Master's thesis

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

Simon Leander Berg Fredriksen

Object tracking and classification using distributed acoustic sensing

Master's thesis in Applied Physics and Mathematics
Supervisor: Jo Eidsvik
February 2024

NTNU
Norwegian University of
Science and Technology

Simon Leander Berg Fredriksen

Object tracking and classification using distributed acoustic sensing

Master's thesis in Applied Physics and Mathematics
Supervisor: Jo Eidsvik
February 2024

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

Preface

This document comprises my Master's thesis in Applied Physics and Mathematics at the Norwegian University of Science and Technology conducted in the autumn of 2023, with Professor Jo Eidsvik from the Department of Mathematical Sciences as supervisor. My specialization in the study is Industrial Mathematics and Applied Statistics, through which I have developed profound knowledge about statistical techniques like linear regression, time series, stochastic processes and Bayesian inference.

The data set used in the thesis is provided by the Center for Geophysical Forecasting at NTNU. The work conducted in this thesis is a continuation of the work done in Fredriksen (2022), in the course *TMA4500 - Industrial mathematics - Specialization Project* completed in the autumn of 2022. The topics and tasks discussed in this thesis are natural extensions of the work in Fredriksen (2022), and was chosen together by Professor Eidsvik and myself.

I would like to express my sincerest gratitude to my supervisor Jo Eidsvik for his exceptional guidance and enthusiasm shown in my research journey. He has been available to answer my many questions, and given good advice as to what topics to include in the thesis. I would also like to thank Kevin Growe from the Department of Electronical Systems for helping me with my questions related to the data set, and The Tien Mai from the Department of Mathematical Sciences for his useful advice and insights. Finally, I would like to thank all my fellow students, friends and family for their continuous support, relevant and irrelevant conversations.

Simon Leander Berg Fredriksen
Trondheim, February 2024

Abstract

Multi-Object Tracking (MOT) comprises a mathematical framework for estimating kinematic properties of objects moving according to a dynamical model in a limited monitored spatial area. Bayesian state estimation, probabilistic measurement models and data association techniques are all central concepts in this framework. In this thesis we discuss the most essential building blocks of MOT, and present the popular MOT-algorithm known as Joint Probabilistic Data Association (JPDA) filter. The main goal of the thesis is to effectively and accurately use JPDA to estimate position and velocity of cars and trains driving along a small road segment on Selsbakkvegen and the railway in Selsbakk, Norway.

Distributed Acoustic Sensing (DAS) is an emerging sensing technology that was originally developed for seismic monitoring, but has over the recent years become a tool in new areas of use. In particular, it has become a sensing technique capable of converting fiber optic cables into distributed and continuous acoustic sensors. We investigate the possibility of using DAS data to extract the positions and acoustic strain induced by cars and trains driving in Selsbakk, and using these as measurements in the JPDA filter to track these objects. In this context, we present a custom signal extraction method that produces processed DAS signal *picks* that are better suited than the original data for direct use in the JPDA filter. This extraction approach is largely based on frequency filters, signal thresholding and clustering methods (DBSCAN). We show how different choices of hyperparameters in the method affects the quality of the signal picks. By using logged timestamps of cars and trains passing Selsbakk station, we systematically optimize these in order to obtain good picks. In addition to this, we present a technique that uses the signal pick strain *amplitudes* for estimating the probability of an object being a car or a train, by the use of Bayes' theorem. The object classifier is developed with the mentioned logged car and train passings as training data, and is built into the JPDA tracker.

We apply the signal extraction method to the DAS data, and use JPDA to track these signal picks, resulting in very promising tracking and classification results. Comparing the estimated tracks to the logged car and train timestamps, the approach successfully identified and tracked 19 out of 20 passing cars and trains. Notably, the performance gets worse when the assumptions of JPDA are not met which occurs, for instance, in the situations of long trains. Furthermore, we provide car and train counts and average velocities throughout an entire day, and results show that cars on average drive with velocity close to the speed limit set for the particular area. Lastly, we discuss the limitations of the proposed tracking method along with possible remedies.

Sammendrag

Multi-objekt målfølging (MOT) omfatter et matematisk rammeverk laget for å estimere kinematiske egenskaper av objekter som beveger seg i henhold til en dynamisk modell i et begrenset overvåket romlig område. Bayesiansk tilstandsestimering, probabilistiske observasjonsmodeller og data-assosieringsteknikker er alle sentrale byggesteiner i dette rammeverket. I denne masteroppgaven gjennomgås de mest grunnleggende konseptene i MOT, og presenterer den populære MOT-algoritmen kjent som Joint Probabilistic Data Association (JPDA) filter. Hovedmålet med masteroppgaven er å effektivt og nøyaktig anvende JPDA til å estimere posisjon og hastighet til biler og tog som kjører langs et segment av Selsbakkvegen og toglinjen på Selsbakk i Norge.

Distributed Acoustic Sensing (DAS) er en fremvoksende sensorteknologi som opprinnelig ble utviklet for seismisk overvåkning, men som gjennom de siste årene har blitt et verktøy i nye bruksområder. Mer konkret er det en sensorteknikk som er i stand til å gjøre om fiberoptiske kabler til distribuerte og sammenhengende akustiske sensorer. Vi undersøker mulighetene for å bruke DAS-data til å finne posisjon og akustisk belastning forårsaket av biler og tog som kjører på Selsbakk, og bruke dette som posisjonsmålinger i JPDA-filteret for å målfølge disse objektene. I den sammenheng presenterer vi en egendefinert signal-ekstraksjonsmetode som produserer prosesserte *signalplukk* som er bedre egnet enn de opprinnelige signalene for direkte bruk i JPDA-filteret. Denne metoden er i stor grad basert på frekvensfiltrering, terskelverdier for signaler og klyngemetoder (DBSCAN). Vi viser hvordan ulike valg av hyperparametere i metoden påvirker kvaliteten på signalplukkene. Ved å bruke loggede tidspunkter til bil og togpasseringer ved Selsbakk stasjon kan vi systematisk optimalisere disse for å oppnå gode plukk. I tillegg presenterer vi en metode som bruker *signalplukkamplitudene* for å estimere sannsynligheten for at et objekt er en bil eller et tog ved hjelp av Bayes' teorem. Objektklassifisereren er utviklet med de nevnte loggede bil- og togpasseringene som treningsdata, og den er innebygd i JPDA-målfølgeren.

Vi anvender signal-ekstraksjonsmetoden på DAS-signalene og bruker JPDA for å målfølge de resulterende signalplukkene, noe som resulterer i veldig lovende målfølgings- of klassifiseringsresultater. Sammenlikner vi de estimerte tilstandene med de loggede bil og tog tidspunktene, så ser vi at fremgangsmåten identifiserte og målfulgte 19 av 20 passerende biler og tog med høy nøyaktighet. Bemerkelsesverdig forverres ytelsen når antakelsene for JPDA ikke er oppfylt, noe som for eksempel skjer i situasjoner med lange tog. Videre gir vi bil- og togantall samt gjennomsnittshastigheter gjennom en hel dag, og resultatene viste at biler gjennomsnittlig følger fartsgrensen satt for området. Til slutt diskuterer vi ulike begrensninger i målfølgingsmetoden med tilhørende mulige løsninger.

Contents

Chapter 1

Introduction

1.1 Background and motivation

In the world of monitoring and surveillance, the demand for advanced sensing technologies capable of providing accurate, real-time information of events in an environment is ever-expanding. This is particularly true in scenarios involving the tracking of multiple dynamic targets in a complex and noisy environment. Traditional sensing methods have often shown shortcomings in meeting these demands, leading to the exploration of alternative solutions. Distributed Acoustic Sensing (DAS) is a sensing technology originally developed for seismic monitoring, but has in recent years evolved into a versatile sensing technique capable of converting fiber optic cables into distributed, continuous sensors. Seeing that this type of sensing technology is able to capture acoustic events along the entire length of the fiber optic cable, it positions itself as a promising sensing tool for Multi-Object Tracking (MOT) applications.

MOT is the science of simultaneously detecting and tracking the movement of multiple objects or targets in a given physical environment over time. The objects can be anything from pedestrians, ships, airplanes and ground vehicles to animals, or any other object that can be observed in a restricted environment. In this thesis we will limit ourselves to cars and trains as objects, driving along a small segment of road and railway at Selsbakk in Norway. According to Bar-Shalom and Li (1995), there exist multiple types of MOT, including

- Point Object Tracking (POT): Each object produces at most one measurement or detection at each time step, and every measurement originate from at most one object, or no object. The physical extent of each object is modeled as a point.
- Extended Object Tracking (EOT): Each object can produce multiple detections, and the extent (shape and size) of the object can be estimated.
- Tracking with multi-path propagation: There are possibly more than one detection per object per time step, because of the multi-path phenomenon.
- Tracking with unresolved measurements: Multiple objects can cause a single detection. This is also called tracking with merged measurements.

The focus in this thesis will be on POT because it is a simpler alternative and we believe it will suffice for our tracking scenarios, all though at the cost of some possibly limiting assumptions on the objects. The primary goal of MOT is the accurate estimation of kinematic properties of the objects, such as position and velocity at any point in a defined period of time. MOT involves many important concepts and building blocks. One of the most important elements is data association (DA), which addresses the challenge of keeping track of which measurements originate from various objects at every time step. Here we run into issues with using the DAS data directly in combination with a POT framework, since the DAS data is often noisy and diffuse, and will work poorly as point measurements of the objects producing these signals. Consequently, it is essential to process the DAS data appropriately with the objective of enhancing the association between measurements and objects.

The fundamental goal of this thesis is to accurately estimate the true position and velocity of cars and trains based on positional measurements of these objects. We aim at demonstrating the potential of DAS data in the context of target tracking, by using processed DAS signals as the measurements in a MOT application. We believe this thesis will shed some light on the potential of DAS and target tracking, in particular offering the following contributions

- Event detection in the DAS data: Identify vehicles and trains moving along the distributed sensor, and extract time and location for the objects causing the acoustic event.
- JPDA tracking using DAS data: Using extracted signals from the DAS data to estimate each object's position and velocity over time.

- Classification of objects detected in the DAS data: Based on signal amplitude caused by the object, estimate the probability of it being a car or a train.

In light of the sustainable development goals formulated by the United Nations (URL: https://fn.no/om-fn/fns-baerekraftsmaal), we believe that the work done and ideas discussed in this thesis could contribute to enhanced traffic and infrastructure monitoring, which is related to sustainability goal number 9. This goal states that we shall build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation. Seeing that DAS is a relatively new technology, and is being used in more and more areas, it certainly contributes to innovation. DAS can be deployed for continuous monitoring of infrastructure such as bridges, pipelines and transportation networks. Since it provides real-time data on the structural health of these installments, DAS can help in early problem detection, enabling timely maintenance and avoiding costly or dangerous infrastructure failures. This provides increased reliability and longevity, and thus resulting in resilient infrastructure. To relate the sustainability goal to the work done in this thesis, we believe that DAS and target tracking can be integrated into smart city initiatives to monitor and manage the flow of traffic and public transportation. This would contribute to the development of Intelligent Transportation Systems (ITS), and could potentially reduce congestion, enhance public safety and consequently reduce emissions related to avoidable traffic problems, thus promoting sustainable urban development.

1.2 Literature review

In this section we discuss and summarize relevant papers and books written on MOT and DAS, and compare them to the work in my thesis. First we review the most established contributions to the field of MOT, and then we present newer papers that combines tracking with the use of DAS data.

POT is a relatively established mathematical framework, and with the development of the Kalman Filter (KF) in the 60s (Kalman, 1960) several new contributions in MOT were made, e.g. Reid (1979). The book by Bar-Shalom and Li (1995) summarized many of these contributions. Here, the Probabilistic Data Association (PDA) filter is considered one of the main methods when tracking one object in a cluttered environment. It also presents the JPDA filter, the multi-object equivalent of the PDA filter. This book is considered one of the most important contributions to the field of MOT, due to its extensive content.

While numerous techniques in MOT have been developed over the last 50 years, the three most widely used approaches in MOT are the JPDA filter, Multi Hypothesis Tracking (MHT) and the Random Finite Sets (RFS) approach (Vo et al., 2015). While JPDA and MHT have been used already for more than two decades, the RFS-based MOT algorithms have received much attention in the last decade. JPDA and MHT are as several other MOT approaches based on DA followed by single-target recursive filtering. As mentioned, DA is the accurate partitioning of measurements into potential tracks and false detections, while filtering is used to estimate the state of each object given their measurement history. The RFS approach however does not focus directly on the DA problem, but it rather seeks both optimal and suboptimal solutions to the multi-object state directly. Vo et al. (2015) gives a recent overview over MOT, and summarizes these three popular MOT approaches.

