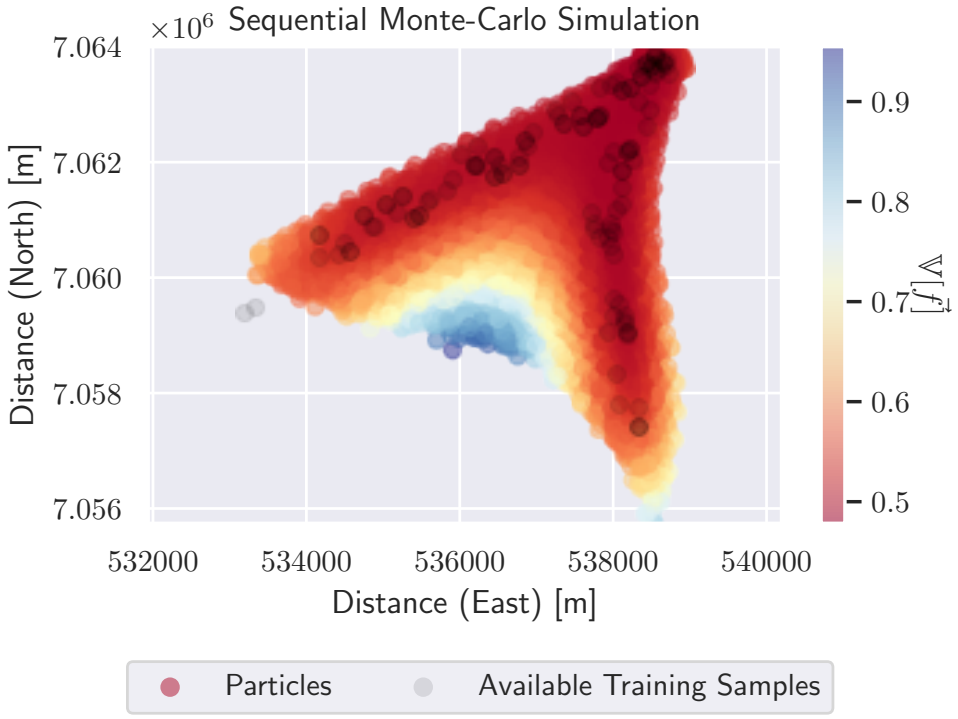


Figure 6.4: Trajectories simulated by sampling from $\vec{f}$. Notice how it is able to represent the multimodal trajectory distribution. The color represent the motion model's confidence when predicting the next step.

6.5 Training Source

There are two potential sources for the training samples $\boldsymbol{y}$:

1. Trajectories from the dataset can be converted into training samples for $\vec{f}$ using the simple first-order finite-difference method in Equation (6.32) between subsequent AIS samples in a trajectory i. This yields the training outputs $\boldsymbol{y}_t^i$ corresponding to the inputs $\boldsymbol{\eta}_t^i$ such that $\boldsymbol{y}_t^i = \vec{f}(\boldsymbol{\eta}_t^i) + \epsilon$.
2. The COG and SOG from the AIS data can be used directly.

$$\boldsymbol{y}_t = \frac{\boldsymbol{x}_{t+1} - \boldsymbol{x}_t}{\tau_{t+1} - \tau_t} \tag{6.32}$$

A concern with using the COG and SOG from the AIS is that this assumes the vessel's own estimates to be accurate. The effect of this design choice is investigated further in Chapter 7.

6.6 Choice of kernel

The approximation used to calculate the posterior distribution in Equation (6.4) is only a first-order Taylor approximation. Hence, it is only a reasonable approximation for functions with a low degree of nonlinearity, and care should be taken when designing the GP dynamics $\vec{f}$. In this thesis, it is argued that an RBF kernel is the preferred choice due to the highly smooth characteristics of the functions drawn from this kernel. The RBF kernel

$$k(\boldsymbol{\eta}, \boldsymbol{\eta}') = \sigma^2 \exp\Big[-\frac{1}{2}(\boldsymbol{\eta}-\boldsymbol{\eta}')^\intercal \boldsymbol{W}^{-1}(\boldsymbol{\eta}-\boldsymbol{\eta}')\Big] \tag{6.33}$$

is found to work well in most cases, as long as the vessel follows a reasonably smooth trajectory. $\boldsymbol{W}$ is a diagonal matrix containing separate length scales for each input dimension. The RBF is the go-to kernel for the GP-EKF in this thesis, though other kernels, such as the Matern class of kernels, may work just as well, if not better.

The derivative of this kernel with respect to the i-th input dimension $\boldsymbol{\eta}[i]$ can be computed as [12, 13]

$$\frac{\partial k(\boldsymbol{\eta}, \boldsymbol{\eta}')}{\partial \boldsymbol{\eta}[i]} = -W_{ii}^{-1}(\boldsymbol{\eta}[i]-\boldsymbol{\eta}'[i])\sigma^2 \exp\Big[-\frac{1}{2}(\boldsymbol{\eta}-\boldsymbol{\eta}')^\intercal \boldsymbol{W}^{-1}(\boldsymbol{\eta}-\boldsymbol{\eta}')\Big] \tag{6.34}$$

A possible extension is to use the sum of two RBF kernels to get some additional flexibility while still having a reasonably smooth function. This is the same kernel proposed in Chapter 5, though without the COG and SOG input dimensions. It works better than the RBF kernel in some of the more complicated cases, though at the cost of having to tune more hyperparameters. In simple cases, hyperparameter optimization **typically** reduces the scale parameter of one of the RBF kernels such that the dependent noise kernel k_1 has a negligible effect. It is important to stress the word "typically," as this kernel is also more prone to finding bad local optima during hyperparameter optimization due to the additional parameters [11]. Consequently, it may end up overfitting. If the length scales become too small, the dynamics $\vec{f}$ are allowed to become highly non-linear, and small changes in the input $\boldsymbol{x}$ may have a large impact on the prediction. The Taylor approximation is not a reasonable approximation in these cases, and the result is unstable uncertainty estimates [5]. The key takeaway is that the more flexible kernels are also more prone to overfitting, and care should be taken during optimization to avoid unrealistic parameters.

Other kernels may be preferred if the posterior density in Equation (6.4) is evaluated using sampling-based methods, such as Sequential Monte-Carlo, as the Taylor approximation does not limit these methods. However, it is not covered by this thesis.

[5] This is typically seen in the uncertainty estimates as large steps at seemingly random time steps, where the uncertainty experiences sudden jumps as some elements in the Jacobian become large.

6.6.1 Optimizing the hyperparameters

The hyperparameters can be optimized using the MLE approach discussed in Section 3.6.1.

The whole dataset cannot be used to find the optimal hyperparameters due to the computational complexity. Two approaches are proposed to mitigate this issue:

1. Find a set of hyperparameters that work well across a wide range of cases. Model selection is then be performed in an offline context, and more extensive testing can be performed to ensure a good model is selected. One possible approach is to use Monte-Carlo simulation to find a set of robust hyperparameters which empirically works well. However, the hyperparameters may become too general for the more challenging scenarios.
2. Rerun the hyperparameter optimization for each scenario. This approach tends to find the optimal parameters in each case. However, it is also error-prone as it is not guaranteed to find a global optimum, and it may cause overfitting.

The first approach is likely the preferred choice in practical applications, as the hyperparameter optimization is prone to bad local optima, and the results should ideally be inspected before deployment to real-world applications. A possible extension is to find a global set of parameters for each distinct vessel type, assuming a vessel's behavior is somewhat consistent across the entire region.

6.7 Summary

This chapter has introduced quite a few new concepts so that it might be nice with a summary of the overall pipeline:

1. Based on the target vessel's current state, such as position, COG and SOG, trajectories with similar initial conditions are selected from the training set.
2. Training inputs $\boldsymbol{\eta}$ and outputs $\boldsymbol{y}$ for $\vec{f}$ are generated from the training trajectories.
3. The kernel hyperparameters are optimized using MLE to find the parameters that best explain the training data. This is either done once to find a global set of parameters or individually for each scenario.
4. The training data and kernel can then used to get the conditional distribution for $\vec{f}$.
5. The EKF procedure in Algorithm 3 is called repeatedly, using the current state of the target vessel as initial conditions. At each step, if desirable, the SL or the PDAF update procedure can be used as well. Sequential Monte-Carlo can alternatively be used to simulate multimodal trajectories.

The only parameters that need manual tuning for the GP-EKF are the time increment $\Delta\tau$ and the initial covariance $\boldsymbol{P}_0$. All other parameters are learned from the data.

Chapter 7

Statistical Testing

Inspired by [4], the performance of the proposed models will be compared on straight-line and curved trajectories independently. This comparison is intended to showcase the methods' ability to consistently predict reasonable trajectories and motivate further research into GP based methods. However, the content of this chapter is not intended to showcase a perfect solution, and the methods are still heavily influenced by how the hyperparameters are selected, as well as any tuning parameters.

7.1 Method

The simulation will test the performance on a total of 700 different test trajectories, where one half corresponds to straight-line trajectories and the remaining half consists of curved trajectories. Trajectories are selected from the dataset proposed in Chapter 4, and will have a total duration of between 15 and 30 minutes. The Direct GP and GP-EKF from Chapter 5 and Chapter 6, respectively, will be tested on the same scenarios as defined in Chapter 4 and get access to the exact same training data.

Three different metrics – trajectory error, path error and normalized estimation error squared – will be used to compare the performance of the various methods.

7.1.1 Trajectory Error

The trajectory error is found by comparing the predicted mean position with the ground truth. As the predicted trajectory is simulated in discrete time, the points with the closest timestamps are used for comparison. The simulation will use $\Delta\tau = 10$ seconds, which yields a maximum error in time $\frac{\Delta\tau}{2} = 5$ seconds, which is considered to be acceptable considering the time-horizon of between 15 and 30 minutes.

7.1.2 Path Error

The path error is defined as the closest point in the predicted trajectory to each point in the ground truth, under the constraint that the corresponding predicted timestamps must be monotonically increasing. In other words, the path cannot move backwards in time.

7.1.3 Normalized Estimation Error Squared

For the uncertainty estimates to provide any value, the predictions must be consistent. In this context, the term consistency is borrowed from the term *filter consistency* used when tuning Kalman filters [34]. The idea is that prediction errors, on average, should scale with the state covariance. In other words, the model should not place much confidence in a prediction that is wrong while being highly confident when a prediction is correct. Consistency can also be interpreted using a frequentistic interpretation of probability, where after many predictions, the state uncertainty should reflect the actual error rate.

The *Normalized Estimation Error Squared (NEES)* is a metric that can be used to quantify consistency and is given by

$$\text{NEES} = (\boldsymbol{x} - \hat{\boldsymbol{x}})^\intercal \boldsymbol{P}^{-1} (\boldsymbol{x} - \hat{\boldsymbol{x}}) \tag{7.1}$$

where $\boldsymbol{x}$ is ground truth and $\hat{\boldsymbol{x}}$ refers to the prediction.

Assuming that the prediction error follows a Gaussian distribution, the NEES follows a Chi-Squared distribution which can be used to form a confidence interval. Comparing the prediction errors with this confidence interval can then be used to get a sense of whether the estimated state uncertainty is consistent with the actual error rate.

7.1.4 Comparing distributions - Boxplot

The boxplot is used to visualize the distribution of the different metrics. The boxplot visualizes the estimated quartiles, i.e. the 25%, 50%, and 75% quantiles, of the distribution. The "whiskers" show the range of values not considered to be outliers. Points outside of $[Q_{25} - 1.5 \text{ IQR}, Q_{75} + 1.5 \text{ IQR}]$ are considered outliers, where $\text{IQR} = Q_{75} - Q_{25}$ is the inter-quartile range [41].

7.1.5 Interpolation

The metrics will be compared at fixed 5 minute intervals using linear interpolation. The error for short trajectories is not extrapolated, so there might be fewer available samples for the metrics when moving past 15 minutes.

7.1.6 Baseline - Constant Velocity Model

As a basis of comparison, the *Constant Velocity Model (CVM)* method is used as a baseline. The model uses the initial COG and SOG to predict a straight line, where the vessel is assumed to keep a constant velocity and heading.

7.1.7 Selecting test cases

The two datasets from Chapter 4 are used to perform the statistical testing.

However, 54% of the dataset originates from only three ferries. Distinct discontinuities characterize the trajectories of these ferries as the vessels dock for short periods before moving back the same way they came. These discontinuities directly contradict the assumptions of smoothness made by the GPs used in this thesis. Whether or not to include these vessels was a challenging decision, as a COLAV will need to be able to handle ferries, just as any other vessel type. However, in a practical application, it is natural to distinguish between different vessel types and allow the use of more specialized models to handle cases such as frequently docking ferries. As no such distinction is made in this thesis, the statistical testing will be more insightful without using edge-cases for more than 50% of the test cases. While it would be possible to not completely remove these vessels, i.e. perform subsampling, it is considered more confusing. Therefore, the samples from these three vessels are removed from the test set, though they are still present in the training set. This thesis has not proposed any method for filtering the training data based on the MMSI; removing these vessels from the training set would be an unfair advantage.

The result is an artificial bias favoring the GP framework proposed in this thesis, as the fraction of smooth trajectories has been artificially increased.

Straigth-line trajectories

The methods are first compared to the CVM on simple straight-line trajectories. The statistics are based on 350 randomly sampled trajectories without replacement that satisfy the following requirements:

1. The sum of subsequent changes in COG must be less than 30 degrees, i.e. $\sum_i |(\mathcal{X}_{t+1} - \mathcal{X}_t)| \leq 30^\circ$. This requirement ensures a straight-line trajectory. Additional care is always taken to use the smallest possible difference as the angles $\mathcal{X} \in [0^\circ, 360^\circ)$ wraps around.
2. There must be sufficient data available for training in the neighborhood around the initial starting point, with similar initial heading and speed. After sanitizing the dataset and removing irrelevant trajectories, at least 3 trajectories need to be available for training.
3. The overall duration of the trajectories must be between 15 and 30 minutes to be considered a test candidate.

Curved Trajectories

The curved trajectory statistics are based on 350 randomly sampled trajectories that satisfy the following requirements:

1. The sum of subsequent changes in COG must be greater than 40 degrees, i.e. $\sum_i |(\mathcal{X}_{t+1} - \mathcal{X}_t)| \geq 40°$. This requirement ensures a minimum curvature in the trajectory. Additional care is always taken to use the smallest possible difference as the angles $\mathcal{X} \in [0°, 360°)$ wraps around.
2. There must be sufficient data available for training in the neighborhood around the initial starting point, with similar initial heading and speed. After sanitizing the dataset and removing irrelevant trajectories, at least 3 trajectories need to be available for training.
3. The overall duration of the trajectories must be between 15 and 30 minutes to be considered a test candidate.

7.1.8 Training Data

The initial conditions for each test trajectory are considered a distinct scenario, and the relevant training data is selected according to the method proposed in Chapter 4. The requirements are restated here with the concrete values used for testing:

1. The trajectories' initial position must be close to the target vessel's position $\boldsymbol{x}_0$. A fixed threshold at $||\Delta \boldsymbol{x}_0|| \leq 200$ m is used, where $\Delta \boldsymbol{x}_0$ is the difference between the target vessel's current position and the initial conditions of a potential training trajectory.
2. The trajectories' initial COG must be close to the target vessel's heading $\mathcal{X}$. A fixed threshold at $\mathcal{X} \pm 20°$ is used, with additional care taken when the angles wrap.
3. The trajectories' initial SOG must be close to the queried velocity v. A fixed threshold at $v \pm 4$ knots is used.

Due to the way trajectories are generated from the AIS dataset, there will be significant overlaps between trajectories. Therefore, naively dividing the trajectories into a train and test set is considered insufficient, as sub-trajectories from the test set might also exist in the training set. Instead, the entire dataset is available for training, but all trajectories with identical MMSI and date as the test trajectory are removed before training. The date requirement ensures that trajectories for the same vessel can be used for training on any other day. There is still the possibility of leaking data from the test set if the trajectory moves past midnight, but this is assumed only to affect a negligible number of trajectories.

7.2 Implementation

The implementation details for each method are described in this section.

7.2.1 Direct GP

The excact GP formulation is implemented using the `GaussianProcessRegressor` from the Python library, *scikit-learn* [27]. The library supports all kernels introduced in Chapter 3 and supports hyperparameter optimization using multiple restarts to avoid bad local optima.

The slightly more complicated kernel in Equation (5.3) from Chapter 5 is used. This kernel is preferred over a single RBF kernel as it simply yields the best performance across a range of different simulations. The noise term σ^2 is included as a White kernel and optimized like any other hyperparameter. The parameters are optimized for each test scenario, using 10 random restarts to reduce the risk of bad local optima.

7.2.2 GP-EKF

The GP-EKF is tested with and without both the SL and the PDAF update steps, as well as using both finite difference and the COG/SOG from the AIS dataset, resulting in 6 different instances.

The GP-EKF requires more flexibility during development. Due to the need for calculating the gradient $\frac{\partial \vec{f}}{\partial \boldsymbol{x}}$, it is impractical to use existing GP implementations. Due to the simple implementation of Algorithm 1, it is easier to implement it from the ground up rather than adapting existing solutions. The GP-EKF used in this thesis is therefore implemented directly in Python using only *scipy*[42] and *numpy*[43] to speed up linear algebra routines. The Cholesky decomposition in Algorithm 1 can be computed using `scipy.linalg.cho_factor`, which calls a highly optimized LAPACK routine. Similarily, `scipy.linalg.solve_trianglular` can be used to solve the lower triangular system of equations by forward substitution. The implementation of GP-EKF and the PDAF update is then straightforward using Algorithm 3 and Algorithm 5 from Chapter 6. The implementation used for the statistical testing use standardized training outputs $\boldsymbol{y}$.

For statistical testing, the proposed RBF kernel from Equation (6.33) with independent length scales will be used due to its simplicity. More complicated kernels are avoided due to challenges with bad local optima during hyperparameter optimization.

The hyperparameters are tuned using the GP implementation in the popular *scikit-learn* [27] Python package. 10 random restarts are used during optimization to reduce the risk of bad local minima. A lower bound constraint for the length scales is also used to avoid obvious overfitting and requires the length scales to be greater than 50. Due to the wide variety of different scenarios on the training set, the hyperparameters are optimized for each iteration. This way, the robustness of the optimization is indirectly tested, as instances of bad local optima are included in the results.

The remaining parameters not found through MLE are tuned through trial and error on a few different trajectories. The initial state uncertainty is set to $\boldsymbol{P}_0 = 500^2 \cdot \boldsymbol{I}$, which was found to work well during development. The PDAF

parameters are available in Table 7.1a and the SL parameters can be found in Table 7.1b.

Parameter		*Value*
Measurement noise	$\boldsymbol{R}_{\text{PDAF}}$	$500^2 \cdot \boldsymbol{I}$
Detection Probability	p_D	0.8
Clutter Rate	λ	$2 \cdot 10^{-3}$
Gate Size	g	2

(a) Parameters used for PDAF update

Parameter		*Value*
Noise	$\boldsymbol{R}_{\text{SL}}$	$500^2 \cdot \boldsymbol{I}$
Search Radius		50

(b) Parameters used for the SL update

7.3 Results

The error metrics for straight-line and curved trajectories are available in Table 7.2 and Table 7.3 respectively, while the NEES results are available in Table 7.4 and Table 7.5. All tables are found at the end of this chapter. Some hand-picked test scenarios are also included in Appendix A for the different GP-EKF variations.

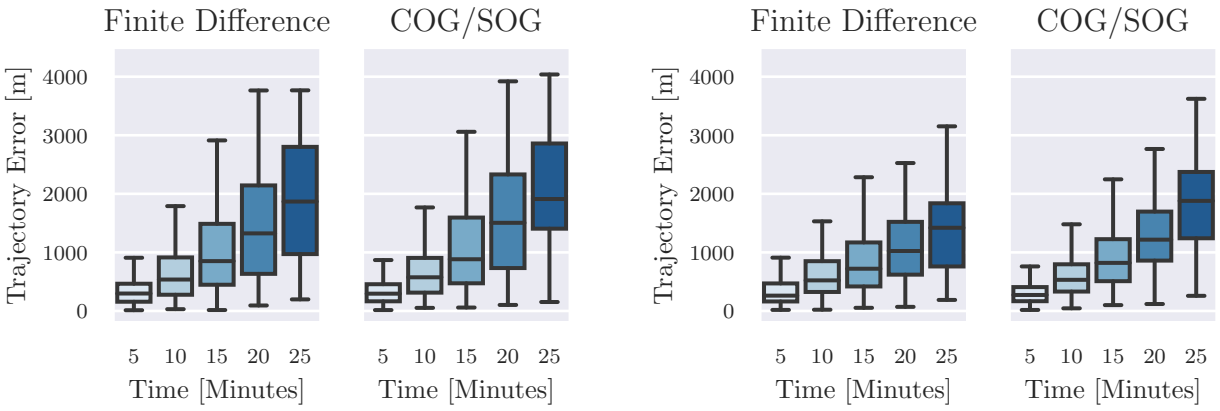
This section will take a hierarchical approach, where the GP-EKF using finite-difference for training data is considered the basic configuration of the GP-EKF and will be used as a baseline when comparing the different variants of the GP-EKF. Unless explicitly stated otherwise, it is safe to assume that the GP-EKF uses finite-difference and no update step. For comparing specific combinations, the reader is referred to Table 7.2 and Table 7.3.

As a base of comparison, the Direct GP and GP-EKF are compared to the CVM on both straight-line and curved trajectories in Figure 7.2. On straight-line trajectories, both methods perform worse than the CVM with higher trajectory error for all respective quartiles. As expected, the CVM struggles on curved trajectories, whereas both the Direct GP and GP-EKF approaches perform significantly better with both lower median error and spread. The Direct GP approach performs slightly better than the GP-EKF, with lower median trajectory error and lower spread for both straight-line and curved trajectories. When comparing the path error for straight-line trajectories in Table 7.2b, the GP-EKF performs consistently better than the Direct GP approach, with lower mean and median path error. The GP-EKF even outperforms the CVM on straight-line paths when predicting beyond 15 minutes.

Both the Direct GP and the GP-EKF approach perform better on curved trajectories as opposed to straight-line trajectories. After manually inspecting the test cases, it appears that many of the straight-line test cases belong to slow-moving vessels that are part of longer curved trajectories. The training samples used to fit the GPs in these cases often contain samples from faster-moving vessels, making the GPs overshoot in their predictions. This hypothesis is corroborated by the path error in Table 7.2b, as the GP-EKF outperforms the CVM on long straight-line paths.

Neither the Direct GP nor the GP-EKF yield consistent uncertainty estimates as found by comparing the NEES to the theoretical $\mathcal{X}^2$ quartiles in Figure 7.4a and Figure 7.4b. For both methods, the NEES is especially bad for straight-line trajectories with estimated quartiles exceeding the theoretical quartiles, which is likely linked to the increased error for the straight-line trajectories. The Direct GP approach performs better than the GP-EKF, with the 25% and 50% quartiles close to the theoretical values, while the basic GP-EKF remains overconfident for all quartiles. The upper quartile for both methods exceeds the theoretical value for the $\mathcal{X}^2$ distribution.

7.3.1 GP-EKF: Finite Difference vs. COG/SOG from AIS



(a) Straight-Line Trajectory **(b)** Curved Trajectory

Figure 7.1: GP-EKF using finite differences and the COG and SOG from the AIS dataset on 350 trajectories. The finite difference approach performs slightly better, with lower median error and spread.

A key design choice for the GP-EKF is deciding which data source to use for training. The model can either be trained using the COG and SOG values contained in the AIS samples or by calculating numerical derivatives of the position through a finite-difference approach. Figure 7.1 compares the difference side-by-side. For straight-line trajectories, the results are remarkably similar. The finite difference approach seems to have a slight advantage on the lower quartile error at $\tau = 25$ minutes, though with an increased spread and the advantage is likely due to random errors. The finite difference performs slightly better than COG/SOG for curved trajectories, especially as time increases, and has both lower median trajectory error and spread.

For the path errors in Table 7.2b and Table 7.3b, the results favor the COG/SOG approach, with overall lower mean and median path errors when compared to the finite difference approach.

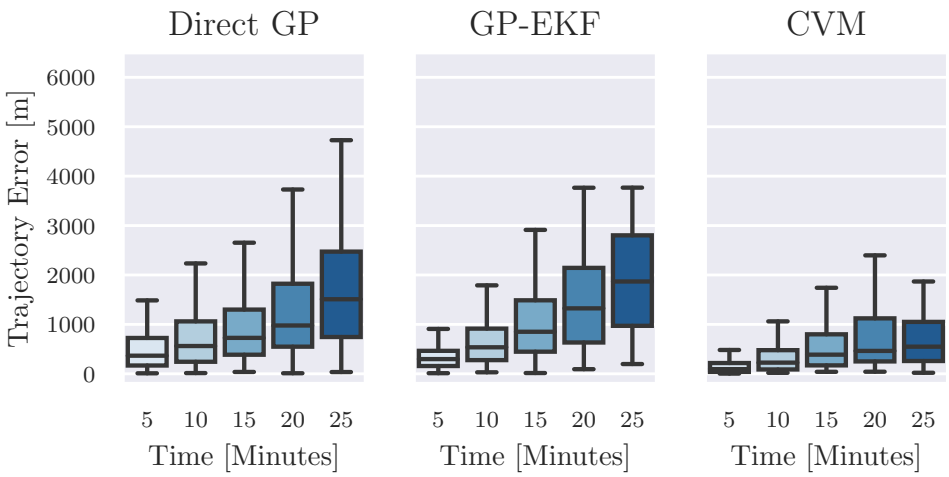
Using COG/SOG yields a considerable NEES improvement over the finite dif-

ference approach, as seen in Figure 7.4c and Figure 7.4d. On curved trajectories, the estimated NEES quartiles are close to the theoretical values when using COG/SOG as input data, while the finite difference approach yields overconfident results. A similar pattern is found for the median NEES values in Table 7.4 and Table 7.5 for all variants of the GP-EKF.

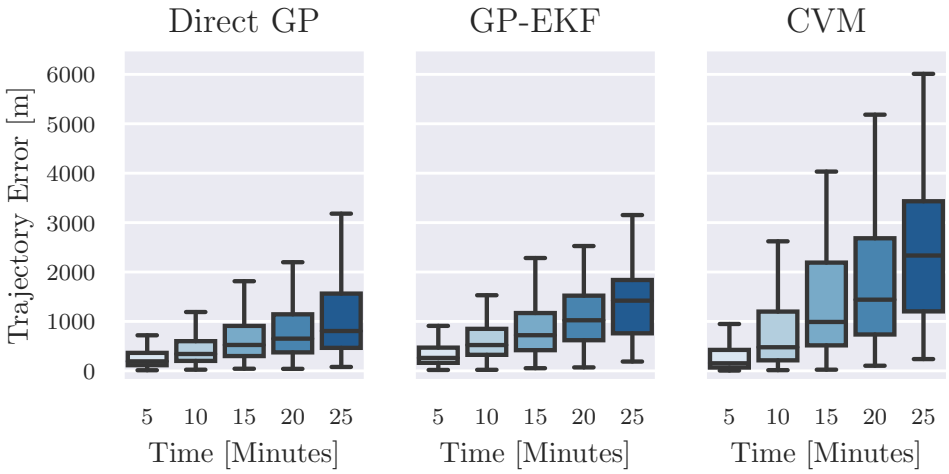
7.3.2 GP-EKF: Incorporating positions data

The effect on the mean prediction for both the PDAF and the SL update procedures is compared to the basic GP-EKF configuration in Figure 7.3. The results are incredibly similar for both straight-line and curved trajectories. On closer inspection, there appears to be a slight advantage to using the PDAF update step, but the improvement is within the margin of error, so it is not possible to draw any definitive conclusion.

However, the SL, and to some extent the PDAF update, have a detrimental effect on the NEES, leading to even more overconfident predictions. This is apparent when comparing the GP-EKF NEES with and without the update steps in Figure 7.4e and Figure 7.4f, as the NEES for the SL update explodes compared to the basic GP-EKF method. Similar patterns are found in Table 7.4 and Table 7.5 regardless of whether the GP-EKF is trained with finite difference or COG/SOG data.

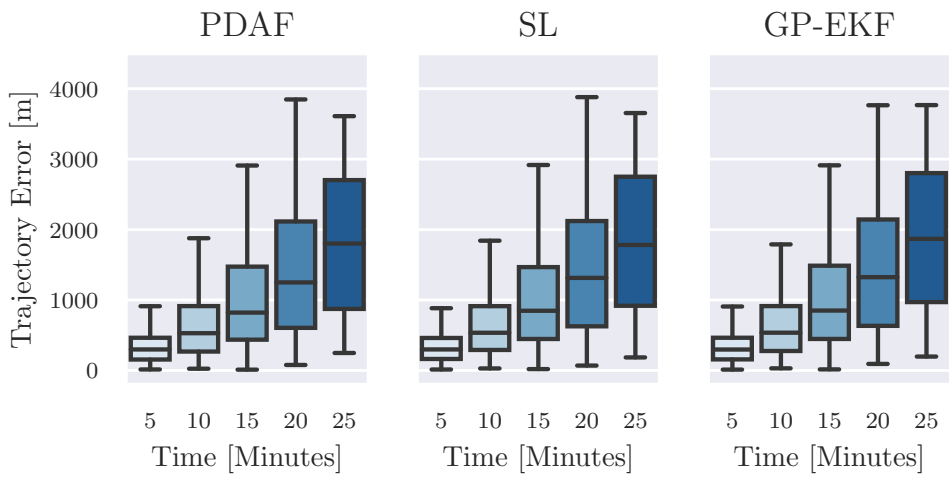


(a) Straight-line trajectories.