In recent years there have been significant advances in sensor technology, and it has become increasingly common that objects occupy more than one sensor resolution cell. All though the approaches that are based on the point object assumption (JPDA and MHT) can still be applied to a large variety of scenarios, there is an increasing need for methods that allow for multiple measurements per object per time step. This is the problem that EOT tries to solve. EOT represents one of the latest research areas in target tracking, and its main goal is to estimate the (unknown and time-varying) object shape and its kinematic properties. Granstrom et al. (2016) goes through the latest research in EOT, and compares it to traditional POT. It is also important to mention that the recent advances in Artificial Intelligence (AI) and Deep Learning (DL) has had a huge impact on the field of MOT. With the help of increased computational power by the use of GPUs and clever pattern recognition techniques such as Convolutional Neural Networks (CNN) (see for instance Pal et al. (2021)), object tracking can now be done without the need for the MOT theoretical framework.

When it comes to the use of DAS data for scientific purposes, there are many papers describing how DAS data can be leveraged to achieve a certain goal, see for instance Gorshkov et al. (2022) and Rahman et al. (2024) for detailed reviews on the recent scientific applications with DAS data. In relation with car detection and tracking using DAS, Wiesmeyr et al. (2021) is a paper dedicated to this. Here, a fiber optic cable is placed in the vicinity of a highway, and with vibrations in the ground caused by passing vehicles, the aim is to obtain traffic information like traffic flow and average speed. The DAS signals are preprocessed into binary images by the use of a Fast Fourier Transform (FFT) and empirical signal thresholding. Furthermore, the line detection technique known as the Hough transform (Hough, 1959) was applied to the binary images, and from this vehicle counts and average speeds could be estimated. However, the algorithm was not able to distinguish between different vehicle types, like cars and trucks. Unlike the work done in this thesis, vehicles are not tracked individually over time, but simply detected and counted. The preprocessing of the DAS signals is also quite different, as this thesis uses the log-RMS data, and does not use this Hough transform.

In Wiesmeyr et al. (2020) the issue of tracking trains along a railway is discussed. Unlike in the previously mentioned paper, machine learning techniques are used to process the raw DAS data here. The data is first passed through an FFT, and the transformed signals are summed up individually into 10 frequency bins. Next, a Principal Component Analysis (PCA) is performed to reduce the feature space to 2 dimensions. Then a pre-trained Support Vector Machine (SVM) is used to classify the two-dimensional data points as vibrations or background. The tracking approach is based on several KFs, but it is very different and more involved than the one used in this thesis.

In Thomas et al. (2023) the potential of DAS is discussed in the context of ITS. The physical theory behind the DAS technology is presented, and the data processing methods required to convert the raw data into information that can be used for tracking the position of vehicles along a road stretch is explained. Furthermore, the dependency between defined DAS signal quality metrics and vehicle size, velocity and distance from the fiber optic cable is explored. The paper also considers factors like how the cables are installed and road conditions, and how it affects the quality of the signals. The authors present concrete quantitative results in terms of signal quality for a selection of scenarios. They also explain how high-quality signals can be used in tasks like detecting complex traffic behaviour (queue formation etc.), while lower quality signals are limited to simpler tasks like vehicle counting and average speed computing. The primary technique used for the identification and tracking of vehicles is the Hough transform. It is used to find line segments in the DAS signals which then are grouped together to form vehicle tracks. An interesting result presented in the paper is that DAS systems that operate by measuring phase shifts of backscattered light along the cable are more capable of tracking smaller and slower moving traffic than systems relying on the measurement of intensity variations in the backscatter.

1.3 Outline of the thesis

The thesis is organized into the following chapters:

- Chapter 2 introduces necessary background theory on the DAS system, and gives a specific description of the data set used for vehicle tracking and classification in Chapter 5.
- Chapter 3 gives a detailed description of the procedure for signal extraction, and discusses the importance of producing meaningful signal picks when it comes to achieving high tracking performance.
- Chapter 4 presents the theory behind Bayesian filtering and the KF. Furthermore, it describes the theory behind MOT, addressing its main tasks and challenges. As a feasible solution to the challenges mentioned, the JPDA algorithm is introduced. A simulation example is provided at the end explaining the importance of careful tuning of hyperparameters.
- Chapter 5 is mainly dedicated to showing the results of vehicle tracking and classification, using the DAS data set from Selsbakk in Norway. It starts by discussing issues related to the modeling of the object dynamics and the assumptions made on these in the thesis. Furthermore we provide multiple examples where JPDA tracking is applied to a variety of scenarios. Lastly, we show the traffic trends during the entire day by counting vehicles and displaying velocity patterns.

Chapter 2

The DAS system

2.1 Physical theory and data processing

DAS is an emerging sensing technology that employs laser-based mechanisms to detect acoustic waves in real-time along an optical fiber cable (Taweesintananon et al., 2021). The fiber optic cable is transformed into an array of thousands of virtual microphones that monitors acoustic events along the length of the cable over time almost simultaneously (Liu et al., 2017). This technology has a wide range of applications, including environmental monitoring, oil and gas production monitoring (In't Panhuis et al. (2014); Horst et al. (2015)), earthquake detection and characterization (Hernández et al., 2022), infrastructure health monitoring (Hubbard et al., 2021), whale detection (Rørstadbotnen et al., 2023) and many other application domains.

A single-mode fiber optic cable is installed in the physical area of interest. Most commonly, it forms part of an already existing tele-communication infrastructure, but it could also be deployed for particular DAS purposes. For acoustic monitoring the cable is typically buried underground, but it could also be placed on the surface, in the air or under water. An Interrogation Unit (IU) emits laser pulses at one end of the cable. The frequency by which the pulses are sent depends on the equipment in place, but it is typically in the thousands of pulses per second. All fiber optic cables contain minuscule imperfections which cause some of the light to be scattered back to the IU, a phenomenon known as Rayleigh backscattering (Taweesintananon et al. (2021); Tribaldos et al. (2021)). The scattered light forms an interference pattern, which remains constant in the absence of fiber disturbances. In standard optical communication this is considered a nuisance, but in DAS this backscattering is exploited. Acoustic events, arising from earthquakes, explosions or vehicles, occurring near the fiber optic cable cause small stretches and contractions in the cable, referred to as *fiber strain*. This leads to a phase change in the backscattered light detected by the IU. The IU measures this phase change at equally spaced locations along the fiber, referred to as *channels*.

The DAS system typically consists of three main parts: An IU, the distributed sensor (in our case the fiber optic cable) and a data processing and analysis system. This last main part of the system is a processing unit, that processes and stores the strain data, and provides an interface in which users can interact with the system. Figure 2.1 shows an illustration of the DAS system and its three main components.

2.2 The Trondheim data set

The DAS data studied in this thesis originate from a fiber optic cable located in close proximity to the railway tracks spanning from Marienborg station to Støren station in Trondheim, Norway. The data was collected along a 50-kilometer stretch of this particular railway line. Figure 2.2 shows the map of the road and railway of this 50km

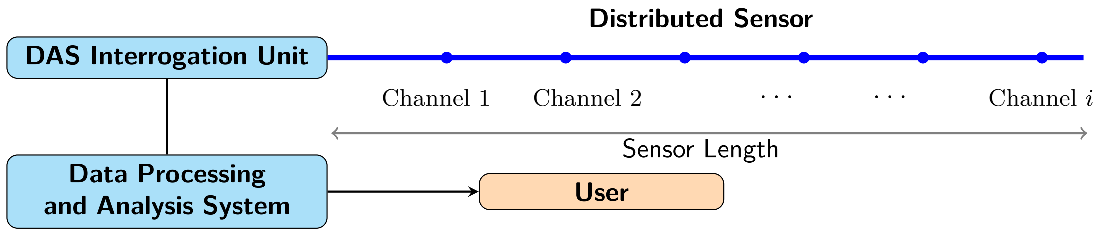


Figure 2.1: The 3 main components of the DAS system: The distributed sensor (fiber-optic cable) is connected to the IU, which in turn is connected to the data processing and analysis system, allowing the user to interact with the system.

stretch, all though we will only focus on a small segment of this stretch, located at Selsbakk which is pointed out on the map. The segment spans from ca. 3960 to 4160 meters from the Marienborg train station, with the Selsbakk station located at 4080 meters from Marienborg. The data was acquired by the use of an OptoDAS interrogator (URL: https://web.asn.com/en/fiber-sensing/main.html) on the 31st of August 2021. The cables used for the data are buried in the ground a few meters away from the railway, and the road segment is parallel to the railway and thus the cable. In the train stations, the cables are encased in concrete armor. The site where the cables are located experiences various sources of signals, including those from passing trains and cars, construction, pedestrian traffic and other unknown sources. Clearly, depending on the source, there will be huge differences in the strain amplitudes registered on the cable at different locations. In fact, these differences are so large that we apply a logarithmic transformation to the data to obtain a reasonable scale for the data, and to be able to compare different sources. Figure 2.3 shows a *heatmap* of the logarithm of the Root Mean Square (RMS) strain values over the entire 50 km stretch from Marienborg to Støren. Here, the strain values are colorized according to their amplitude, with the x-axis representing distance from Marienborg station, and the y-axis representing time. Selsbakk station is pointed out on the stretch by the red cross on the left side. We can observe several types of events in this heatmap:

- Cars driving: They appear as green, thin lines moving both away from (south) and towards (north) Marienborg station. Especially around 35-45 kms from Marienborg we can see car traffic from E6.
- Trains driving: They appear as yellow, thicker lines driving in both directions. The thinner yellow lines are shorter passenger trains, while the thicker yellow lines correspond to longer freight trains.
- Stationary noise: This might be due to activity such as construction.

Instead of using the raw strain data, which contains both positive and negative strain values corresponding to compression and extension respectively, the RMS of this data is computed, resulting in a smoothed version of the absolute raw data. The procedure is simple: Suppose we have a time series $(x_1, x_2, ..., x_N)$ of fixed length, corresponding to a desired time interval. Take a window with size N_w (typically 100-200 ms), and compute the RMS value

$$RMS = \sqrt{\frac{1}{N_w}\sum_{i=1}^{N_w} x_i^2}, \tag{2.1}$$

for the data points falling inside the window. The RMS value is saved at the point corresponding to the center of the window. Then the window is shifted some number of points, the RMS is computed again, and saved again to the center point. This is done for every channel. RMS data has only positive values as it is squared and is the mean of the amplitudes for a window of size N_w. Using the RMS data is useful since it gives us information about the average amplitude of the DAS signals over time.