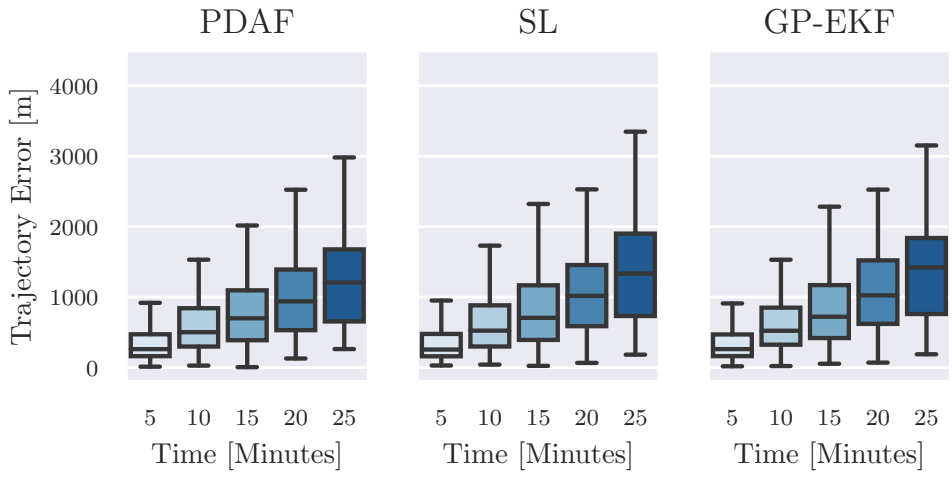


(b) Curved trajectories.

Figure 7.2: GP-EKF and Direct GP compared to a CVM on 350 straight-line and 350 curved trajectories. For straight-line trajectories, the CVM outperforms both the GP-EKF and the Direct GP, while the Direct GP yield slightly better performance than the GP-EKF. For curved trajectories, the CVM struggles as expected. Both the GP-EKF and the Direct GP yields far better results with lower error and lower spread, while Direct GP has the lowest overall trajectory error quartiles. Interestingly, both GP-EKF and Direct GP performs better on curved trajectories than straight-line trajectories, with lower overall error and spread.

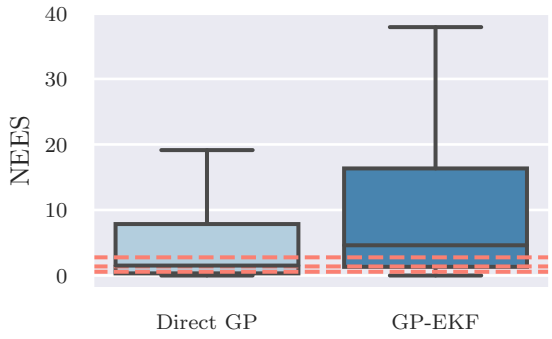


(a) Straight-Line Trajectories

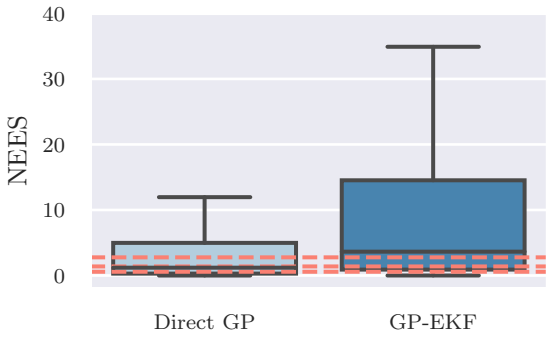


(b) Curved Trajectories

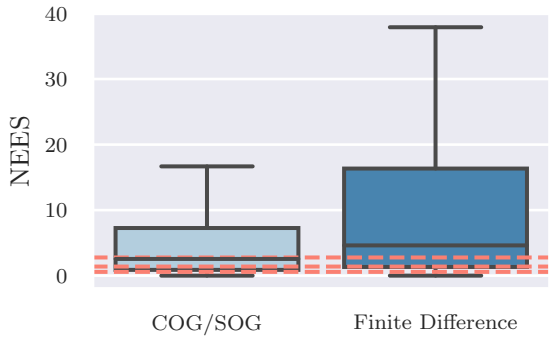
Figure 7.3: GP-EKF with and without the SL and PDAF updates on 350 straight-line and 350 curved trajectories. While there are some slight differences, the PDAF and SL update does not appear to have any considerable effect on the trajectory errors. On straight-line trajectories, the results are almost identical across all three variants. On curved trajectories, the PDAF yields slightly lower error for the median and upper quartiles, though the difference is well within the margin of error.



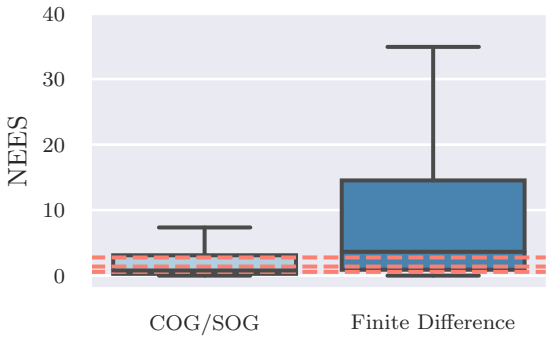
(a) Straight-Line Trajectory NEES Baseline



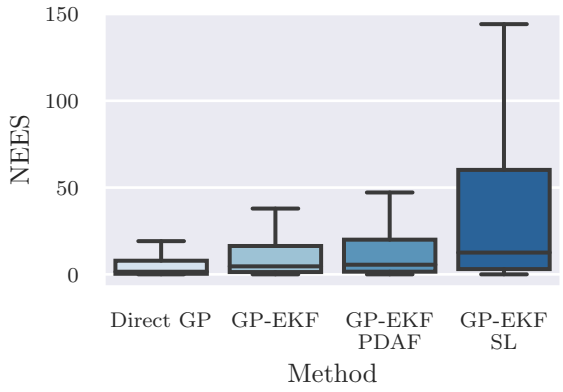
(b) Curved Trajectory NEES Baseline



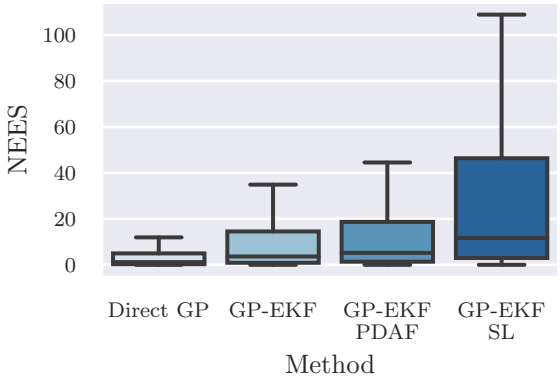
(c) Straight-Line Trajectory COG/SOG vs Finite Difference



(d) Curved Trajectory COG/SOG vs Finite Difference



(e) Straight-Line Trajectory with update steps



(f) Curved Trajectory with update steps

Figure 7.4: A comparison of the NEES for the different methods. **(a)-(b)** compare the NEES for the Direct GP and the base GP-EKF to the theoretical quartile values for the $\mathcal{X}^2$ distribution (red dashed lines) and is intended as a base of comparison. **(c)-(d)** compare the difference in consistency between using COG/SOG and finite difference as input data. **(e)-(f)** compare the effect of the update steps for the GP-EKF. The theoretical quartiles are not included in the last row due to the large differences in scale, and the reader is advised to use the Direct GP and basic GP-EKF for comparison.

Summary	Method	Time [Minutes] Training Source	5	10	15	20	25
Mean	CVM	COG/SOG from AIS	172	383	623	786	772
	Direct GP	Position	603	720	944	1269	1727
	GP-EKF	COG/SOG from AIS	328	668	1090	1560	2048
		Finite Difference	331	656	1029	1417	1917
	GP-EKF w/ PDAF	COG/SOG from AIS	331	668	1070	1494	1921
		Finite Difference	332	657	1025	1392	1856
	GP-EKF w/ SL	COG/SOG from AIS	325	654	1048	1489	1972
		Finite Difference	334	663	1025	1414	1891
Median	CVM	COG/SOG from AIS	94	226	387	463	549
	Direct GP	Position	367	562	727	978	1509
	GP-EKF	COG/SOG from AIS	295	575	883	1505	1911
		Finite Difference	298	537	851	1325	1868
	GP-EKF w/ PDAF	COG/SOG from AIS	301	554	850	1401	1845
		Finite Difference	299	529	822	1250	1802
	GP-EKF w/ SL	COG/SOG from AIS	297	526	829	1324	1805
		Finite Difference	299	536	848	1315	1782

(a) Trajectory errors in meters

Summary	Method	Time [Minutes] Training Source	5	10	15	20	25
Mean	CVM	COG/SOG from AIS	58	180	351	491	504
	Direct GP	Position	362	318	428	575	794
	GP-EKF	COG/SOG from AIS	82	158	245	283	330
		Finite Difference	92	194	309	360	394
	GP-EKF w/ PDAF	COG/SOG from AIS	81	157	248	295	319
		Finite Difference	91	192	309	367	393
	GP-EKF w/ SL	COG/SOG from AIS	87	169	253	298	329
		Finite Difference	99	211	330	396	398
Median	CVM	COG/SOG from AIS	29	82	187	301	418
	Direct GP	Position	97	138	224	311	390
	GP-EKF	COG/SOG from AIS	60	97	150	182	262
		Finite Difference	62	121	188	210	237
	GP-EKF w/ PDAF	COG/SOG from AIS	57	97	145	181	196
		Finite Difference	62	117	190	217	237
	GP-EKF w/ SL	COG/SOG from AIS	62	101	149	189	247
		Finite Difference	66	124	188	213	217

(b) Path error in meters

Table 7.2: Error summary for 350 straight-line trajectories. Mean and median summary statistics are calculated for the trajectory and path error at fixed timestamps. Linear interpolation is used between samples. Errors for short trajectories are not extrapolated, and therefore not included in the 20 and 25 minute bins.

Summary	Method	Time [Minutes] Training Source	5	10	15	20	25
Mean	CVM	COG/SOG from AIS	483	1184	1936	2148	2684
	Direct GP	Position	329	507	717	898	1305
	GP-EKF	COG/SOG from AIS	355	691	1009	1363	1895
		Finite Difference	400	706	897	1159	1552
	GP-EKF w/ PDAF	COG/SOG from AIS	349	663	925	1167	1392
		Finite Difference	398	688	863	1078	1417
	GP-EKF w/ SL	COG/SOG from AIS	351	672	950	1260	1751
		Finite Difference	401	703	885	1134	1540
Median	CVM	COG/SOG from AIS	153	478	990	1441	2334
	Direct GP	Position	192	342	524	652	806
	GP-EKF	COG/SOG from AIS	273	532	821	1218	1880
		Finite Difference	262	522	721	1024	1421
	GP-EKF w/ PDAF	COG/SOG from AIS	262	501	724	961	1133
		Finite Difference	261	502	699	939	1207
	GP-EKF w/ SL	COG/SOG from AIS	264	500	732	1036	1616
		Finite Difference	256	524	706	1018	1333

(a) Trajectory Error in meters

Summary	Method	Time [Minutes] Training Source	5	10	15	20	25
Mean	CVM	COG/SOG from AIS	214	587	1066	1354	1748
	Direct GP	Position	162	222	347	484	765
	GP-EKF	COG/SOG from AIS	159	283	428	451	573
		Finite Difference	194	357	466	492	502
	GP-EKF w/ PDAF	COG/SOG from AIS	153	269	393	424	415
		Finite Difference	190	343	450	482	525
	GP-EKF w/ SL	COG/SOG from AIS	160	285	414	435	512
		Finite Difference	194	357	469	501	542
Median	CVM	COG/SOG from AIS	61	290	729	998	1497
	Direct GP	Position	72	132	225	280	344
	GP-EKF	COG/SOG from AIS	107	184	287	262	313
		Finite Difference	115	221	337	354	331
	GP-EKF w/ PDAF	COG/SOG from AIS	100	171	242	226	219
		Finite Difference	114	210	315	355	385
	GP-EKF w/ SL	COG/SOG from AIS	103	186	249	249	240
		Finite Difference	117	222	319	376	362

(b) Path error in meters

Table 7.3: Error summary for 350 curved trajectories. Mean and median summary statistics are calculated for the trajectory and path error at fixed timestamps. Linear interpolation is used between samples. Errors for short trajectories are not extrapolated, and therefore not included in the 20 and 25 minute bins.

Summary	Method	Time [Minutes] Training Source	5	10	15	20	25
Mean	Direct GP	Position	3.88	11.79	21.07	34.11	42.03
	GP-EKF	COG/SOG from AIS	2.88	83.09	112.68	67.59	24.88
		Finite Difference	25.66	79.24	153.31	64.10	185.33
	GP-EKF w/ PDAF	COG/SOG from AIS	2.99	94.05	126.13	22.34	33.41
		Finite Difference	28.90	89.65	156.13	66.86	137.51
	GP-EKF w/ SL	COG/SOG from AIS	97.91	530.52	582.50	231.68	191.07
		Finite Difference	31.76	102.66	214.88	274.35	530.58
Median	Direct GP	Position	0.66	1.18	2.09	3.23	6.05
	GP-EKF	COG/SOG from AIS	1.30	2.39	3.36	5.64	7.92
		Finite Difference	1.80	4.10	7.26	12.52	25.43
	GP-EKF w/ PDAF	COG/SOG from AIS	1.42	2.85	4.81	9.62	13.24
		Finite Difference	1.99	4.79	8.90	18.12	34.17
	GP-EKF w/ SL	COG/SOG from AIS	2.79	9.13	22.42	49.73	104.53
		Finite Difference	2.96	10.16	27.99	73.56	139.15

Table 7.4: NEES for 350 straight-line trajectories at fixed timestamps. Linear interpolation is used between samples.

Summary	Method	Time [Minutes] Training Source	5	10	15	20	25
Mean	Direct GP	Position	2.74	6.05	10.62	14.79	323.69
	GP-EKF	COG/SOG from AIS	6.08	65.25	235.24	8.86	15.08
		Finite Difference	36.86	28.44	94.24	39.93	89.61
	GP-EKF w/ PDAF	COG/SOG from AIS	6.32	65.66	234.01	13.75	22.44
		Finite Difference	36.94	29.88	59.10	52.79	70.95
	GP-EKF w/ SL	COG/SOG from AIS	10.21	72.72	258.65	92.60	171.51
		Finite Difference	45.92	59.75	114.19	158.79	249.64
Median	Direct GP	Position	0.29	0.85	2.13	3.49	3.91
	GP-EKF	COG/SOG from AIS	0.62	0.76	0.91	1.18	1.85
		Finite Difference	1.53	3.09	4.89	8.72	18.20
	GP-EKF w/ PDAF	COG/SOG from AIS	0.80	1.27	2.16	4.46	3.35
		Finite Difference	1.67	4.07	8.13	13.30	22.23
	GP-EKF w/ SL	COG/SOG from AIS	1.66	6.53	15.23	40.30	77.91
		Finite Difference	2.38	9.56	19.29	44.08	116.12

Table 7.5: NEES for 350 curved trajectories at fixed timestamps. Linear interpolation is used between samples.