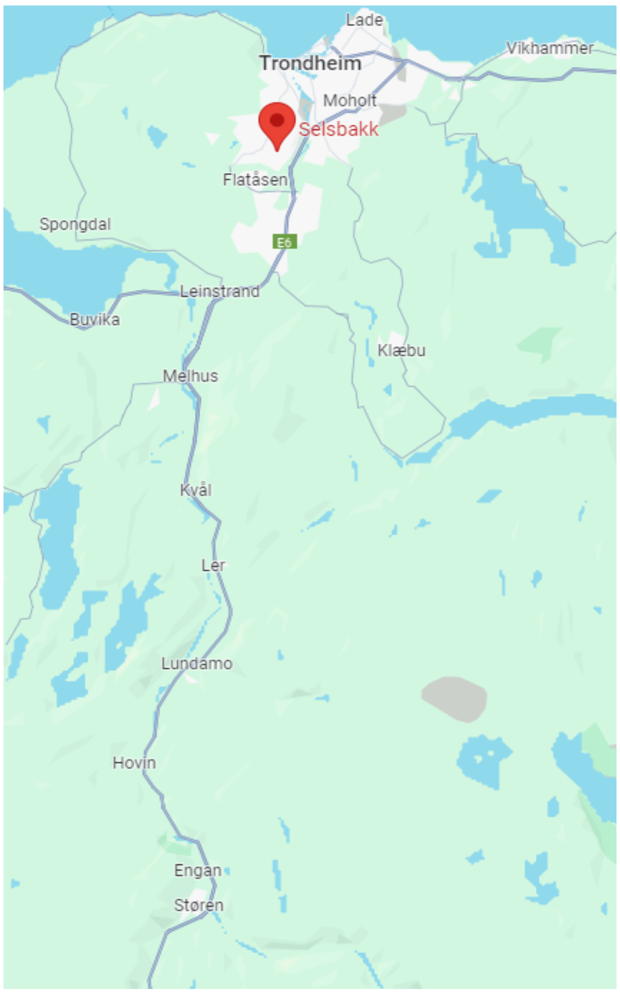
Taweesintananon et al. (2021) gives a thorough technical description of the DAS setup in the experiment conducted in that paper, and a similar setup is used for the production of our data. The light pulses were repeatedly sent with a free-space wavelength of $\lambda_o = 1550$ nm, where the sampling period at the optical receiver ($\Delta\tau$) was 10^{-8} s. The spatial sampling interval (SSI) is given by the sampling period as

$$SSI = \left(\frac{c}{2n_g}\right)\Delta\tau, \tag{2.2}$$

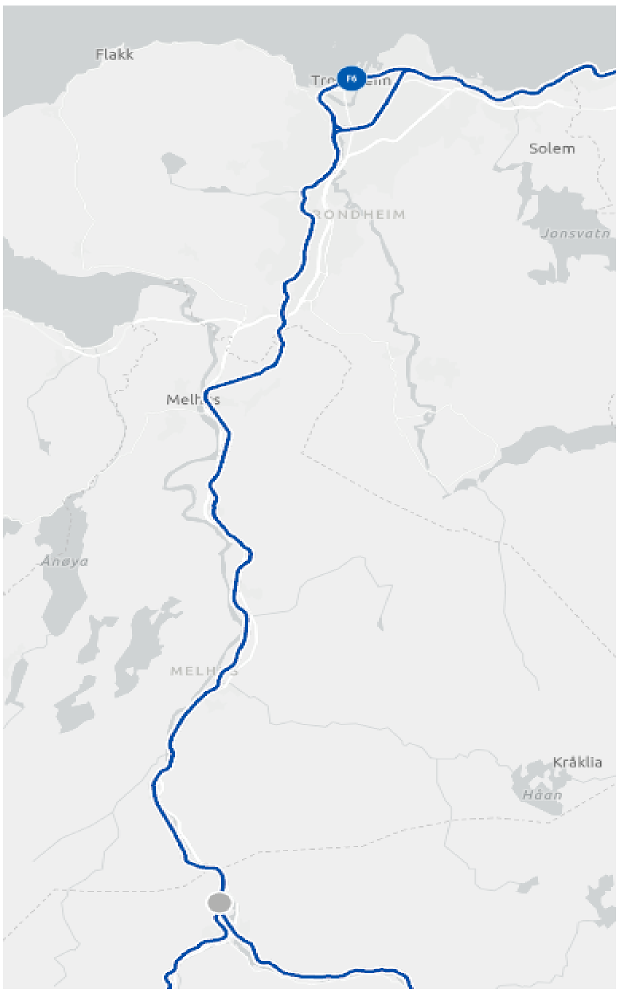
where $n_g \approx 1.47$ is the refractive group index, which contains some information about the properties of the fiber, and $c \approx 3.0 \cdot 10^8$ m/s is the speed of light in vacuum. This results in $SSI \approx 1.02$m, which is the minimum physically possible spatial sampling interval for this cable. Furthermore, let Φ_x be the phase of the backscattered light from the spatial sampling location x, which can be written in radians as

$$\Phi_x = \frac{4\pi n_g x}{\lambda_o}. \tag{2.3}$$

The IU extracts the time-differentiated phase $\dot{\Phi}_x$ at the location x. Since the length of our cable is approximately 50 km, the time-sampling interval dt was originally set 0.5 ms. This is a relatively low value, but it was to ensure sufficient sampling if the frequency of cars and trains was very high. When it comes to spatial sampling size, the SSI can be adjusted to the desired spatial resolution by applying a moving average to $\dot{\Phi}_x$ around x. In our case, the resolution was determined by averaging across 4 spatial samples, leading to a width $L_W = 4 \cdot SSI = 4.08$m between adjacent channels. This is done to improve the signal-to-noise ratio (SNR) and reduce the computational load, compared to using the SSI directly. That is, L_W is the number by which we multiply the number of channels we are from Marienborg station to know how many meters away we are from it. For a more detailed derivation, we refer the reader to Taweesintananon et al. (2021). Since we are using the log-RMS data and not the raw data, the temporal



(a)



(b)

Figure 2.2: **Left:** Map of the 50km road stretch from Trondheim to Støren, with Selsbakk pointed out. The map is retrieved from Google Maps (URL: https://www.google.com/maps/@63.3420978,10.403653,11.31z?entry=ttu) **Right:** The 50km railway from Trondheim to Støren. Retrieved from Bane NOR (URL: https://togkart.banenor.no/)

sampling interval can safely be increased. In our case, we use $dt_{RMS} = 0.2$s. This is beneficial, since we drastically reduce the amount of data and thus computational load, without loosing significant information in the data.

With the help from Center for Geophysical Forecasting (CGF)(URL: www.ntnu.no/cgf), we were provided access to a list of logged events for a time interval on 31st of August of 2021. The events logged whether a car or a train passed the Selsbakk station, and the time of the event. In the case of multiple cars passing with very short time breaks, the number of cars was also logged with the same timestamp. Figure 2.4 shows a map of this road segment and the railway at Selsbakk. Furthermore, Figure 2.5 shows a zoomed in version of Figure 2.3, namely a heatmap of the log-RMS strain values for the road segment in focus with Selsbakk station in the center, and the logged events as red (train passings) and blue (car passings) dots on top of the heatmap. The events appear to align nicely with the increase in strain value as a consequence of the presence of a vehicle or train. The full event log consisting of 29 events is reported in Table 2.1.

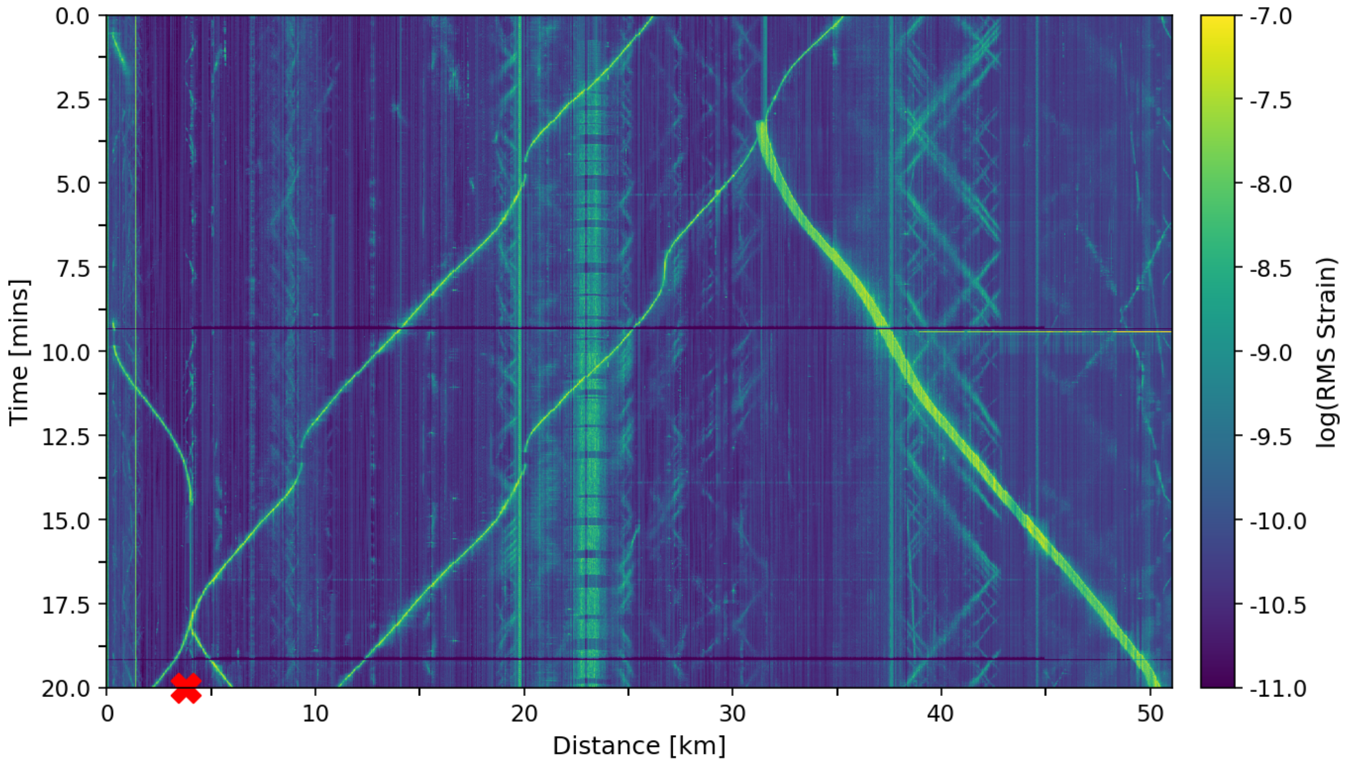


Figure 2.3: Heatmap of the log-RMS fiber strain over the entire 50 km stretch from Marienborg to Støren. The red cross represents Selsbakk station.

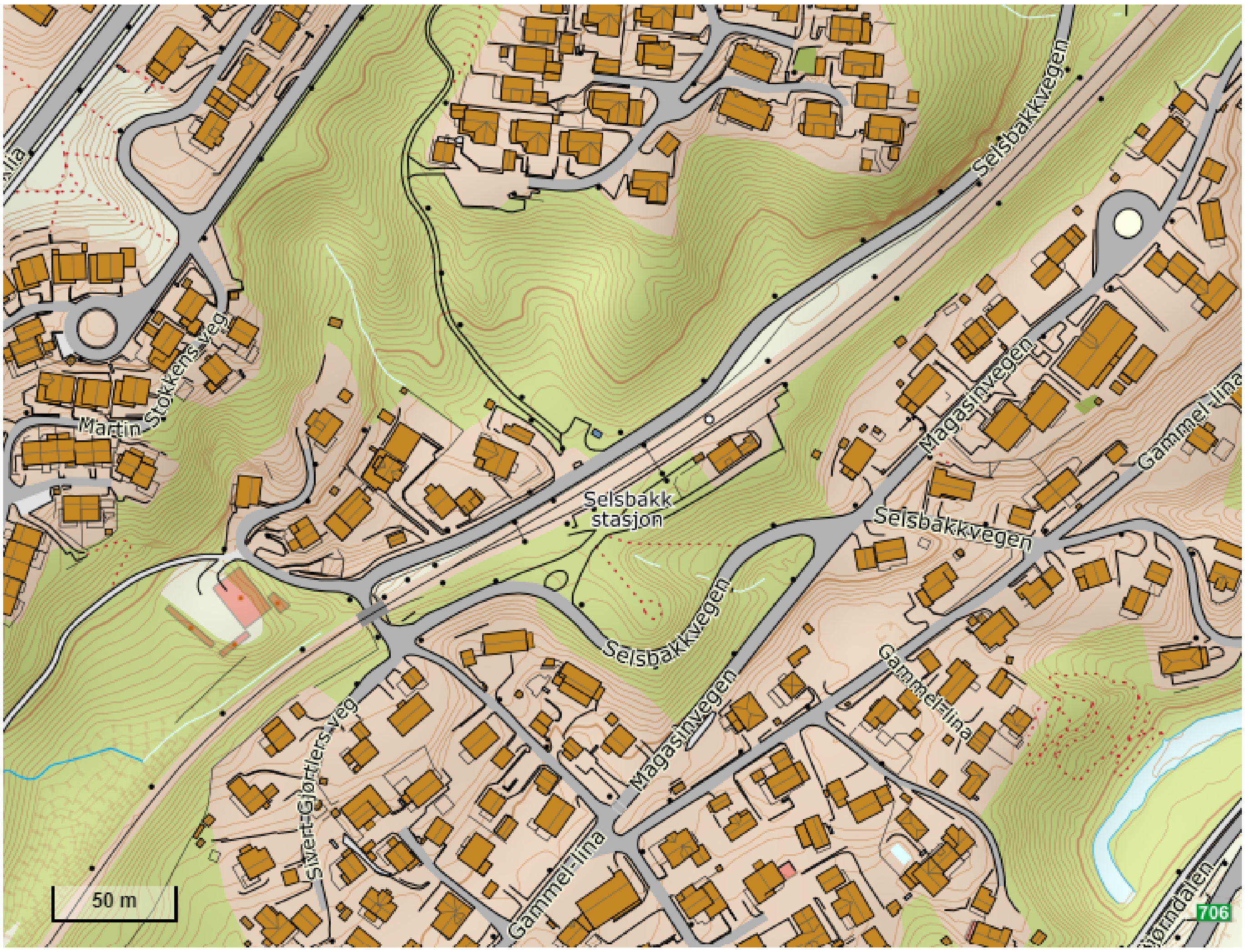


Figure 2.4: Map of the road and railway segment in focus at Selsbakk, Norway. Retrieved from Norgeskart (URL: https://www.norgeskart.no/#!?project=Fastmerker&layers=1002&zoom=16&lat=7037298.69&lon=268870.93&p=searchOptionsPanel&markerLat=7037430.3177006245&markerLon=268383.61303611647&sok=selsbak)

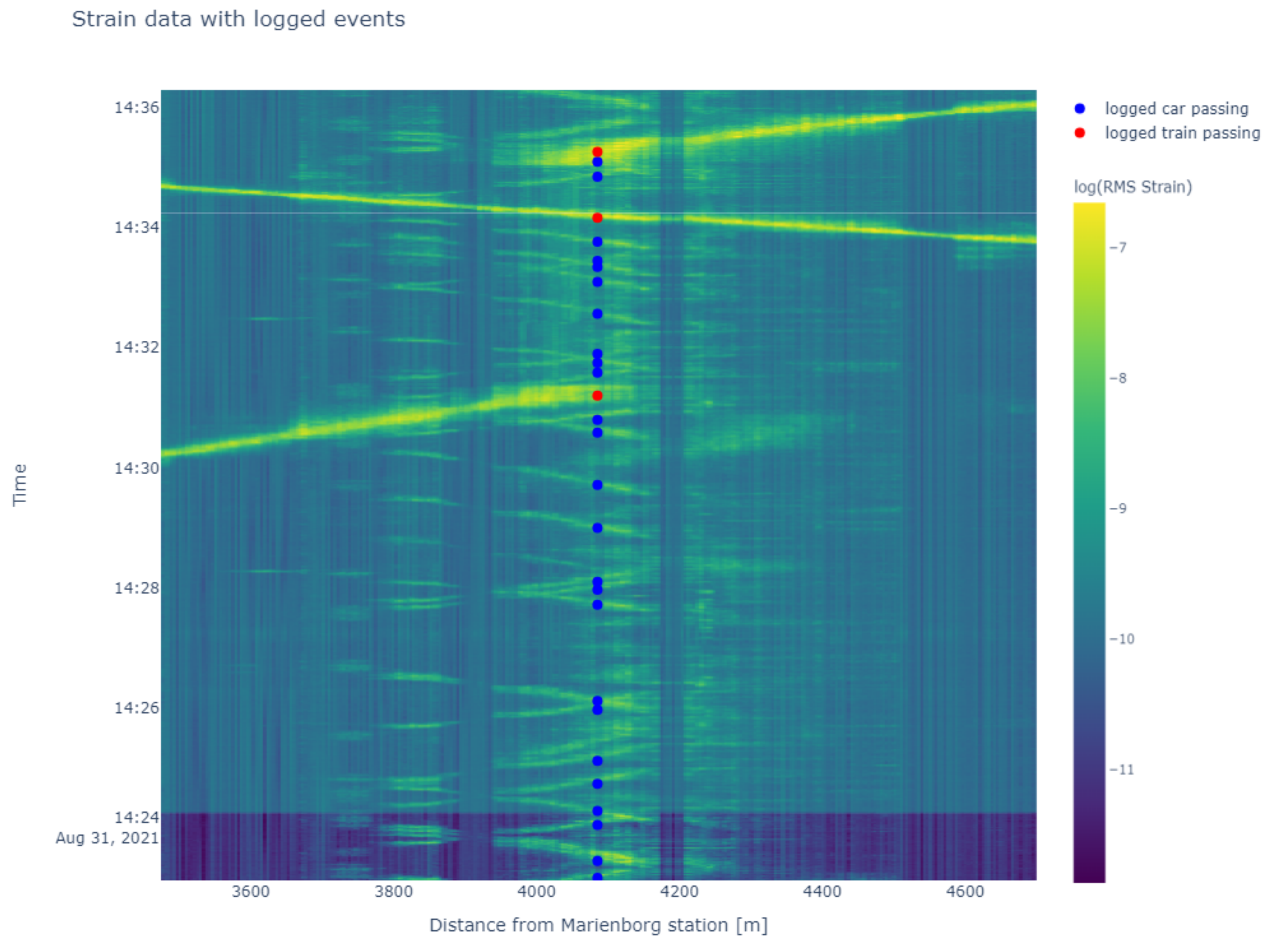


Figure 2.5: Heatmap of log-RMS strain values from ca. 3400 to ca. 4700 meters from Marienborg train station for ca. 12 minutes from 14:24 the 31st of August 2021. The red dots represent logged train passings, and the blue dots represent logged car passings at Selsbakk station.

Time	**Data**
16:22:45	3 cars
16:23:10	1 car
16:23:27	3 cars
16:24:03	1 car
16:24:17	1 car
16:24:44	1 car
16:25:07	1 car
16:25:58	1 car
16:26:07	2 cars
16:27:43	1 car
16:27:58	1 car
16:28:06	1 car
16:29:00	1 car
16:29:43	1 car
16:30:35	2 cars
16:30:48	1 car
16:31:12	1 train
16:31:35	1 car
16:31:45	1 car
16:31:54	1 car
16:32:34	1 car
16:33:06	1 car
16:33:21	1 car
16:33:27	1 car
16:33:46	1 car
16:34:10	1 train
16:34:51	1 car
16:35:06	2 cars
16:35:16	1 train

Table 2.1: Event log from Selsbakk station the 31st of august, 2021 (GMT+1 time).

Chapter 3

Signal extraction

In this chapter we describe the process used for extracting meaningful and accurate signal representations from the DAS data. These data points are going to serve as measurements in the later presented MOT framework. The DAS data set provides log-RMS strain data at each channel for a given time interval. This can be viewed as a high-dimensional time series with the number of dimensions equal to the number of channels. Time series often contain a certain level of noise, and this is certainly true in our case. The sources from which the noise originate are many and can be very complex. Some examples include environmental causes (temperature fluctuations, humidity change etc.), sensor limitations and instrumentation errors. Noise can also be introduced during signal processing and data acquisition. Furthermore, acoustic events such as vehicles driving on a road next to a fiber optic cable cause wave-like disturbances in the fiber. This is due to the nature of sound waves, which propagate as pressure waves in the surrounding medium. Consequently the strain variations can be detected across multiple consecutive channels and at multiple consecutive timestamps. This leads to a number of challenges:

- Issues with spatial and temporal localization: The RMS data does not provide precise information about the object's exact location at each timestamp. Without a systematic approach to extract representative locations, the identification of the event or the object becomes ambiguous. The fact that we have multiple strain values for the same event at the same time introduces uncertainty and makes it complicated to pinpoint the exact location of the event, at least in the context of POT.
- Issues with DA: Occasionally objects move close to each other in time and space. A tracking algorithm will have trouble associating the right measurement to the right object, if multiple measurements originate from each object at a given timestamp. If the objects are close enough, the measurements might even overlap each other. This violates the fundamental assumption in POT, which is that each object can generate at most one measurement per time step.
- Increased computational load: As will be discussed later, the computational load increases drastically with the number of measurements used in a MOT algorithm (Bar-Shalom & Li, 1995). Without a systematic approach to extract representative measurements in place, this can lead to performance bottlenecks in the tracking application. This is especially true in real-time tracking applications, where computational efficiency is crucial for accurate and efficient tracking.

As an attempt to reduce the noise in the time series data, we apply a frequency filter. All though it removes some noise, the resulting data is not processed enough for direct use as measurements in a tracking algorithm. Therefore, it is necessary to implement a method for extracting at most one representative value of the signal at a specific channel at a specific time, called a signal pick. This must be done while minimizing the likelihood of extracting noise as a signal pick. In other words, we need a method that

- Given high-amplitude signals across multiple consecutive channels that likely originate from the same target, finds a single, representative signal.
- Given perpetuated high-amplitude signals across multiple consecutive timestamps from the same channel that likely originate from the same target, finds a single, appropriate timestamp for the event.
- Effectively rules out signals that likely originate from noise.

Figure 3.1 shows how the strain data evolves over time for one channel at the Selsbakk station. It is easy to recognize the vehicles and trains passing this checkpoint by the increase in strain amplitude. However, none of these are unimodal signals, and therefore the signals must be grouped together somehow. Figure 3.2 shows a similar situation but across the entire railway segment, for a single timestamp. We see that there is activity around Selsbakk station

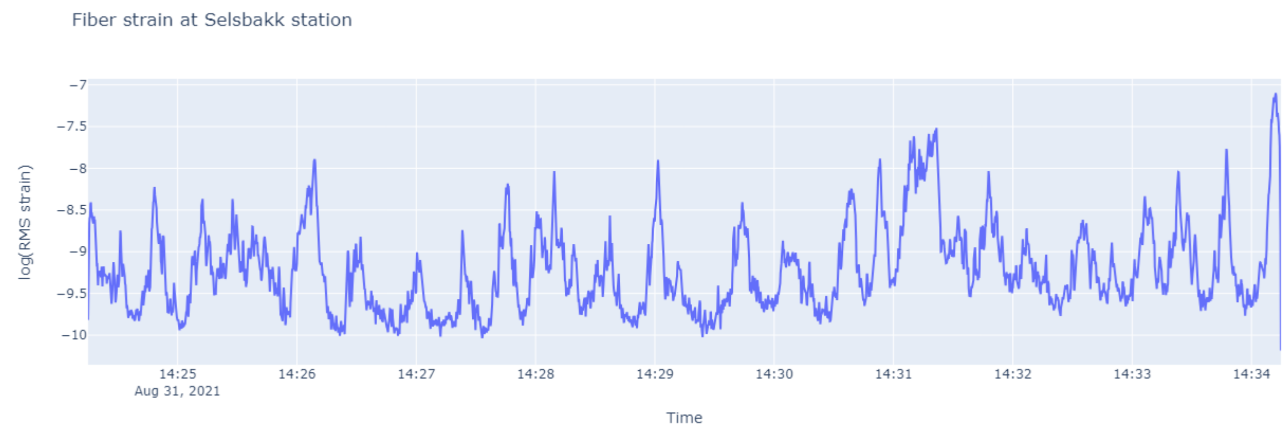


Figure 3.1: Log-RMS fiber strain at Selsbakk station (channel 1000) during 10 minutes.

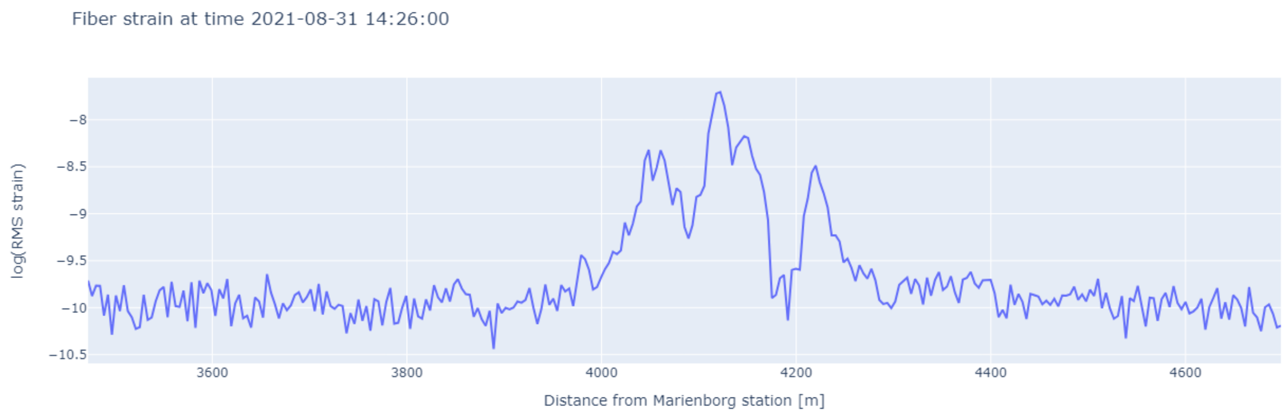


Figure 3.2: Logarithm of fiber strain across the entire railway segment at 14:26, the 31st of August 2021.