Chapter 8

Discussion

The results from Chapter 7 showcase the GP framework's ability to predict a vessel's trajectory using historical AIS data. In this section, the results will be discussed further together with theoretical considerations.

8.1 Choice of kernel

Finding a good kernel that works well across a wide range of scenarios can be quite challenging. The choice of kernel forms a prior belief about the underlying function and has a significant impact on the type of function that the GP will learn. If the kernel is too simple, the hyperparameter optimization struggles to find a set of parameters that sufficiently explain the data. On the other hand, an overly complex kernel may quickly end up overfitting.

In this thesis, the kernels are selected by assuming smooth vessel trajectories and are reasonable for vessels in transit as they tend to change course slowly. More specialized vessels, such as local ferries in the AIS dataset, require more complicated kernels to perform well. These vessels move in more distinct patterns and are harder to predict due to the vessels' behavior when docking for short periods before moving back the same way they came.

The Direct GP approach performs well with more complicated kernels, and the added flexibility does not significantly impact the consistency of the performance. Overfitting is still a concern, but during the development of the Direct GP approach, the more flexible kernels were found to yield higher accuracy in most scenarios. However, this thesis does not focus too much on kernel design, and no statistical testing is performed when using different kernels.

The GP-EKF is more sensitive to the choice of kernel. As already discussed in Chapter 6, instabilities may occur if the kernel and its hyperparameters are not chosen carefully. More complicated kernels certainly have their benefits, but the added complexity during model selection makes it challenging to get robust predictions across a wide range of scenarios.

8.2 Sensitivity to parameters

The results for the GP-EKF in Chapter 7 heavily depend on the choice of parameters. The basic GP-EKF configuration depends on the initial uncertainty $\boldsymbol{P}_0$, and as it is a pure prediction method, it has a significant impact on the resulting uncertainty of the trajectory. The SL update and the PDAF update also heavily depend on the choice of their respective parameters. Finding a suitable procedure for selecting good parameters for both of these methods is still an open research question.

The problem with both of the update methods is the lack of an appropriate interpretation of the parameters. For the SL approach, it is unknown what the measurement noise $\boldsymbol{R}$ truly should be due to the overly simplistic measurement model, and it is currently tuned by trial and error. Similarly, the clutter rate λ and the detection probability p_D of the PDAF update currently have no good interpretations when used for trajectory prediction.

The GPs also depend on the result of hyperparameter optimization. The MLE approach from Section 3.6.1 works well in most cases, but there are several issues where unrealistic hyperparameters cause trouble. This is especially problematic for the GP-EKF, as the best parameters for the motion model $\vec{f}$ do not necessarily yield the best trajectory prediction, and the Jacobian is sensitive to short length scales. Therefore, it is vital to keep the numerous approximations used by GP-EKF in mind, as they add artificial limitations to the motion model $\vec{f}$.

8.3 Incorporating position in GP-EKF predictions

Considering the statistical results for the PDAF and SL updates in Figure 7.3, there is little evidence to support that these methods for incorporating position data actually have any benefit on average. The problem is that both these methods make assumptions about the target vessel's predicted position, and the historical AIS samples originate from the same underlying trajectory. In practice, the AIS data contains samples from a wide range of different overlapping trajectories. The PDAF is, in theory, supposed to reduce the effect of this issue with the spread-of-innovation term. However, any effort put into finding a good set of parameters has been unsuccessful. While the idea of incorporating data association into the mix is indeed promising, the simple clutter model used by PDAF in this thesis is likely an unrealistic assumption for long-term trajectory prediction.

Consequently, one might question the need for these methods and ask whether the basic GP-EKF performs well enough on its own. However, practical experience with these methods suggests that only using the open-loop GP-EKF prediction often results in underconfident estimates, especially if the true trajectory distribution is multimodal. For example, in branching trajectories, the basic GP-EKF might get stuck between two trajectories, causing significant uncertainty and resulting in a trajectory estimate that follows neither of the branching trajectories. Incorporating position data allows the model to correct for such poor estimates and yields

estimates that are more consistent with the training data. However, the downside is that there is a chance for the proposed trajectory estimate to follow the wrong branch, as can be seen in Figure A.6.

There is also the argument of improving the stability of the predictions. The basic GP-EKF is sensitive to outlier samples, which might significantly impact the predicted trajectory distribution. An update step can correct errors induced by such outlier samples and improve the robustness of the predictions.

Neither the SL update nor the PDAF update is a satisfactory solution to this problem, as both tend to result in overconfident estimates. However, there are clear benefits to incorporating the available position data, which motivates further research.

8.4 Independent Outputs

The formulation of vector-valued GPs in this thesis assumes independent output dimensions. This choice is intended to reduce some of the complexity, as GPs with dependent outputs complicate the derivations, and there is less literature on how to select good kernels when the dimensions are dependent. As a result, the GPs used in this thesis are unable to express any covariance between the outputs.

This assumption is a limiting factor for the direct GP approach as the uncertainty is constrained to specific directions. Without the covariance terms, the model cannot express that the uncertainty is along a given path, such as uncertainty caused by differences in velocity between training samples. A better parametrization may be to instead assume independent lateral and longitudinal components, similar to the formulation used by [15]. Such a decomposition into course and velocity would allow the model more fine-grained control to express uncertainty in specific directions. Another approach could be to relax the assumption of independent output dimensions, though it is unclear how this would be achieved in practice.

For the dynamical GP-EKF approach, the problem of having independent outputs is less problematic. The iterative EKF procedure and propagation of uncertainty allow the model to express covariance in the prediction uncertainty, even if the GP $\vec{f}$ is limited to independent outputs. However, re-parametrization into lateral and longitudinal components could still be beneficial.

8.5 Shared Kernel

A similar simplification is the assumption of having a shared kernel between each output dimension. This choice is also intended to reduce complexity and computational efforts, as optimizing the hyperparameters requires a considerable amount of time. Doubling the time spent optimizing is therefore avoided.

Standardization of the training data allows the use of one shared kernel, even if there are differences in scale for the training data, which is considered flexible

enough in this thesis.

Whether allowing independent kernels will cause the model to perform better is still an open question. However, it would enable the model to learn separate length scales and scale parameters for each output dimension, which could prove beneficial.

8.6 Clustering

Clustering of trajectories has not been prioritized in this thesis. The current implementation of the different methods instead filters the available training data before each prediction, using a set of initial conditions. While this method has worked effectively in this thesis, considerable computational improvements can be made from clustering the trajectories offline and selecting the appropriate cluster when performing predictions. The hyperparameters can then be optimized in advance for each cluster. Combining this work with trajectory clustering methods such as TRACLUS [23] may be a good way to tackle some of the computational challenges of the methods proposed in this thesis.

8.7 Branching Trajectories

The Direct GP approach works well for unimodal trajectory distributions, where there are only minor variations between the trajectories used for training. Due to the assumption about Gaussianity, it cannot express multimodal trajectory distributions. Moreover, the method fails on branching trajectories, as it attempts to use a single Gaussian distribution to describe several modes at once. The result is a prediction mean that lies somewhere in between the branching trajectories and has large uncertainty.

The motion model used by GP-EKF will, in theory, handle branching trajectories better. The vector-field $\vec{f}$ does indeed express multimodal trajectories as seen in Figure A.6, and the trajectory distribution can for instance be found numerically through Sequential Monte-Carlo simulations, as seen in Figure 6.4. However, the GP-EKF approach cannot express multimodal uncertainty, as the state is assumed to be a unimodal Gaussian distribution. This assumption forces the GP-EKF to express the distribution as unimodal. While the GP-EKF artificially limits the flexibility of the motion model, it makes it easier to compare the results of Direct GP and GP-EKF, since both distributions are assumed to be Gaussian. Thus, the GP-EKF has demonstrated the motion model's ability to predict trajectories and has hopefully motivated further research into less constrained ways of simulating the trajectories, such as Sequential Monte-Carlo.

8.8 Training Data

One concern with the GP-EKF is the need for accurate gradients for use in training. Compared to similar approaches in [12, 16], the sampling interval of AIS is in the range of several seconds or even minutes. The finite difference approach for calculating the gradients should, in theory, be a poor choice for curved trajectories, as it would only be able to capture the average velocity over a large sampling interval, but unable to represent the curvature of the trajectory. During testing, several examples where the numerical gradients caused premature turns were found. The gradients were calculated using points before and after a turn, effectively predicting an unrealistic shortcut. Using the reported COG and SOG is therefore expected to perform better, assuming that the vessels report accurate COG and SOG estimates. When comparing the trajectory error of the two possible sources in Figure 7.1, the performance is remarkably similar, though the finitie-difference approach has a slight advantage. However, the COG/SOG approach yields significantly improved NEES, meaning the uncertainty estimate is far more consistent when using COG/SOG as input data. This improvement is hypothesized to be due to the finite-difference approach effectively filtering out some of the inherent variability in speed and course for different vessels because of the long sampling interval. As a result, the vessels in the finite-difference dataset may appear far more similar than in the COG/SOG dataset, causing overconfident predictions.

8.9 GP-EKF time dependency

The time component of the GP-EKF's motion model was initially added to better handle trajectories with sharp turns. For example, if $\vec{f}$ relied on position alone, the GP would have no way of expressing that the speed vector at position $\boldsymbol{x}$ might change over time. The time component was later found to explain some of the variability in the velocity, yielding better uncertainty estimates.

In practice, the MLE approach for hyperparameter optimization will, in many cases, yield very large length scales[1] for the time components of the kernel, effectively disabling the time dependency. The time component is therefore only really used when the dataset contains a lot of time-dependent noise.

However, the added dimension and corresponding hyperparameter also increase the risk of overfitting [11], and whether or not to include the time dependency becomes a trade-off between complexity and performance. For highly smooth trajectories, the necessity of including a time dependency is unlikely.

[1] In fact, the time length scale tends to reach a maximum value during optimization.

8.10 Numerical issues

As with any other data-driven method, the results of using GPs heavily depend on the quality of the data. Outlier samples, such as a vessel suddenly making a u-turn, can cause issues as these samples deviate significantly from the assumed Gaussian distribution.

The problem proves especially problematic for the basic GP-EKF implementation. For example, for GP input values close to one such outlier, the resulting function has a high degree of non-linearity, which conflicts with the Taylor approximation. However, the update steps can, to some extent, alleviate the problem.

While it certainly affects the Direct GP as well, the problem is less severe as this method directly interpolates the position data.

8.11 Computational Complexity

A key concern with using GPs is the computational complexity required to invert the covariance matrix and solve large systems of linear equations. Indeed, the exact formulation of GPs is constrained to small datasets only, and the computations quickly become infeasible as the number of samples increases due to the cubic complexity. Practical experience with GPs indicates that working with several thousand samples works just fine, as long as the hyperparameters are kept fixed. However, it is more problematic for hyperparameter optimization, as the marginal likelihood from Equation (3.24) requires the inversion of the entire covariance matrix at each iteration, which tends to be tediously slow when using more than $N = 1000$ samples.

The hyperparameter optimization turns out to be the main bottleneck of the methods proposed in this thesis. This issue motivates the need for a more clever way to select hyperparameters. For example, a set of global hyperparameters could be used for real-time applications instead of optimizing for each scenario. Furthermore, as there are likely some variations between different vessels and scenarios, clustering could be used as an alternative to preparing a fixed number of hyperparameters to be used in real-time.

Even in an offline setting, hyperparameter optimization quickly becomes infeasible for large datasets. In a real-world application, using approximate methods might be necessary. Variational formulations of the GP, such as the SVGP, could be good a solution. In combination with stochastic optimization and mini-batching, these approximations can utilize large datasets during training. However, these approximate methods also add a substantial amount of complexity, causing the GP to lose its advantage over other methods such as neural networks.

8.12 Using Gaussian Processes for trajectory prediction

Recall the three research questions raised in the introduction.

1. How can GPs be used to model the long-term vessel trajectory of new vessels?
2. Is it computationally feasible to use GPs in the context of AIS Big Data?
3. Is a GP able to provide consistent uncertainty that reflects the true error rate?

The first question has already been extensively covered by Chapter 5 and Chapter 6. As for the second question, there are certainly computational challenges to be aware of, but this thesis has demonstrated that there do exist solutions to these problems and that it is feasible to use GPs in combination with AIS data. The last question is perhaps the most challenging to answer. Accurately representing uncertainty, especially in the presence of complex dynamics, is tricky. There is a wide range of possible ways a prediction might fail, and the potentially non-linear dynamics of vessels make the problem even more challenging. The methods proposed in this thesis still suffer from overconfident predictions, and more research is required on how to perform robust model selection. This being said, the GP framework has certainly shown great promise and flexibility as a method for expressing uncertainty when learning from data.

Chapter 9

Conclusion

The GP framework utilized in this thesis yields a powerful way of expressing beliefs about likely future trajectories. The GP's interpretation as a statistical distribution over functions allows the combination of prior knowledge and data to be used for learning complex relationships while also expressing uncertainty. Furthermore, the Bayesian framework enables these methods to incorporate expert knowledge as prior beliefs and can be considered a mix of model-based and data-driven methods.

Two different formulations using GPs are proposed. The first method directly applies the GP framework to model the trajectory as a function of time. It works well and yields good results during statistical testing. However, this formulation is limited to unimodal trajectory distributions and cannot express multimodal beliefs about branching traffic lanes.

The second method takes an indirect approach and instead uses a GP to learn an unknown motion model expressed as a vector-field. This motion model is flexible enough to express multimodal distributions, though at the cost of higher complexity. This method is then paired with an EKF prediction to simulate trajectories by assuming that the distribution is Gaussian and linearizing the motion model. This GP-EKF approach works reasonably well, but it is sensitive to the choice of parameters and numerical instabilities.

Finally, an alternative solution using Sequential Monte-Carlo is discussed briefly, as the assumptions made by the GP-EKF limit the motion model's ability to express multimodal trajectory distributions.

As the GP-EKF is a purely open-loop prediction based on gradients only, two update procedures are proposed to serve as weak feedback in an attempt to improve the robustness of the GP-EKF. However, both the SL and the PDAF update steps introduced in this thesis cause a significant increase in overconfident predictions and are not considered satisfactory solutions.

The statistical testing showcases the benefit of using GPs over the simpler CVM. For example, on curved trajectories, the GP-based methods yield far better estimates with both lower error and less variability when compared to the CVM. However, for straight-line trajectories, there is still a way to go as the GP-based

methods perform slightly worse than the CVM. The results are also affected by choice of parameters, and they are primarily intended to showcase that using GPs for long-term trajectory prediction is indeed a viable solution.