(at 4080 m), but these signals are not unimodal either, and those signals likely originating from the same target should be grouped together.

3.1 Data smoothing

Smoothing of time series is a common preprocessing technique used to reduce the impact of noise, and to uncover underlying patterns in the data. There are many common smoothing techniques, including moving average (Macaulay, 1931), kernel smoothing (Härdle & Vieu, 1992), low-pass filters (King & Rebelo, 1993) and exponential smoothing (Gardner Jr, 1985). We will move forward with the most straight forward of those, namely the moving average. The idea behind this technique is simple: Define a fixed size window, that moves through the time series. For every position of the window, the average of the data points inside the window at its current position is calculated, forming a new, smoother time series. It is important to note that when we apply this technique, a time lag is introduced in the data. The smoothed result is shifted in time compared to the original data, and the extent of the lag is dependent on the window size that we choose. The larger the window size, which we denote by κ, the larger the time lag, since we are basing the average on a longer history of strain data. In practice, one shifts the time series to center the smoothed signal at aligned time point. However, the selection of κ affects the variations in the signal. The parameter κ is a hyperparameter subject to optimization when we are searching for good signal picks that will be used as measurements in the tracking framework later on.

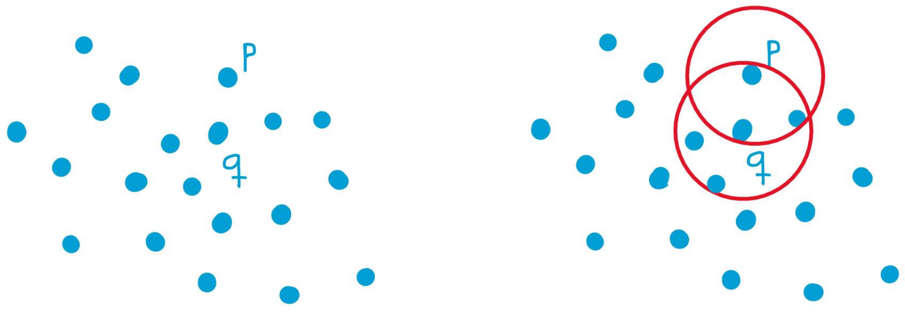


Figure 3.3: Asymmetric case of direct density-reachability. Here p is a border point, while q is a core point with some ε and $MinPts = 5$. Point p is directly density-reachable from q, but q is not directly density-reachable from p.

3.2 Signal peak clustering using DBSCAN

After obtaining a smoothed time series, we can analyze it and effectively create signal picks used for tracking later. First of all, it is necessary to establish a signal amplitude threshold a that determines a limit for which signal peaks will be considered noise, and not included in the signal extraction process. However, even after smoothing using an appropriate κ and only considering peaks with amplitudes above a, the data contains many signal peaks, and when comparing to the heatmap of the data, some of the peaks does not originate from any vehicle or train. In other words, each vehicle seems to potentially cause multiple peaks in time for every channel, and multiple peaks in space direction for each timestamp, as mentioned earlier. Therefore, we wish to group these signal peaks together. An appropriate approach for this problem is to apply a 2D clustering algorithm to the (channel,time) points at which the smoothed log-RMS strain data peaks.

Density-Based Spatial Clustering of Applications with Noise (Ester et al., 1996), or DBSCAN, is a popular clustering algorithm that groups data points based on their density, and identifies clusters of high-density regions and classifies potential outliers as noise. The main concept of the algorithm is that for each point in a cluster, the neighborhood of a given radius with the point in its center has to contain a minimum number of points. We define the *ε-neighborhood*, denoted by $N_\varepsilon(p)$ to be the neighborhood around a point p such that

$$N_\varepsilon(p) = \{q \in D | \text{dist}(p, q) < \varepsilon\}, \tag{3.1}$$

where D denotes the collection of data points, and $\text{dist}(p, q)$ is a distance function (typically Euclidean). We distinguish between two types of points in a cluster C: points inside of C (core points) and points on the border of C (border points). It is required that for every point $p \in C$ there is a point $q \in C$ so that $p \in N_\varepsilon(q)$ and $|N_\varepsilon(q)| >= MinPts$ (core point condition), where we define $MinPts$ to be a lower limit for how many points should be inside the radius of a point to be considered a core point. Point p is then *directly density-reachable* from q. This concept is illustrated in Figure 3.3. Here p is directly density-reachable from q, since $p \in N_\varepsilon(q)$ and $|N_\varepsilon(q)| >= MinPts$. Point q is not directly density-reachable from p since $|N_\varepsilon(p)| < MinPts$ even if $q \in N_\varepsilon(p)$. It is clear that for pairs of core points this condition holds symmetrically, this is however not true if we have one core point and one border point. Furthermore, a point p is *density reachable* from a point q for a given value of ε and $MinPts$ if there is a sequence of points $q = p_1, p_2, ..., p_n = p$ such that p_{i+1} is directly density-reachable from p_i. Also here, the condition is symmetric for pairs of core points, but not generally. The left side of Figure 3.4 illustrates this concept. Here q is a core point while p is a border point. p is density reachable from q since there exist a sequence of points from q to p such that the current point is directly density-reachable from the previous. The point q however is not density reachable from p, since $N_\varepsilon(p) < MinPts$, and therefore no point can be directly density reachable from p.

In the case where we have two border points in the same cluster, they are possibly not density reachable from each other since the core point condition might not hold for them both. However, there must be a core point from which both of these border points are density reachable, within the same cluster. We introduce the concept of *density-connectivity*, which ensures that border points in the same clusters are indirectly connected through core points, all though they are not directly density-reachable from each other. So, a point p is density-connected to a point q given an ε and $MinPts$ if there is a point o such that p and q are density reachable from o for the same ε and $MinPts$. This concept is illustrated in the right part of Figure 3.4.

With the concepts of density reachability and connectivity introduced, we can properly define the concept of cluster and noise. Let D be the collection of data points we wish to cluster. A cluster C with respect to ε and $MinPts$ is a non-empty subset of D that satisfies

1. $\forall p, q$: If $p \in C$ and q is density reachable from p with respect to ε and $MinPts$, then $q \in C$ (Maximality

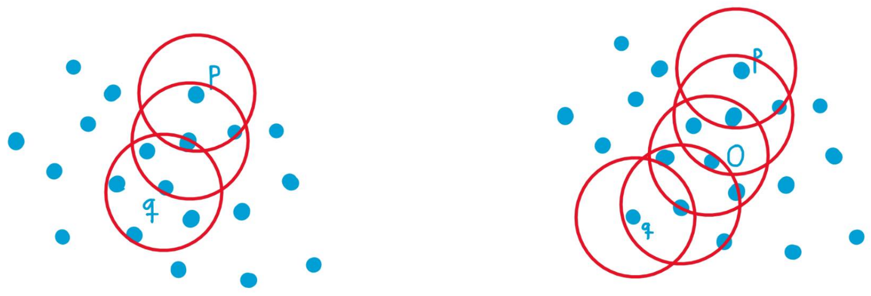


Figure 3.4: **Left:** Asymmetric case of density-reachability. Here p is a border point, while q is a core point with some ε and $MinPts = 5$. Point p is density-reachable from q, but q is not density-reachable from p. **Right:** Illustration of density-connectivity being a symmetric condition. Both p and q are border points, and they are density connected to each other by the point o.

condition),

2. $\forall p, q \in C$: p is density-connected to q with respect to ε and $MinPts$ (Connectivity condition).

On the other hand, we define noise as the set of points in D that does not belong to any of the clusters C_i in D. So

$$noise = \{p \in D | \forall i : p \notin C_i\} . \tag{3.2}$$

3.2.1 The algorithm

DBSCAN arbitrarily selects an initial point $p \in D$ and identifies all points that are density-reachable from p with respect to ε and $MinPts$. If p is a core point, then a cluster is created. However if p is a border point, it means that no points meet the criteria for density reachability from p and DBSCAN continues to the next point in D. The algorithm can be abstracted into the following three steps:

1. Find all neighbor points in the ε-neighborhood of every point in D, and identify core points, with more than $MinPts$ neighbors.

2. For each core point find all its density-connected points and assign them to the same cluster as the core point.

3. Iterate through the remaining unvisited points in D. Those points that do not belong to any cluster classifies as noise.

Algorithm 1 implements DBSCAN in its completeness, where the `RangeQuery` function computes distances from each point in D and checks which distances are lower than ε and thus qualify as neighbors to each point.

Figure 3.5 shows an example application of the DBSCAN algorithm to simulated data points in 2D, for a selection of hyperparameter choices. From the first row in the figure, we see that with a low ε, the data points need to be extremely close to each other to form a cluster. With $MinPts = 1$, a single point in itself can be a cluster, and therefore the result is many small clusters, even if some points actually can be considered noise. With increasing $MinPts$, we see that more and more data points are considered noise, since the data points are not located close enough to be considered a cluster. In the second row ε is slightly increased. In the cases where $MinPts > 1$, we see that the algorithm clustering results align better with the clusters we can visually observe. In the final row we further increase ε. Now the radius is clearly too large, since even obvious outliers falls within the radius, and thus form part of a cluster according to the algorithm. This can be slightly compensated by increasing $MinPts$, as can be seen in the plot in row 3 and column 3. All in all, ε and $MinPts$ significantly influences the clustering results in DBCAN, and the optimal combination depends on the characteristics of the data.

In summary, the signal extraction algorithm that we are proposing uses a combination of moving average smoothing and DBSCAN clustering, and the detailed procedure is described in Algorithm 2.

3.3 Evaluation of the signal picks

After having processed the strain data, and obtained a refined set of strain values eligible for representing measurements in a MOT scheme, we would like to evaluate the accuracy of the signal picks, in accordance to the logged data from

Algorithm 1 DBSCAN Algorithm

```
Input: $D$: Database
Input: $\varepsilon$: Radius
Input: $MinPts$: Density threshold
Input: dist: Distance function
for $p$ in $D$ do
    if label($p$) $\neq$ undefined then
        Continue
    end if
    Neighbors $N \leftarrow$ RangeQuery($D, dist, p, \varepsilon$)  ▷ Compute distances from each point in $D$, find points for which the distance is lower than $\varepsilon$.
    if $|N| < MinPts$ then
        label($p$) $\leftarrow$ Noise

        Continue
    end if
    $c \leftarrow$ next cluster label
    label($p$) $\leftarrow c$
    Seed set $S \leftarrow N \setminus \{p\}$
    for $q$ in $S$ do
        if label($q$) = Noise then
            label($q$) $\leftarrow c$
        end if
        if label($q \neq$ Undefined) then
            Continue
        end if
        Neighbors $N \leftarrow$ RangeQuery($D, dist, p, \varepsilon$)  ▷ Compute distances from each point in $D$, find points for which the distance is lower than $\varepsilon$.
        label($q$) $\leftarrow c$
        if $|N| < MinPts$ then
            Continue
        end if
        $S \leftarrow S \cup N$
    end for
end for
```

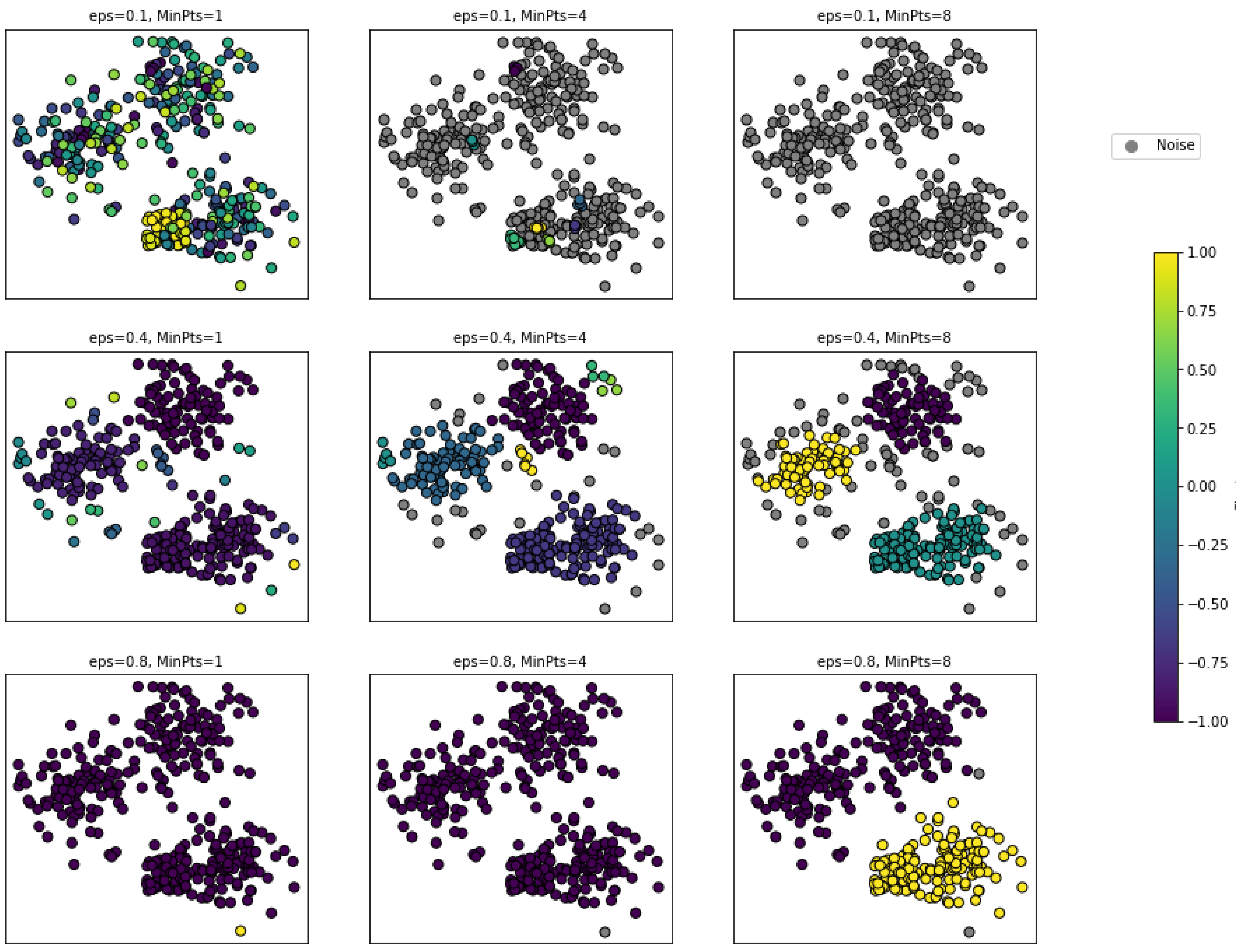


Figure 3.5: DBSCAN applied to simulated 2D data points, for all combinations of ε and $MinPts$ when $\varepsilon \in \{0.1, 0.4, 0.8\}$ and $MinPts \in \{1, 4, 8\}$.

Algorithm 2 Signal Extraction Algorithm (Parameters are tuned from Hausdorff distance between picks and log data)

```
Input: $D$: Strain array
Input: $\varepsilon$: Radius
Input: $MinPts$: density threshold
Input: dist: Distance function
Input: $a$: Amplitude threshold
Input: $\kappa$: window size
$N, M = D$.shape
$D^* \leftarrow$ EmptyArray$(N, M)$
for $i$ in range$(M)$ do                                   ▷ Iterate over each channel in the strain array.
    $D^*[:, i] \leftarrow$ smooth$(D[:, i], \kappa)$                    ▷ $D^*$ is the smoothed strain array.
end for
$P \leftarrow$ identify_peaks$(D^*, a)$    ▷ identify_peaks returns the (channel,time)-points of the strain peaks in $D^*$.
$C \leftarrow$ DBSCAN$(P, \varepsilon, MinPts)$
$D^{**} \leftarrow$ ZeroArray$(N, M)$
for $label$ in unique$(C)$ do                              ▷ Iterate through every unique peak cluster.
    if $label \neq$ -1 then
        $P_c \leftarrow P[label ==$ unique$(C)]$   ▷ $P_c$ contains the (channel,time)-points corresponding to all peaks belonging to cluster $label$.
        $\bar{t} \leftarrow$ mean$(P_c[:, 0])$                ▷ Get the mean timestamp of all peaks in cluster $label$.
        $\bar{x} \leftarrow$ mean$(P_c[:, 1])$                  ▷ Get the mean channel of all peaks in cluster $label$.
        $D^{**}[\bar{t}, \bar{x}] \leftarrow D^*[\bar{t}, \bar{x}]$
    end if
end for
Output: $D^{**}$: Processed strain array containing smoothed strain amplitudes from $D^*$ at entry $(\bar{x}, \bar{t})$, where $\bar{x}, \bar{t}$ are the rounded average channel and timestamp respectively in each cluster.
```

Selsbakk station presented in the previous chapter. We can view these data points as two distinct sets. Denote the set with picks by P, and the set with logged events by E. We need a well-defined metric than can measure the overall alignment of the points in the two sets. The Hausdorff distance is a good option, as it effectively identifies similar or dissimilar shapes in clusters of data, which indeed is our case.

3.3.1 The Hausdorff distance

Let (M, d) be a metric space, so that M is a set, and $d : M \times M \to \mathbb{R}$ is a distance metric on M. For every pair of subsets $P \subset M$ and $E \subset M$, the Hausdorff distance (Rockafellar & Wets, 2009) between P and E is defined as

$$d_H(P, E) = \max \left\{ \sup_{x \in P} d(x, E), \sup_{y \in E} d(y, P) \right\}, \tag{3.3}$$

where $d(a, B) = \inf_{b \in B} d(a, b)$ is the smallest distance from a point $a \in P$ to the subset $B \subseteq P$. In other words, it is the maximum between the largest distance from a point in P to the closest point in E, and the largest distance from a point in E to the closest point in P. So, two sets P and E are close in Hausdorff distance if every point in either set is close to some point in the other set. Figure 3.6 illustrates this concept. The application of the Hausdorff distance between the signal picks and logged events will give a decent picture of the conformity between signal picks and true event times, and in consequence work as a loss function in the process of optimizing the hyperparameter configuration used to produce the signal picks.

3.3.2 Optimization of hyperparameters

In the signal extraction algorithm proposed in Algorithm 2, there are a number of hyperparameters that needs to be set:

- κ - window size in moving average smoothing.
- a - strain amplitude threshold.
- ε - radius in the DBSCAN algorithm.
- $MinPts$ - minimum number of points within ε for a point to be considered a core point.

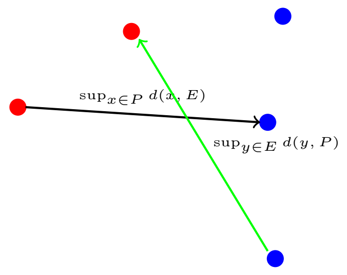


Figure 3.6: Here, the largest distance from a point in P (red set) to the closest point in E (blue set), and vice versa, is displayed. The Hausdorff distance between P and E is the maximum of these, and is the distance colored in green.

The three first hyperparameters are subject to optimization, as they can lead to signal picks of different quality depending on how they are set. The values of κ, a and ε should be set so that the Hausdorff distance between the resulting set of signal picks P and logged events E is minimized. Vehicles, depending on their noise level, can cause strain signals of varying amplitude for different channels. A single vehicle can cause unimodal signals but also sometimes multimodal signals. The idea is that the DBSCAN algorithm will cluster signal peaks that are very close together, and that are assumed to originate from the same vehicle. But since vehicles can cause unimodal signals, single signal peaks must also be treated as a cluster in itself, if it reaches above the amplitude threshold a, which is meant to filter out background noise. Therefore, $MinPts$ must be set to 1 so that every data point in the clustering scheme can be considered as a separate cluster.

We use a simple, grid-based optimization technique to find optimal hyperparameters. By looking at the strain data, we found that an acceptable initial value for the amplitude threshold was $a = -8.8$, and initially $\varepsilon = 0.05$. With two of the hyperparameters fixed, we then calculate $d_H(P, E)$ for a grid $\kappa = 1, 2, ..., 40$ of window sizes. The Hausdorff distances for the respective κ can be seen in Figure 3.7. There is clearly a minimum at $\kappa = 30$, and thus, κ is fixed to this value when we optimize a. We compute $d_H(P, E)$ for a ranging from -9.5 to -8, with a step size of 0.05. Figure 3.8 shows the Hausdorff distances for all amplitude thresholds, and we find a minimum at $a = -8.8$, surprisingly enough our initial guess.

With κ and a fixed to their optimal value, we can proceed to optimizing ε. However, direct optimization of this parameter will not lead to a satisfactory result, since $|P|$ will increase proportionally with the decrease of ε, which in turn will lead to a decrease in $d_H(P, E)$. Essentially, the more points in P, the smaller the maximum distance from a point in P will be to the closest point in E. Therefore, it is necessary to regularize $d_H(P, E)$, by adding a penalty term that penalizes the metric based on the number of signal picks. We use a penalty term given by

$$\alpha \left| |P| - |E| \right|, \tag{3.4}$$

so that the objective function to be optimized becomes

$$d_H(P, E) + \alpha \left| |P| - |E| \right|. \tag{3.5}$$

Here α is a non-negative regularization parameter. We calculate (3.5) on a grid of ε ranging from 10^{-5} to 0.2 , with step size 0.001, and with $\alpha = 10$. The resulting Hausdorff distances are displayed in Figure 3.9, and a minimum appears approximately at $\varepsilon = 0.01$. Figure 3.10 shows a heatmap of the strain data, with the resulting signal picks together with the logged events. The size of the black circles are proportional to the average strain value in the particular cluster.

While the hyperparameter values we have found will provide signal picks best possibly aligned to the logged events at channel 1000, compared to other values, it is by no means a guarantee that it will lead to the best set of signal picks across all channels. In fact, by further experimentation with the hyperparameters and signal picking across all channels, it seems that with this configuration we actually loose some important signal picks at some channels, compared to using other values for κ, a and ε. However, we also saw that the signal picks aligned very nicely with the amplitude peaks at most channels. We therefore believe that the signal picks using the optimized hyperparameter values will suffice in quality when they are to be used as measurements in MOT. Another note to be made, is that $d_H(P, E)$ remains fairly low and constant for a certain range of values of κ, a, ε, which in turn implies that the signal picks will be of decent quality for a range of these hyperparameters. So even if we are not guaranteed the best possible picks by using the mentioned hyperparameter values, there will be fairly little consequences of this in terms of tracking performance. The purpose of using the Hausdorff distance for optimization was merely to get a rough overview for appropriate values for κ, a and ε. To show the importance of choosing appropriate hyperparameter values in terms of signal pick quality, we created signal picks with different hyperparameter values shown in Figure 3.11 and 3.12. In Figure 3.11 a window size of $\kappa = 3$ was used, which corresponds to very little smoothing in time. Consequently, a much larger number of clusters appears in the time direction, and a high number of irrelevant picks also appears with