The implementation used in this thesis still suffers from numerical instabilities and bad local optima during hyperparameter optimization. There is still more research required to understand how model selection can be performed more robustly.

While there still are some quirks, this thesis has hopefully showcased the power of using GPs for long-term trajectory prediction. The GP framework is a compelling method, and the formulations proposed in this thesis are merely examples of how GPs can be utilized. Especially the motion model used by GP-EKF is highly flexible, and the simple EKF prediction scheme used in this thesis artificially limits the true power of this method. Relaxing these assumptions, such as by using Sequential Monte-Carlo, is a promising extension to this work.

9.1 Future Work

There are several possible extensions to the methods proposed in this thesis:

1. Allow independent kernels for each output dimension of the GPs.
2. Include correlation between the outputs.
3. Use GP approximations to utilize more of the available data. Alternatively, consider more clever preprocessing of the dataset, such as clustering.
4. Investigate the effect of different kernel choices and perform more rigorous model selection
5. Use a more informative prior for the motion model. As an example, the CVM could be used as the mean function and yield a constant velocity estimate in areas with low data.
6. Relax the GP-EKF assumptions in order to improve the uncertainty estimates. Using the motion model in a Sequential Monte-Carlo context may yield significantly better estimates, as it avoids linearization as well as the assumption of a Gaussian posterior distribution.
7. Develop a more realistic likelihood for the GP-EKF update procedure. Neither the SL nor the PDAF introduced in this thesis accurately represents the possibility that the vessel is following the wrong trajectory and that the measurements might be all wrong. For the PDAF it might be good to look into a more elaborate clutter model and not assume a fixed clutter rate.

Bibliography

[1] T. Tengesdal, T. Johansen, and E. Brekke, "Risk-based autonomous maritime collision avoidance considering obstacle intentions," *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*, pp. 1–8, 2020.

[2] Y. Kuwata, M. T. Wolf, D. Zarzhitsky, and T. L. Huntsberger, "Safe maritime autonomous navigation with colregs, using velocity obstacles," *IEEE Journal of Oceanic Engineering*, vol. 39, no. 1, pp. 110–119, 2014. DOI: 10.1109/JOE.2013.2254214.

[3] B. R. Dalsnes, "Long-term vessel prediction using ais data," Master's thesis, Norwegian University Of Science and Technology, 2018.

[4] S. Hexeberg, "Ais-based vessel trajectory prediction for asv collision avoidance," Master's thesis, Norwegian University Of Science and Technology (NTNU), 2017.

[5] H. Bayraktar and F. Turalioglu, "A kriging-based approach for locating a sampling site - in the assessment of air quality," *Stochastic Environmental Research and Risk Assessment*, vol. 19, pp. 301–305, Oct. 2005. DOI: 10.1007/s00477-005-0234-8.

[6] L.-F. Cheng, G. Darnell, C. Chivers, M. Draugelis, K. Li, and B. Engelhardt, "Sparse multi-output gaussian processes for medical time series prediction," *BMC Medical Informatics and Decision Making*, vol. 20, Jul. 2020. DOI: 10.1186/s12911-020-1069-4.

[7] E. Siivola, S. Weber, and A. Vehtari, "Qualifying drug dosing regimens in pediatrics using gaussian processes," *Statistics in Medicine*, vol. 40, no. 10, pp. 2355–2372, 2021. DOI: https://doi.org/10.1002/sim.8907. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/sim.8907. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.8907.

[8] G. Cox, G. Kachergis, and R. Shiffrin, "Gaussian process regression for trajectory analysis," Aug. 2012.

[9] E. Brochu, V. M. Cora, and N. de Freitas, *A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning*, 2010. arXiv: 1012.2599 [cs.LG].

[10] d. g. Krige, *A Statistical Approach to Some Mine Valuation and Allied Problems on the Witwatersrand*. publisher not identified, 1951. [Online]. Available: https://books.google.no/books?id=M6jASgAACAAJ.

[11] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005, ISBN: 026218253X.

[12] D. Ellis, E. Sommerlade, and I. Reid, "Modelling pedestrian trajectory patterns with gaussian processes," in *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, 2009, pp. 1229–1234. DOI: 10.1109/ICCVW.2009.5457470.

[13] J. Ko and D. Fox, "Gp-bayesfilters: Bayesian filtering using gaussian process prediction and observation models," *Autonomous Robots*, vol. 27, no. 1, pp. 75–90, 2009.

[14] K. Kowalska and L. Peel, "Maritime anomaly detection using gaussian process active learning," Jan. 2012, pp. 1164–1171, ISBN: 978-1-4673-0417-7.

[15] H. Rong, A. Teixeira, and C. Guedes Soares, "Ship trajectory uncertainty prediction based on a gaussian process model," *Ocean Engineering*, vol. 182, pp. 499–511, 2019, ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2019.04.024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0029801818315427.

[16] S. A. Goli, B. H. Far, and A. O. Fapojuwo, "Vehicle trajectory prediction with gaussian process regression in connected vehicle environment," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 550–555. DOI: 10.1109/IVS.2018.8500614.

[17] K. P. Murphy, *Machine Learning, A Probabilistic Perspective*, T. Dietterich, Ed. The MIT Press, 2012.

[18] A. Girard, C. Rasmussen, J. Candela, and R. Murray-Smith, "Gaussian process priors with uncertain inputs - application to multiple-step ahead time series forecasting," vol. 15, Jan. 2002, pp. 529–536.

[19] B. R. Dalsnes, S. Hexeberg, A. L. Flåten, B.-O. H. Eriksen, and E. F. Brekke, "The neighbor course distribution method with gaussian mixture models for ais-based vessel trajectory prediction," in *2018 21st International Conference on Information Fusion (FUSION)*, 2018, pp. 580–587. DOI: 10.23919/ICIF.2018.8455607.

[20] G. Pallotta, M. Vespe, and K. Bryan, "Traffic knowledge discovery from ais data," Jan. 2013, pp. 1996–2003, ISBN: 978-605-86311-1-3.

[21] F. Mazzarella, M. Vespe, and C. Santamaria, "Sar ship detection and self-reporting data fusion based on traffic knowledge," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 8, pp. 1685–1689, 2015. DOI: 10.1109/LGRS.2015.2419371.

[22] F. Mazzarella, V. Fernandez Arguedas, and M. Vespe, "Knowledge-based vessel position prediction using historical ais data," Oct. 2015. DOI: 10.1109/SDF.2015.7347707.

[23] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory clustering: A partition-and-group framework," in *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '07, Beijing, China: Association for Computing Machinery, 2007, pp. 593–604, ISBN: 9781595936868. DOI: 10.1145/1247480.1247546. [Online]. Available: https://doi.org/10.1145/1247480.1247546.

[24] S. Hexeberg, A. Flåten, B. H. Eriksen, and E. Brekke, "Ais-based vessel trajectory prediction," *2017 20th International Conference on Information Fusion (Fusion)*, pp. 1–8, 2017.

[25] H. S. Mellbye, *Bayesian intent inference in probabilistic graphical models*, 2020.

[26] P. M. Aronow and B. T. Miller, *Foundations of Agnostic Statistics*. Cambridge University Press, 2019. DOI: 10.1017/9781316831762.

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[28] A. Xie, F. Yin, Y. Xu, B. Ai, T. Chen, and S. Cui, "Distributed gaussian processes hyperparameter optimization for big data using proximal admm," *IEEE Signal Processing Letters*, vol. 26, no. 8, pp. 1197–1201, 2019. DOI: 10.1109/LSP.2019.2925532.

[29] Y. Xu, F. Yin, W. Xu, J. Lin, and S. Cui, "Wireless traffic prediction with scalable gaussian process: Framework, algorithms, and verification," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1291–1306, 2019. DOI: 10.1109/JSAC.2019.2904330.

[30] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated, 2014, ISBN: 1461471370.

[31] D. Tran, R. Ranganath, and D. M. Blei, *The variational gaussian process*, 2016. arXiv: 1511.06499 [stat.ML].

[32] M. Titsias, "Variational model selection for sparse gaussian process regression," 2008.

[33] A. G. d. G. Matthews, M. van der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrá, Z. Ghahramani, and J. Hensman, "GPflow: A Gaussian process library using TensorFlow," *Journal of Machine Learning Research*, vol. 18, no. 40, pp. 1–6, Apr. 2017. [Online]. Available: http://jmlr.org/papers/v18/16-537.html.

[34] E. Brekke, *Fundamentals of Sensor Fusion*. 2019.

[35] Wikipedia contributors, *Automatic identification system — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Automatic_identification_system&oldid=1025591994, [Online; accessed 16-June-2021], 2021.

[36] Kartverket, *Kartprojeksjonar*, https://www.kartverket.no/til-lands/posisjon/kartprojeksjonar, [Online; accessed 17-June-2021; Only in Norwegian].

[37] R. P. Jain, E. Brekke, and A. Rasheed, "On particle filters for vessel trajectory prediction using historical ais data," in writing, 2021.

[38] V. M.-H. Ong, D. J. Nott, M.-N. Tran, S. A. Sisson, and C. C. Drovandi, "Likelihood-free inference in high dimensions with synthetic likelihood," *Computational Statistics & Data Analysis*, vol. 128, pp. 271–291, 2018, ISSN: 0167-9473. DOI: \url{https://doi.org/10.1016/j.csda.2018.07.008}. [Online]. Available: %5Curl%7Bhttps://www.sciencedirect.com/science/article/pii/S0167947318301749%7D.

[39] D. T. Frazier, D. J. Nott, C. Drovandi, and R. Kohn, *Bayesian inference using synthetic likelihood: Asymptotics and adjustments*, 2021. arXiv: 1902.04827 [stat.CO].

[40] Y. Bar-Shalom and X. Li, *Multitarget-multisensor Tracking: Principles and Techniques*. Yaakov Bar-Shalom, 1995, ISBN: 9780964831209. [Online]. Available: https://books.google.no/books?id=GfOoMQEACAAJ.

[41] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007. DOI: 10.1109/MCSE.2007.55.

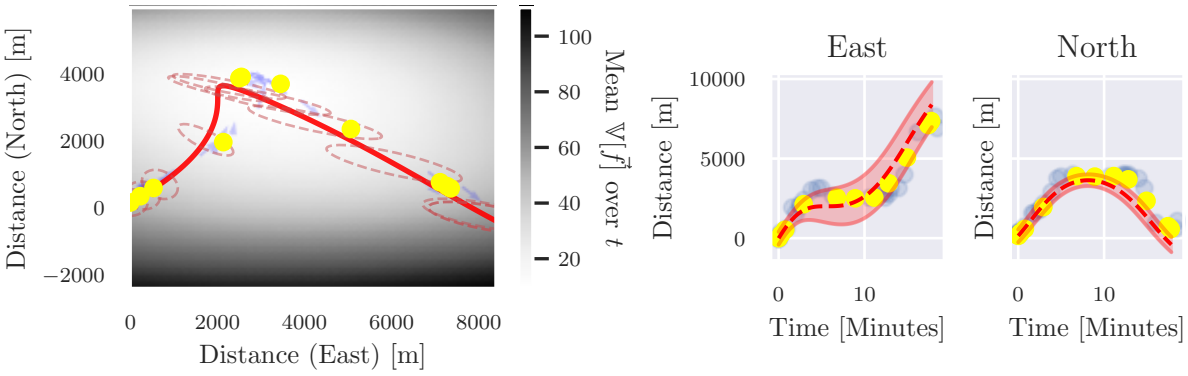
[42] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[43] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. DOI: 10.1038/s41586-020-2649-2. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2.

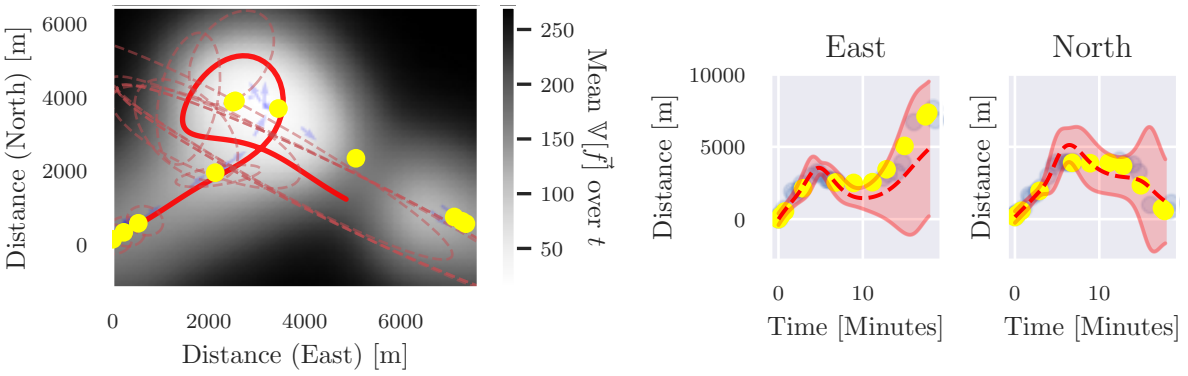
Appendix A

GP-EKF example scenarios

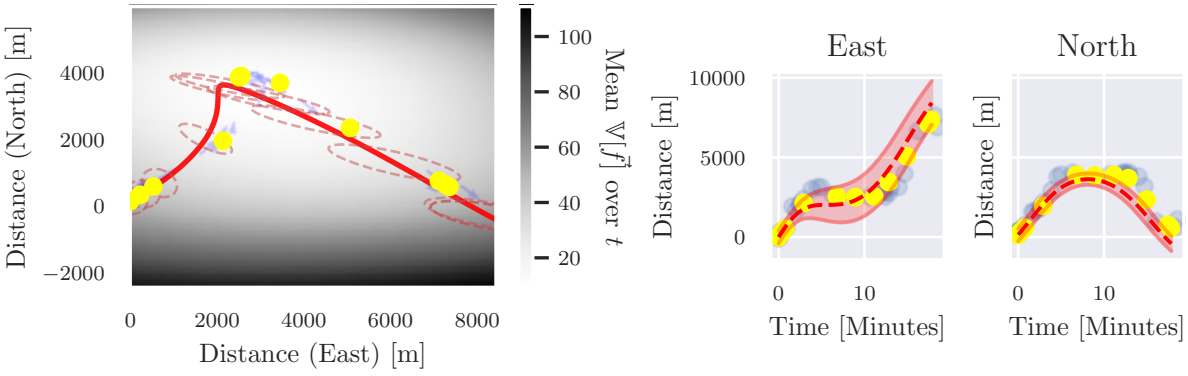
This chapter contains several hand-picked scenarios for the GP-EKF from statistical testing in Chapter 7. The legends and descriptions are omitted from the figures, but all figures follow the same pattern. Red corresponds to the prediction, blue is the training data, and yellow is the ground truth. The ellipses show the 95% CI. Each whole page figure includes the prediction from the basic GP-EKF, GP-EKF with COG data, GP-EKF with PDAF and GP-EKF with SL update.