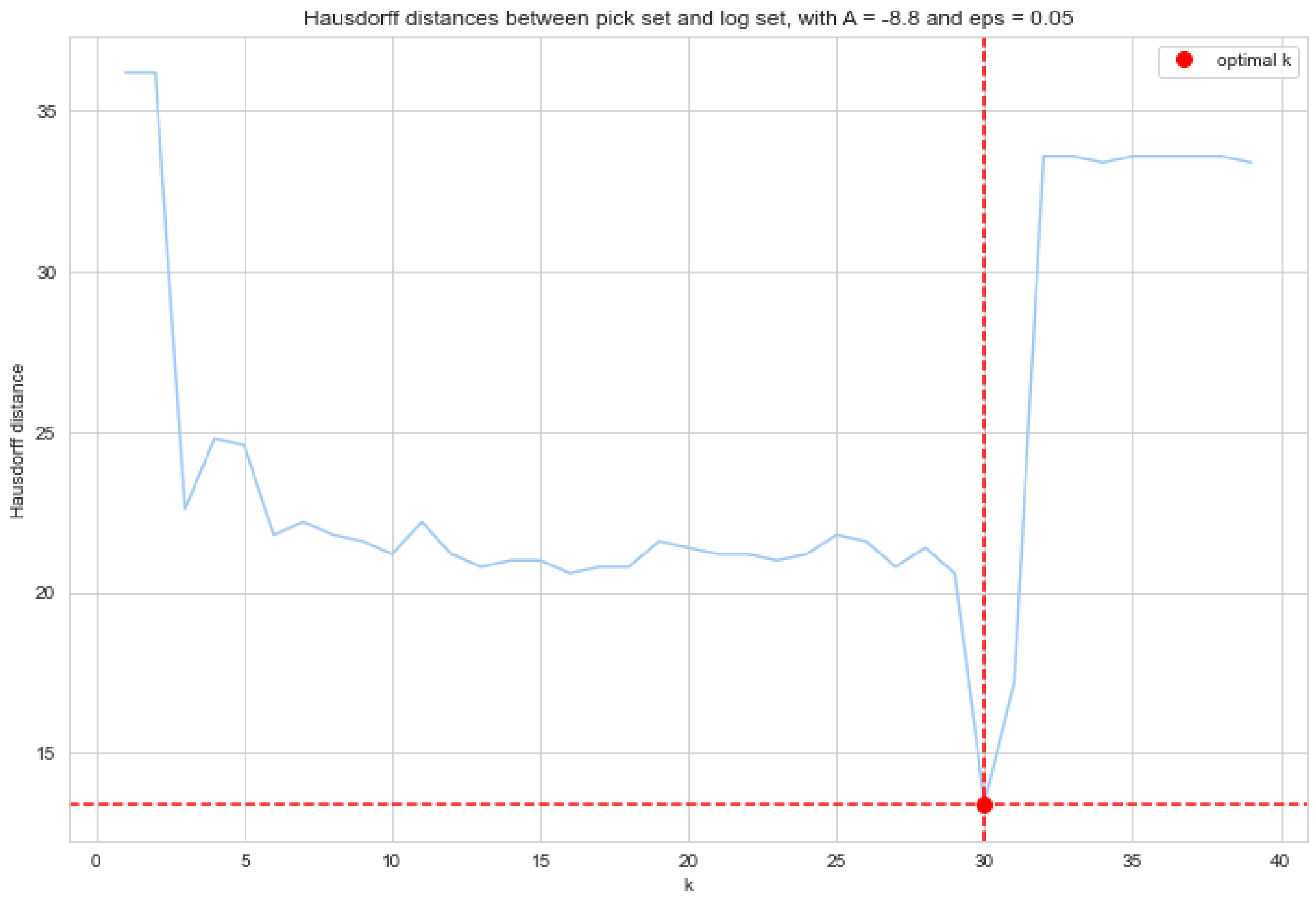


Figure 3.7: Hausdorff distances for a selection of κ-values, with $a = -8.8$ and $\varepsilon = 0.05$.

this choice of κ. This will lead to difficulties with DAs if we would apply a tracking algorithm to these picks. So, with a too low window size, we see that we are not able to gather high-amplitude signals in time originating from the same object to a satisfactory extent. In Figure 3.12 a radius of $\varepsilon = 0.1$ was used. Here we see that way too many signal peaks are being merged into the same cluster due to the high radius, and consequently we end up with much fever signal picks. A tracking algorithm would not be able to effectively estimate the state of the objects since there would be too few measurements to work with.

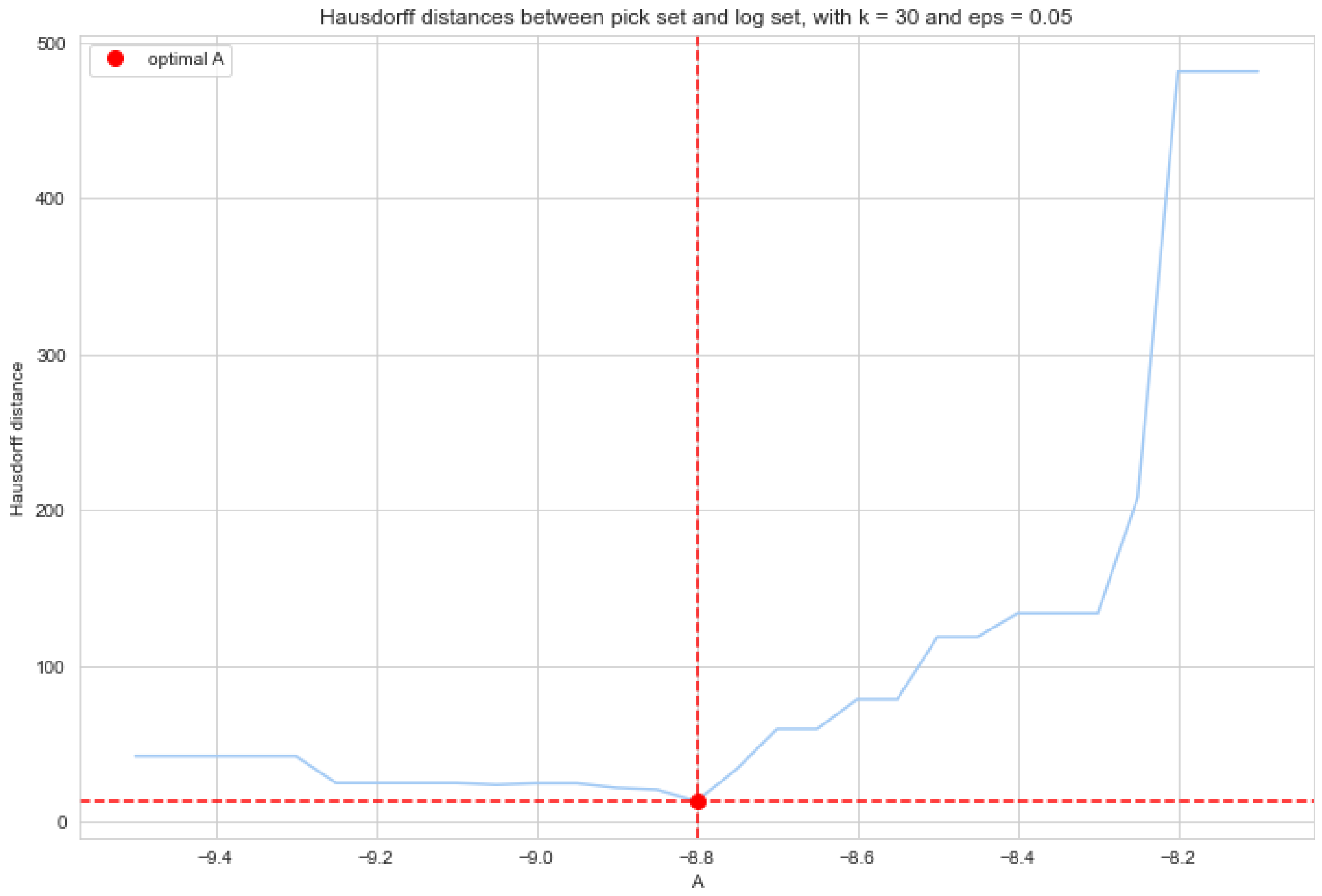


Figure 3.8: Hausdorff distances for a selection of a-values, with $\kappa = 30$ and $\varepsilon = 0.05$.

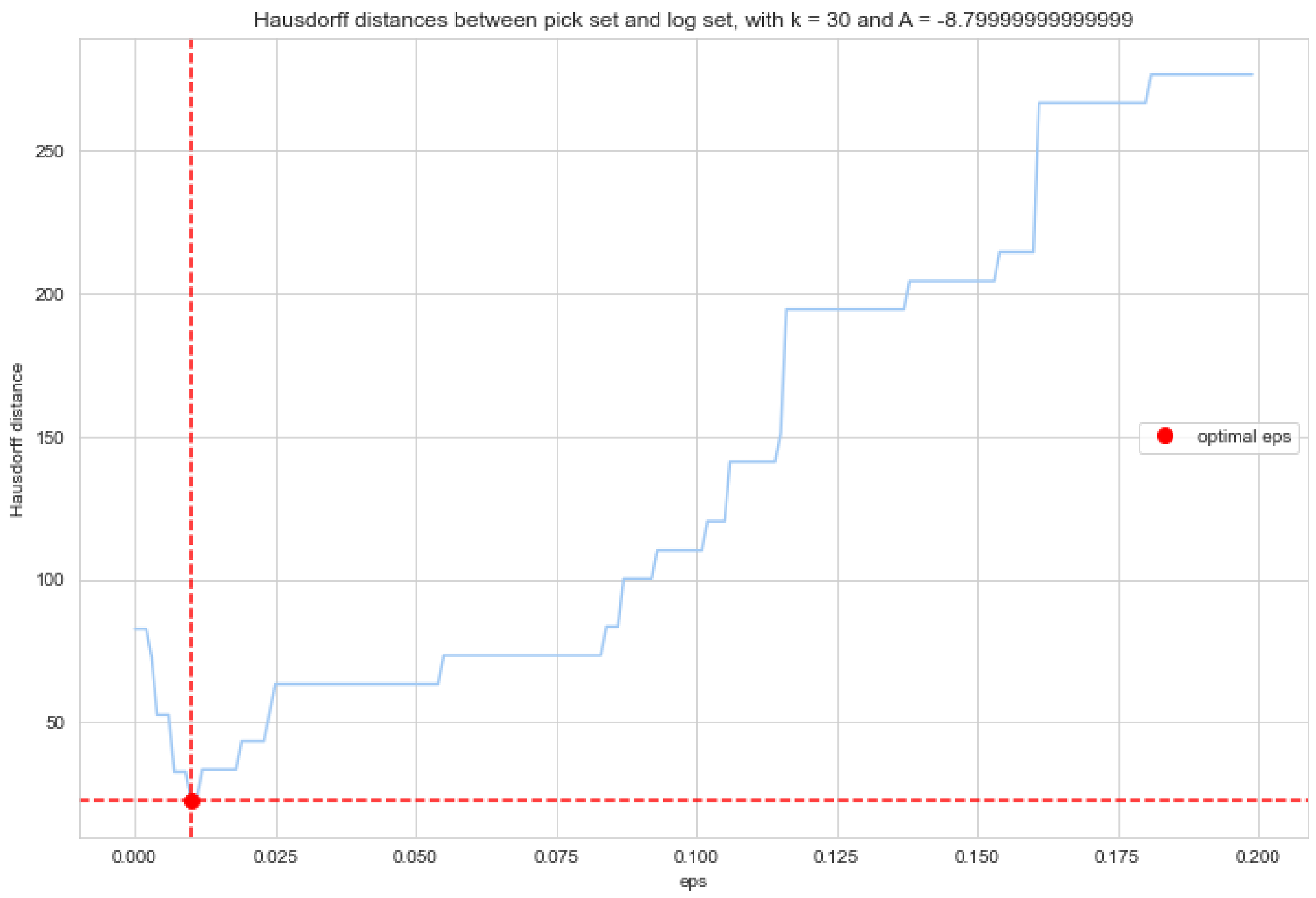


Figure 3.9: Hausdorff distances for a selection of ε-values, with $\kappa = 30$ and $a = -8.8$.