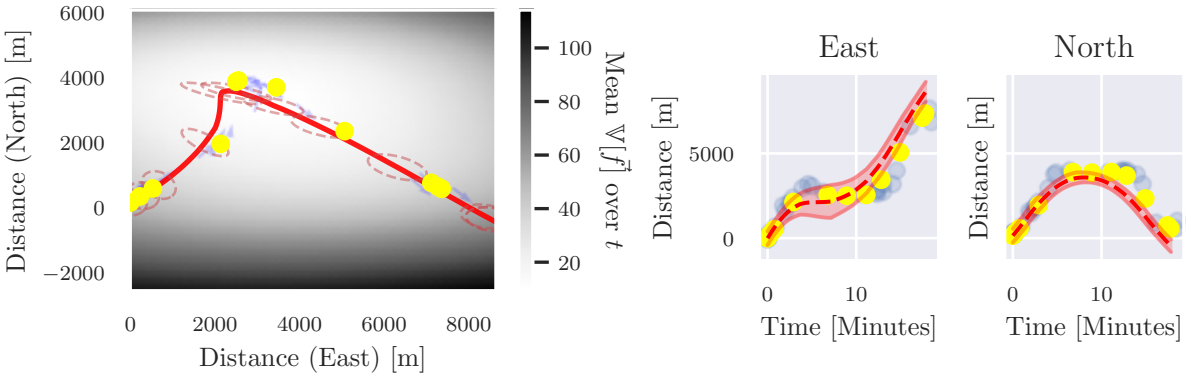
(a) Basic GP-EKF



(b) GP-EKF using COG / SOG data

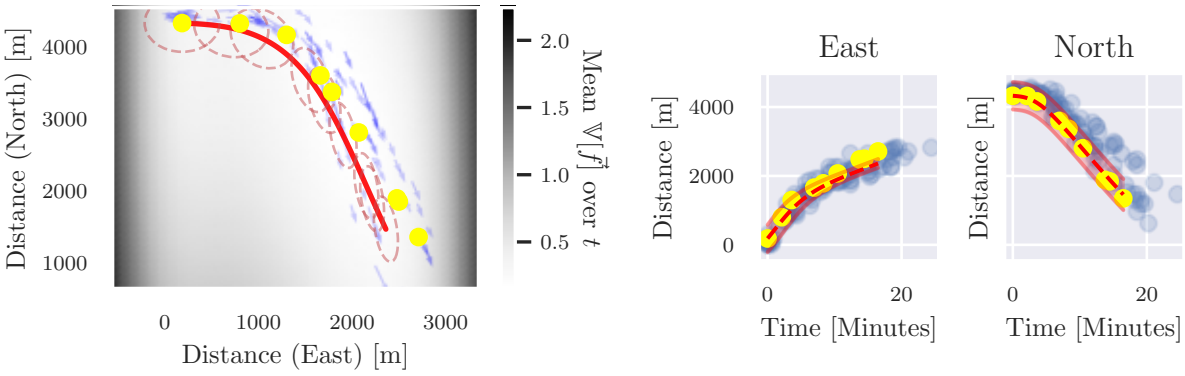


(c) GP-EKF With PDAF update

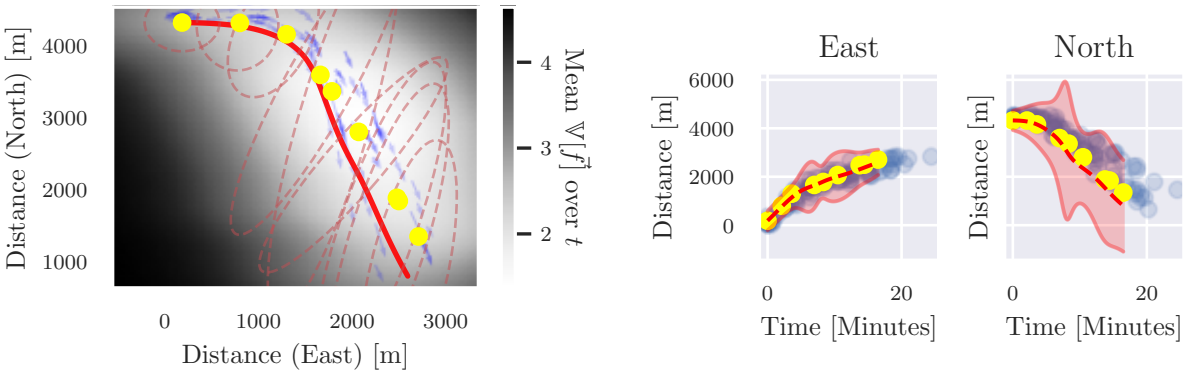


(d) GP-EKF with SL update

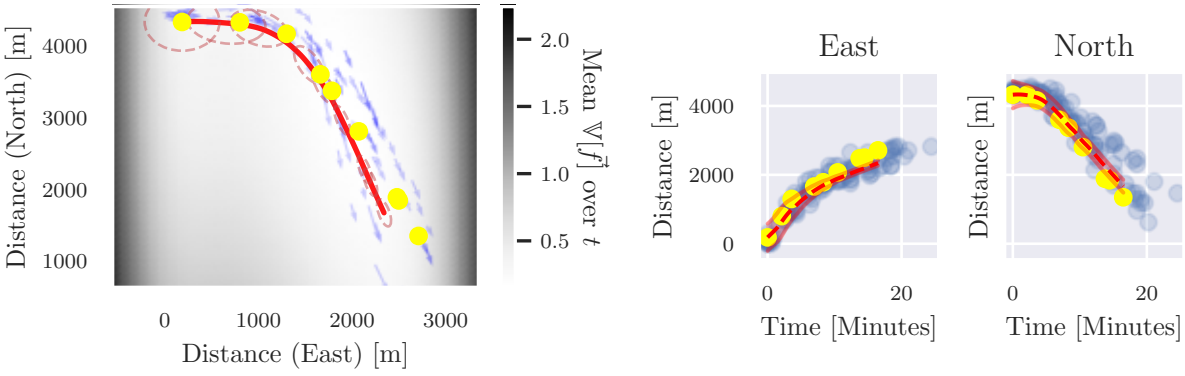
Figure A.1: Case 1



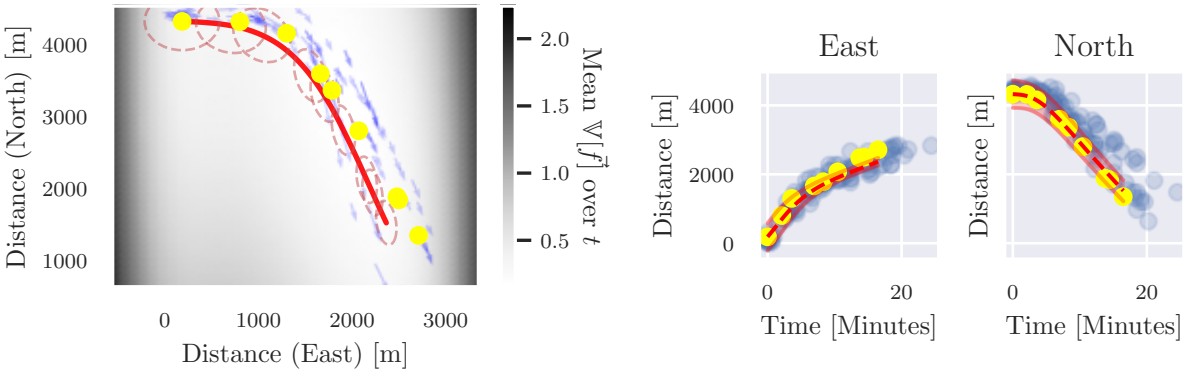
(a) Basic GP-EKF



(b) GP-EKF using COG / SOG data

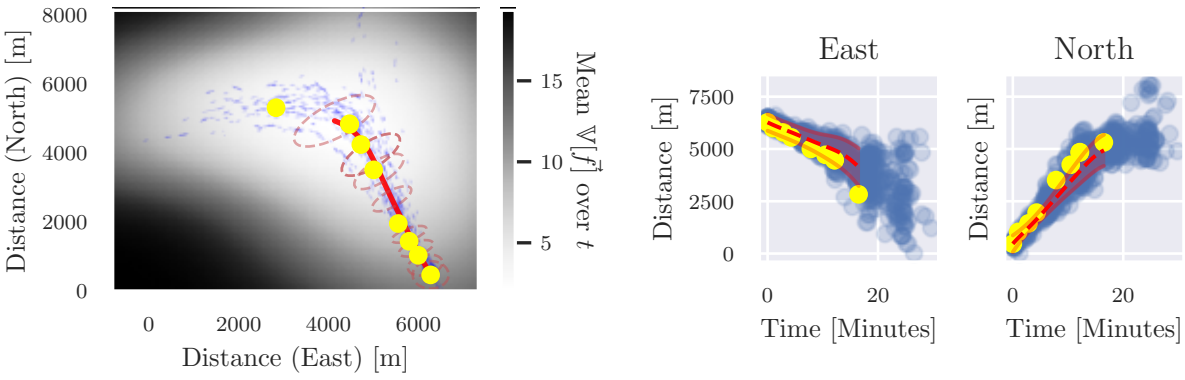


(c) GP-EKF With PDAF update

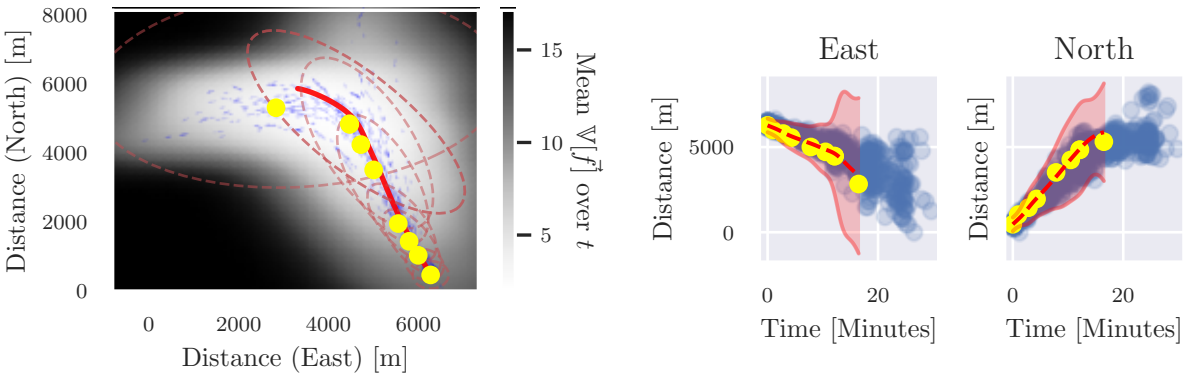


(d) GP-EKF with SL update

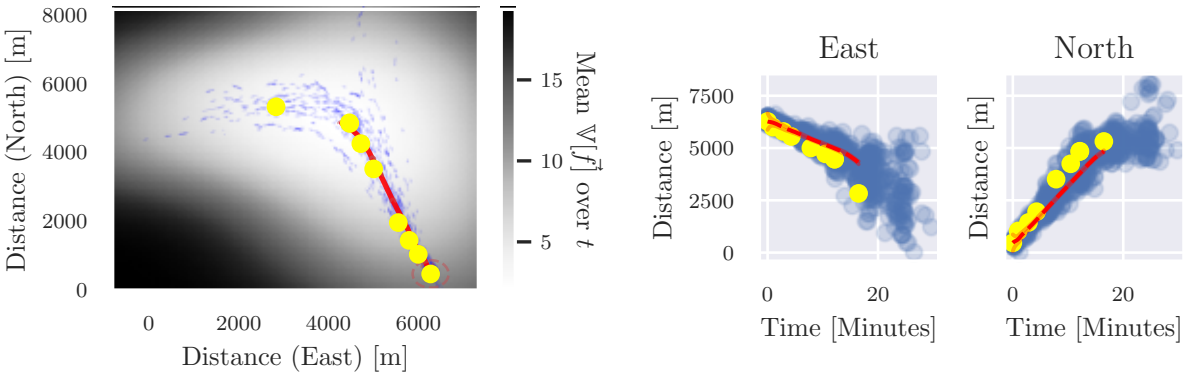
Figure A.2: Case 2



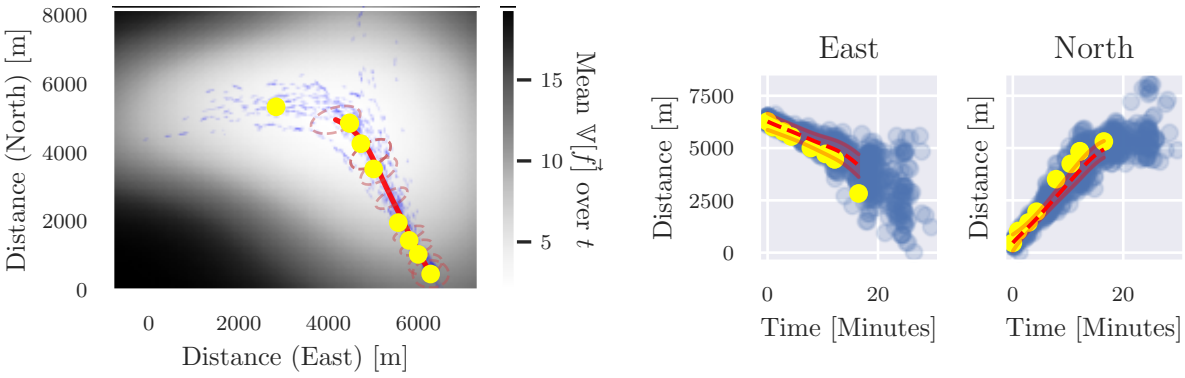
(a) Basic GP-EKF



(b) GP-EKF using COG / SOG data

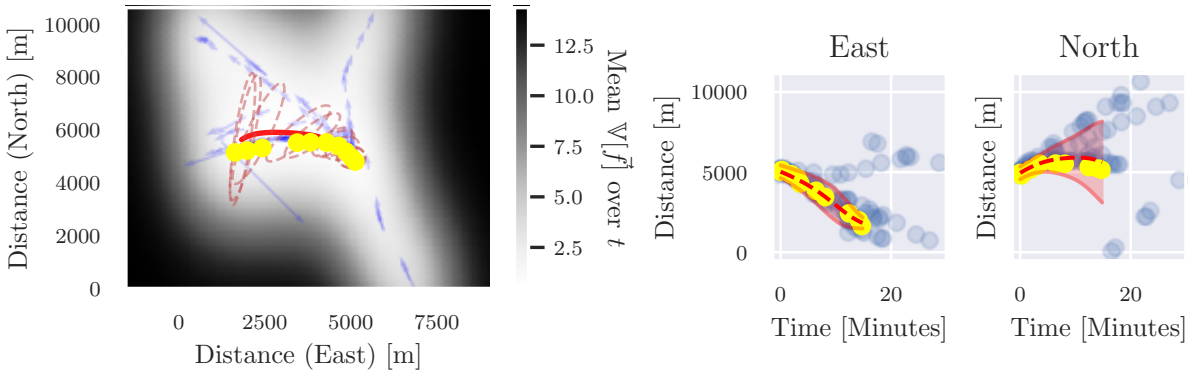


(c) GP-EKF With PDAF update

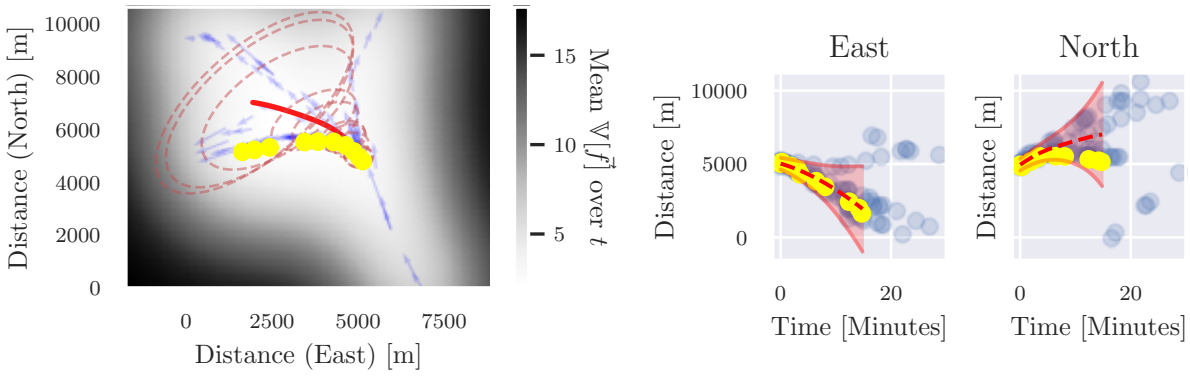


(d) GP-EKF with SL update

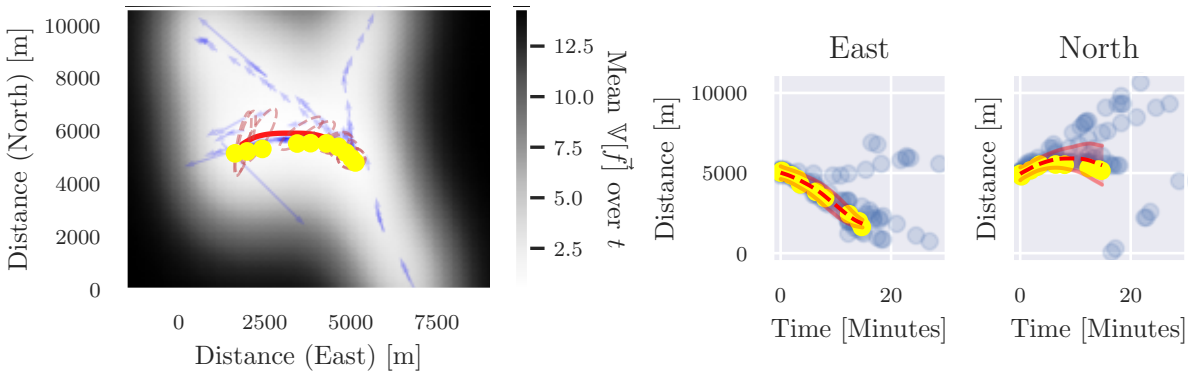
Figure A.3: Case 3



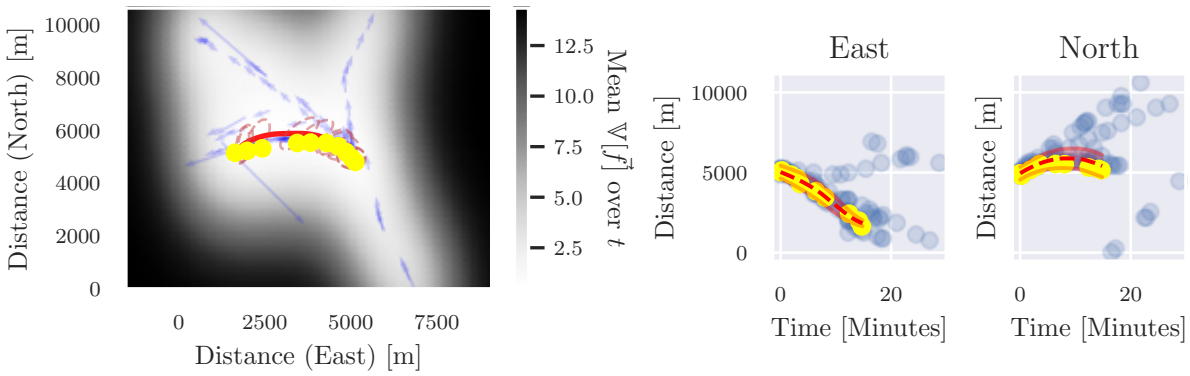
(a) Basic GP-EKF



(b) GP-EKF using COG / SOG data

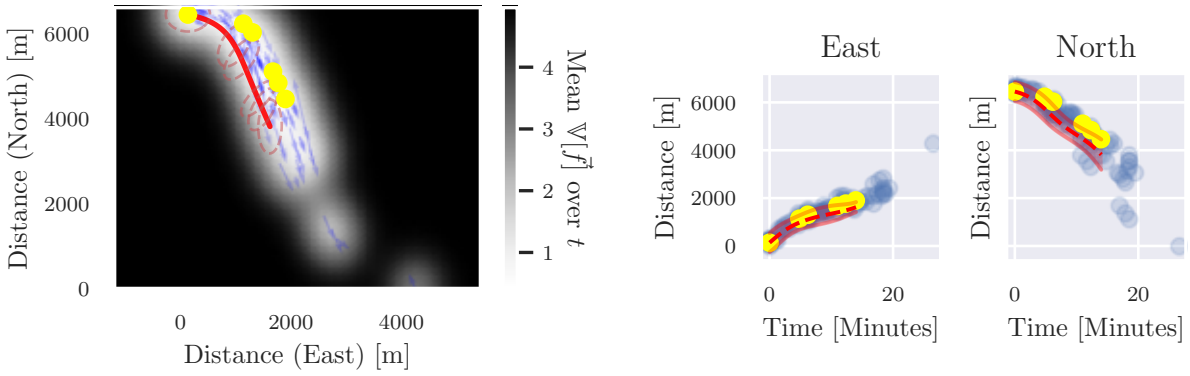


(c) GP-EKF With PDAF update

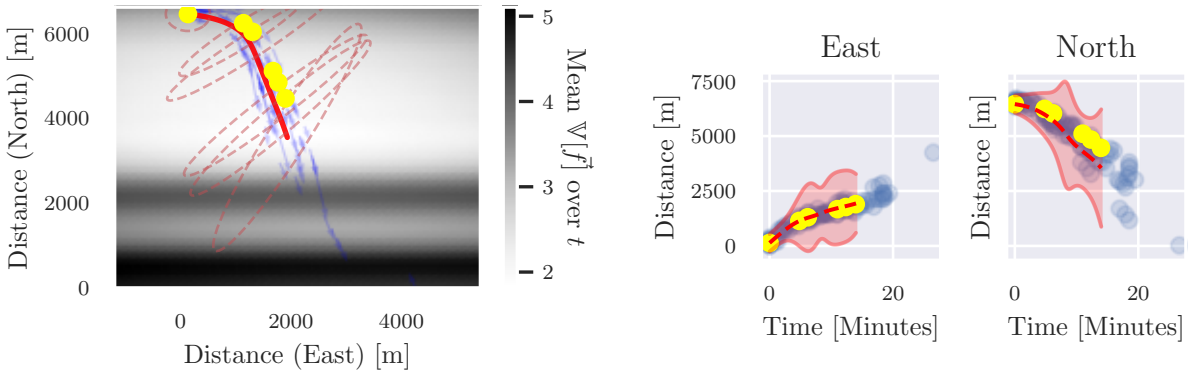


(d) GP-EKF with SL update

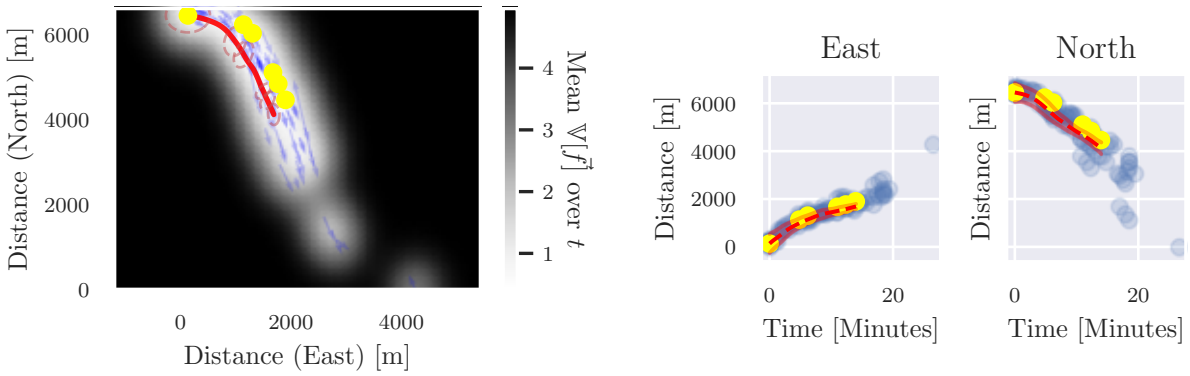
Figure A.4: Case 4



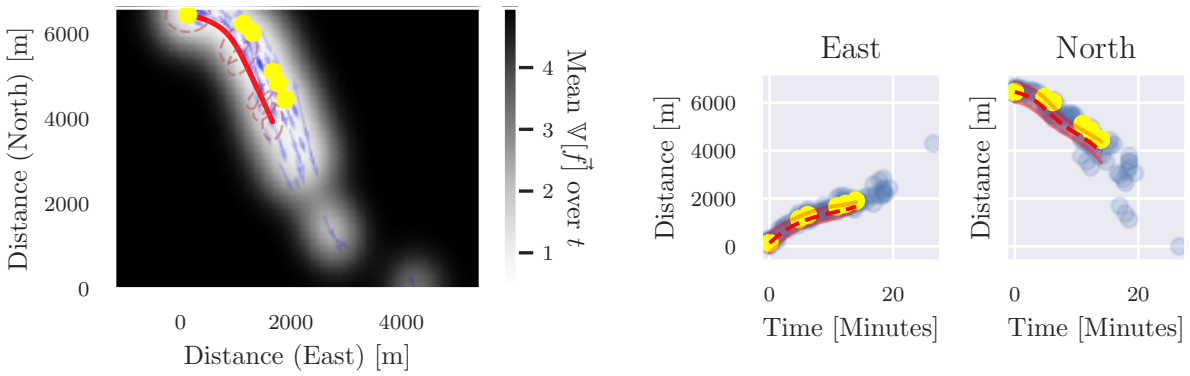
(a) Basic GP-EKF



(b) GP-EKF using COG / SOG data

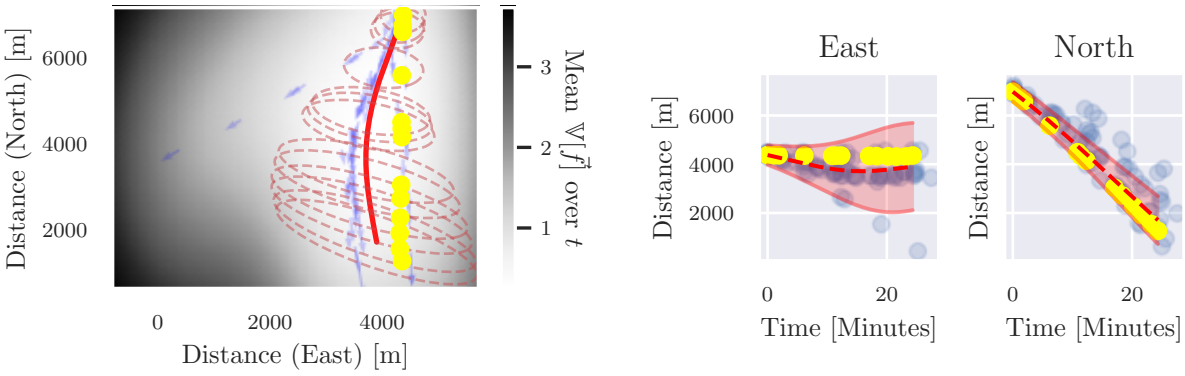


(c) GP-EKF With PDAF update

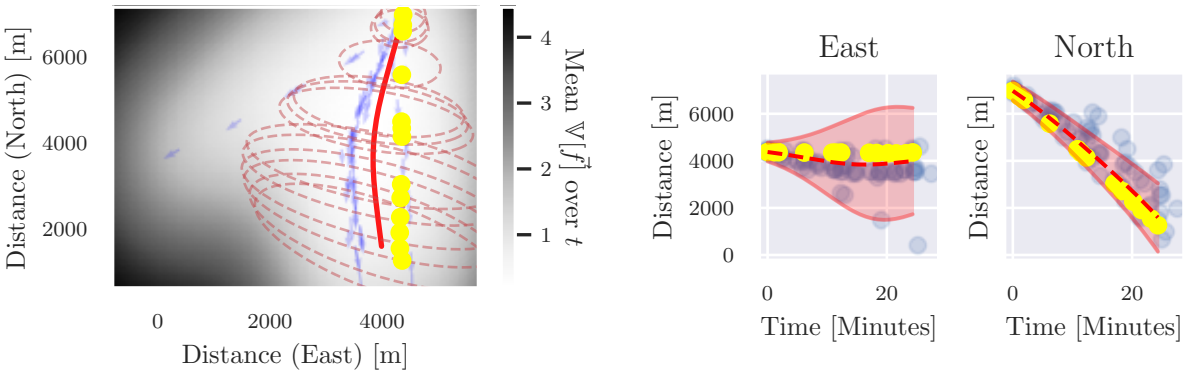


(d) GP-EKF with SL update

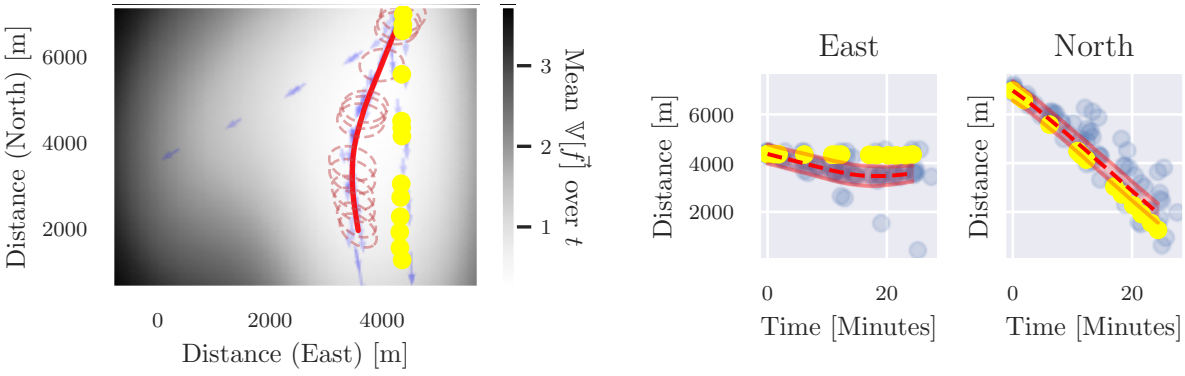
Figure A.5: Case 5



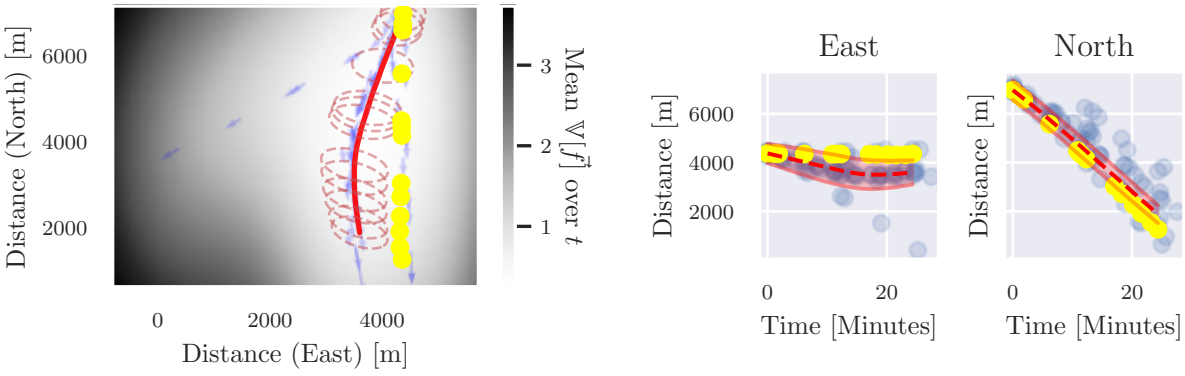
(a) Basic GP-EKF



(b) GP-EKF using COG / SOG data

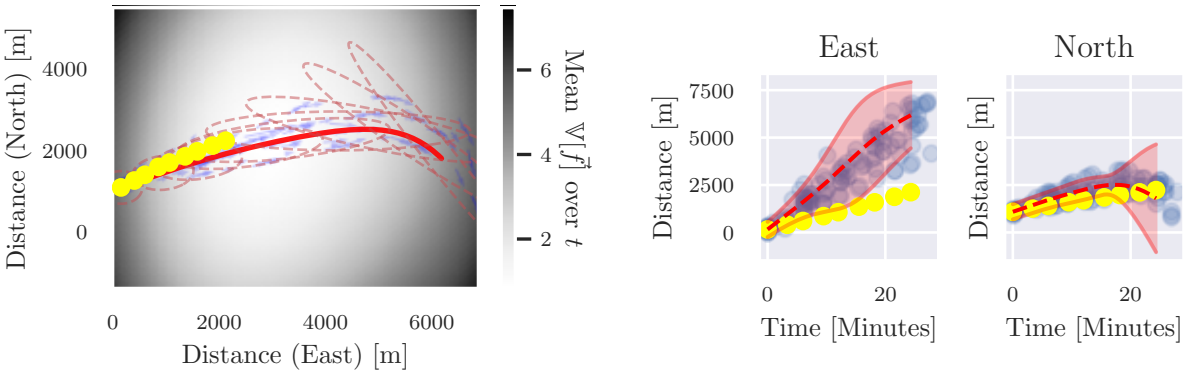


(c) GP-EKF With PDAF update

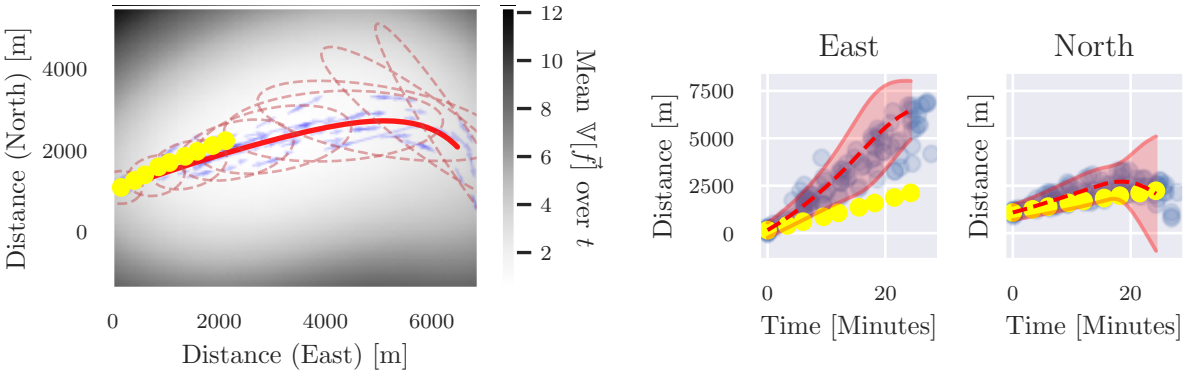


(d) GP-EKF with SL update

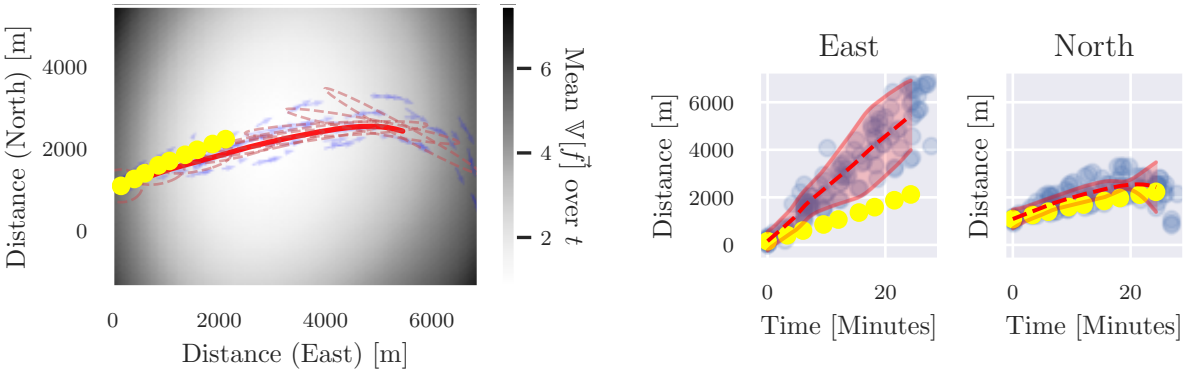
Figure A.6: Case 6: Notice how the PDAF and SL helps the GP-EKF select a single branch, insteading getting stuck in the middle.



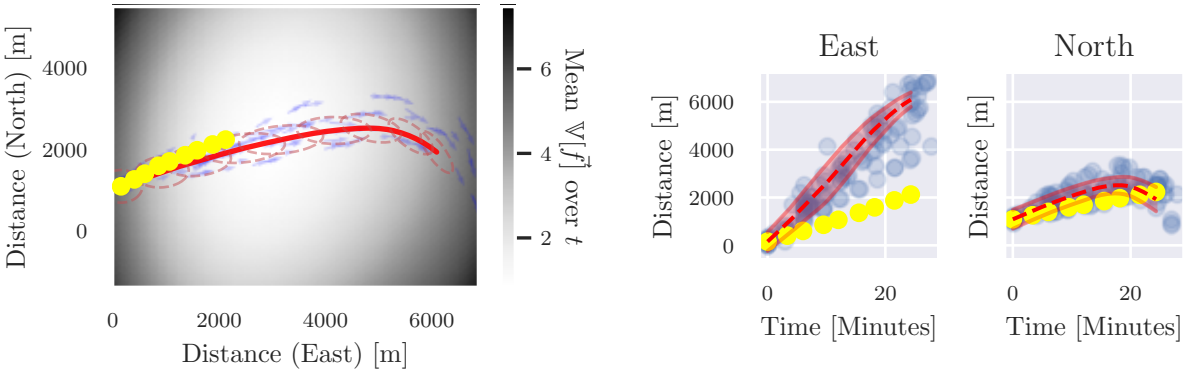
(a) Basic GP-EKF



(b) GP-EKF using COG / SOG data

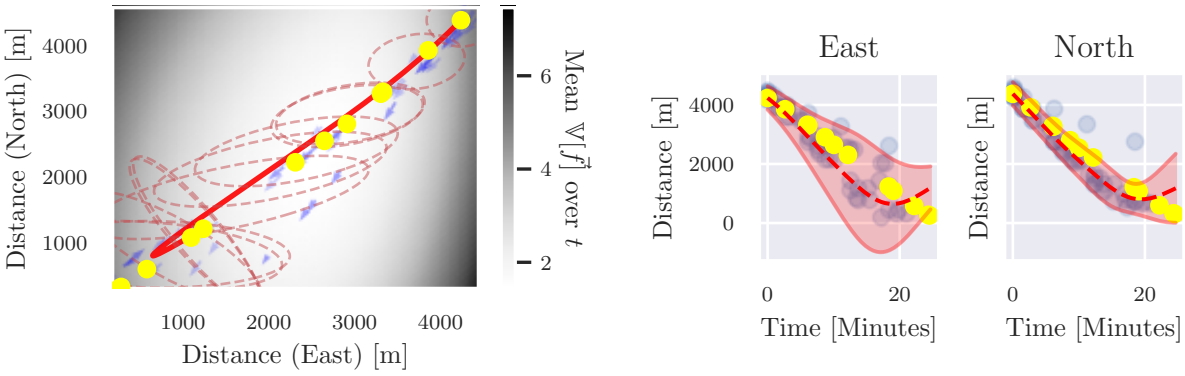


(c) GP-EKF With PDAF update

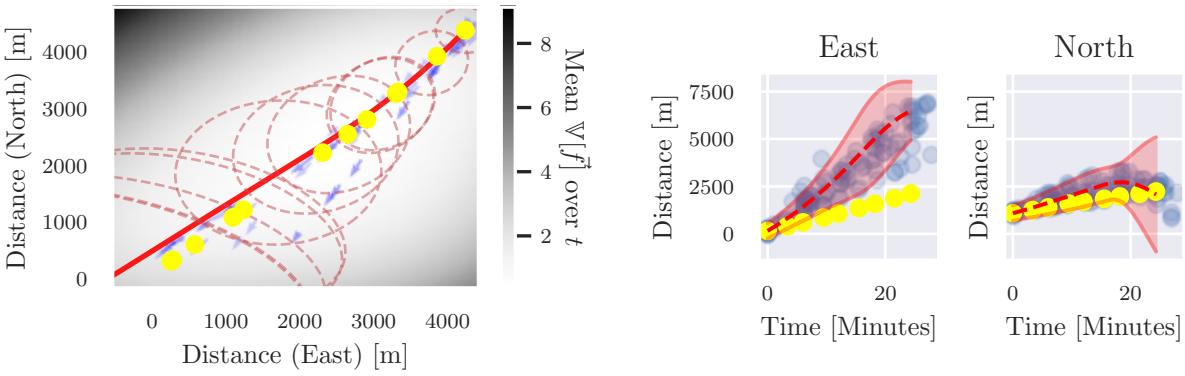


(d) GP-EKF with SL update

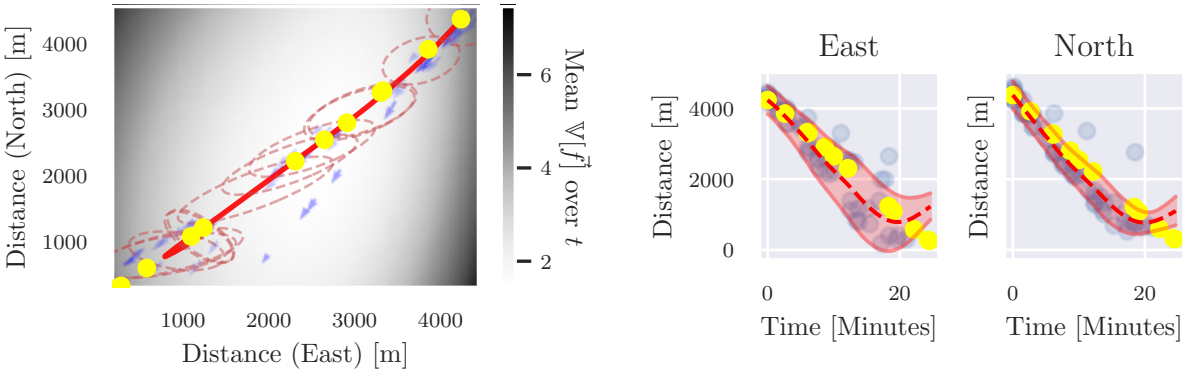
Figure A.7: Case 7: Notice how the training data contains faster vessels, making the GP-EKF overestimate the true velocity.



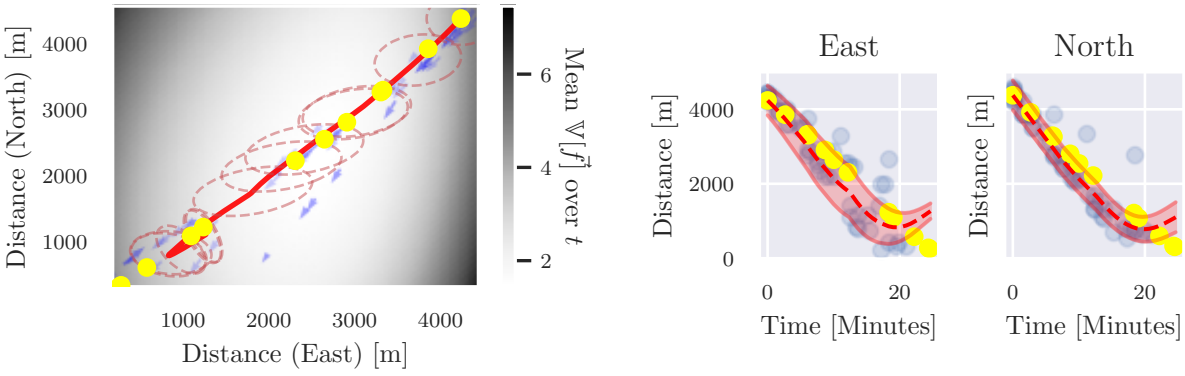
(a) Basic GP-EKF



(b) GP-EKF using COG / SOG data



(c) GP-EKF With PDAF update



(d) GP-EKF with SL update

Figure A.8: Case 8: Notice how the training data contains faster vessels, making the GP-EKF overestimate the true velocity.

Håvard Skåra Mellbye

Gaussian Process AIS Trajectory Prediction

NTNU
Norwegian University of
Science and Technology