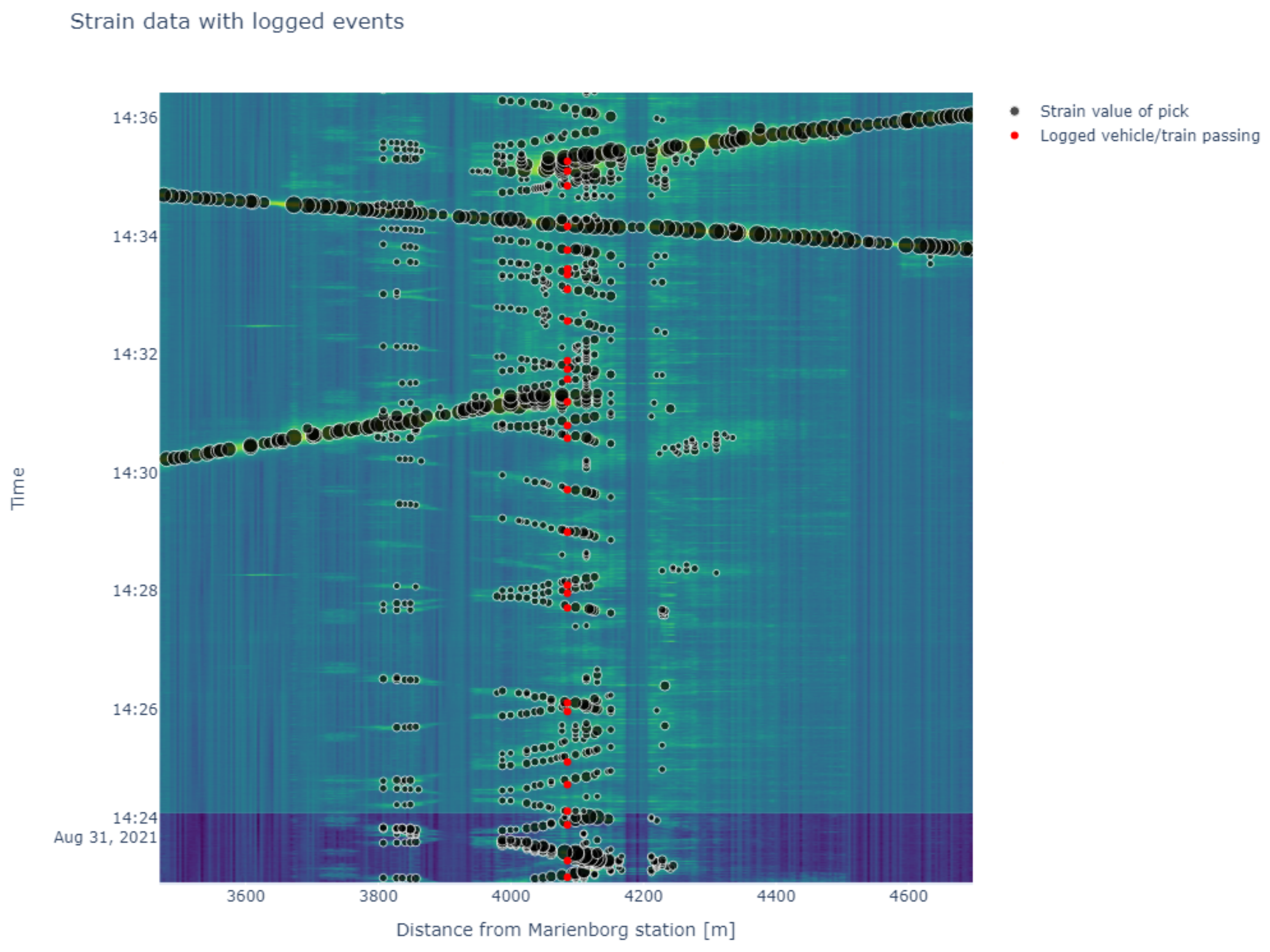


Figure 3.10: Heatmap of strain data, with signal picks using hyperparameters $\kappa = 30$, $a = -8.8$, $\varepsilon = 0.01$ and $MinPts = 1$.

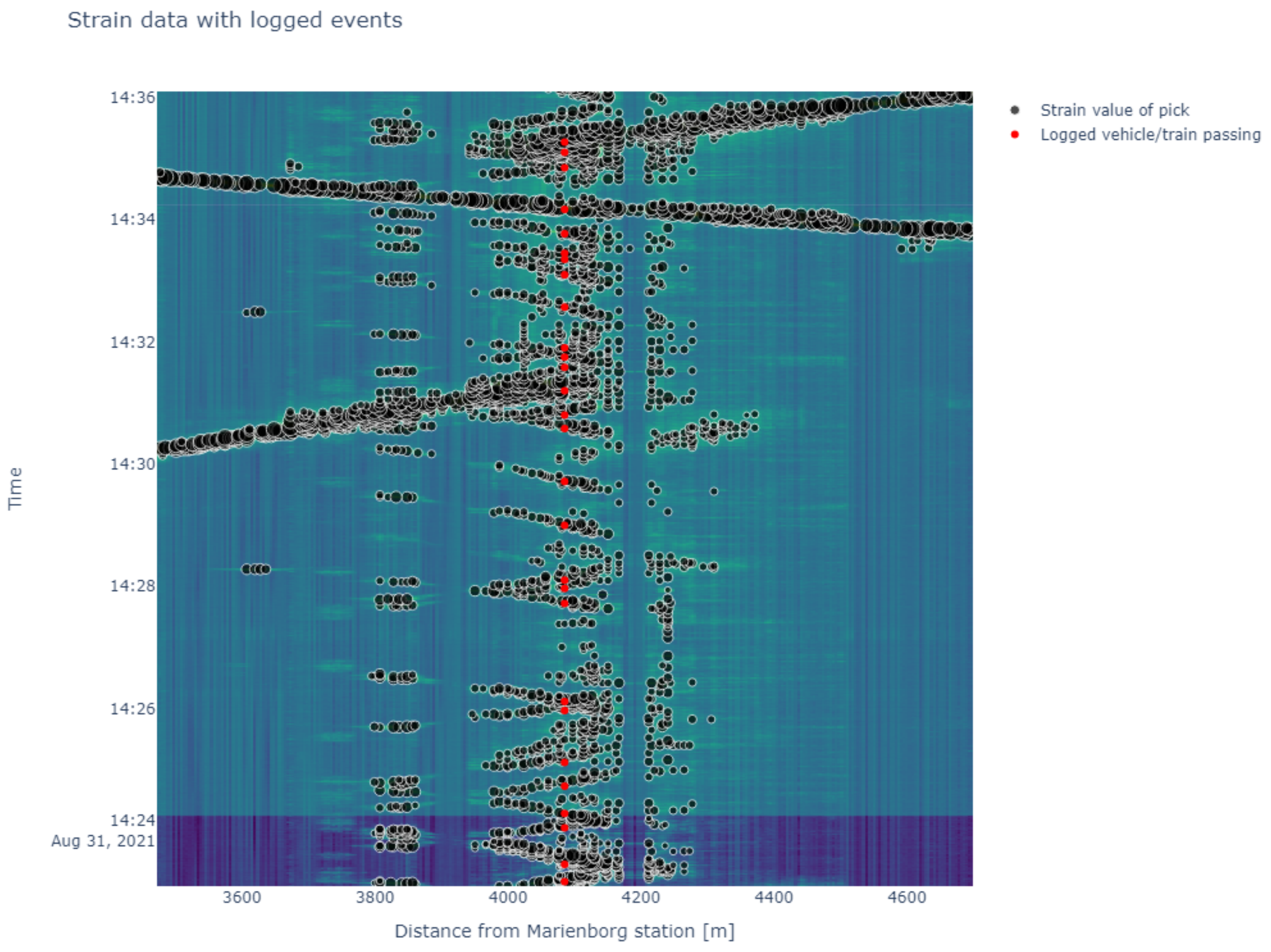


Figure 3.11: Heatmap of strain data, with signal picks when using $\kappa = 3$, $a = -8.8$, $\varepsilon = 0.01$ and $MinPts = 1$.

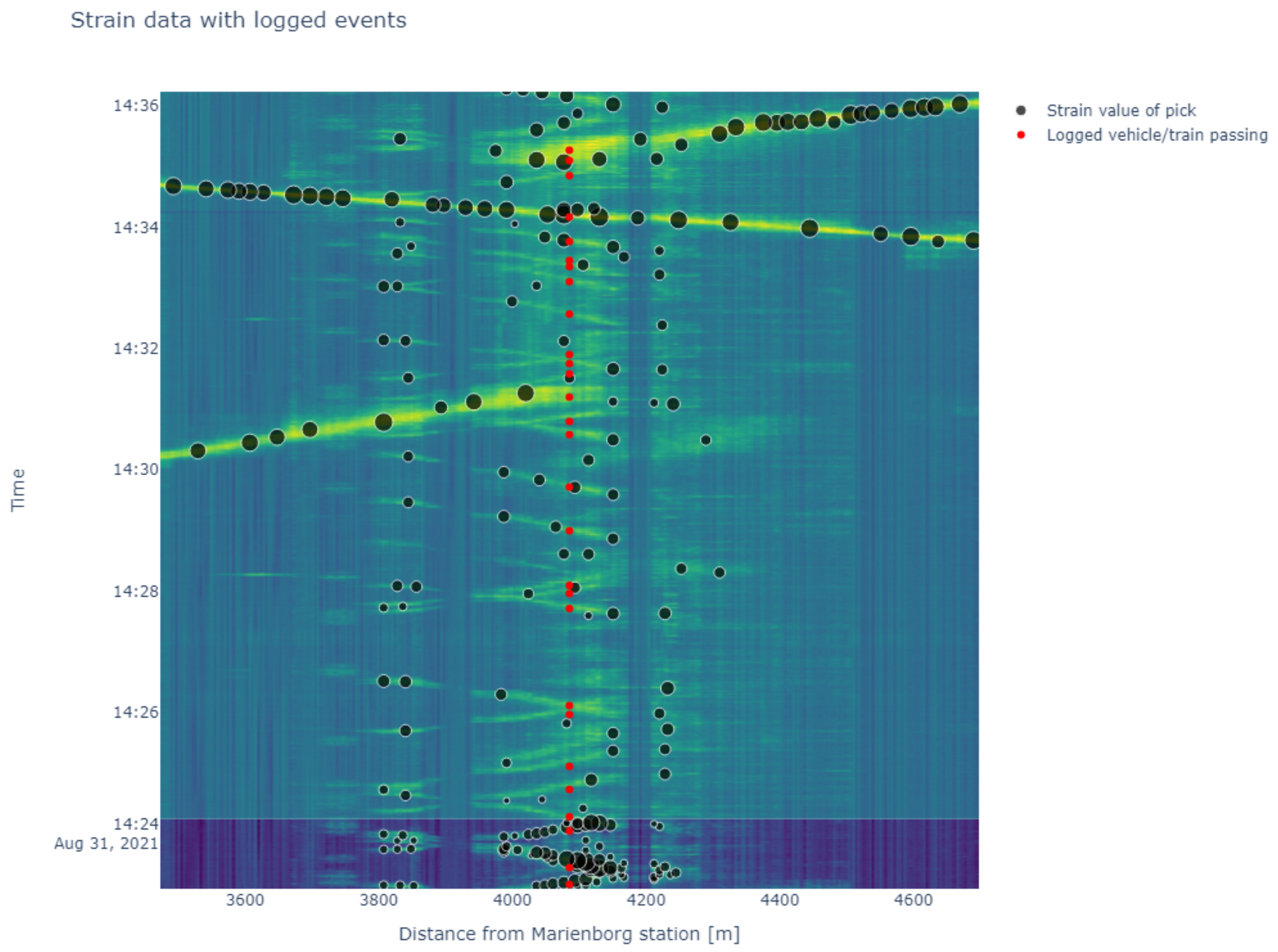


Figure 3.12: Heatmap of strain data, with signal picks when using $\kappa = 30$, $a = -8.8$, $\varepsilon = 0.1$ and $MinPts = 1$.

Chapter 4

The JPDA algorithm

In this chapter we discuss the necessary building blocks for the MOT framework, and give a thorough theoretical description of the JPDA filter (Bar-Shalom & Li, 1995). We follow the notation used in the material of L. Svensson (n.d.). **Important:** Some parts of the theory and the equations derived in this chapter are similar to the ones discussed in Fredriksen (2022), all though with some extensions. For the implementation of the JPDA tracker we used the Stone Soup Framework (URL: https://stonesoup.readthedocs.io/en/v1.0/#), which is a Python library for state estimation and tracking. In addition, we provide an example use of the JPDA tracker in a simulation study.

4.1 Bayesian Filtering and the Kalman filter

As explained in Särkkä (2013), the primary goal of Bayesian filtering is to estimate a latent unobserved random variable, often called the state $\mathbf{x}$, from a noisy observed random variable, commonly known as the measurement $\mathbf{z}$. In the context of this thesis, the state will be a two-dimensional random variable, consisting of the position and velocity of an object moving in the time-space plane. $\mathbf{z}$ represents noisy measurements of the position of the object, reducing it to a one-dimensional random variable. Given time steps $k = 1, 2, ..., T$, the objective is to estimate $\mathbf{x}_k$ at each time step k based on the history of measurements $\mathbf{Z}_{1:k} = \{\mathbf{z}_1, ..., \mathbf{z}_k\}$. An important note is that every $\mathbf{z}_k$ here has a different size, as we at each new time step encounter not only object measurements, but also data from unknown sources, referred to as *clutter*.

In the context of state space models, the initial state is assumed to be distributed according to an initial prior density. Furthermore, the state sequence $\{\mathbf{x}_k, k = 1, 2, ..., T\}$ can be formulated as a Markov model. We also assume the measurements $\mathbf{z}_k$ to be conditionally independent given the current state $\mathbf{x}_k$, with a corresponding conditional density, so that

$$\begin{aligned} \mathbf{x}_0 &\sim p(\mathbf{x}_0), \\ \mathbf{x}_k &\sim p(\mathbf{x}_k|\mathbf{x}_{k-1}), \\ \mathbf{z}_k &\sim p(\mathbf{z}_k|\mathbf{x}_k), \end{aligned} \tag{4.1}$$

where $p(\mathbf{x}_0)$ denotes the prior density of the initial state, $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is the dynamical model and $p(\mathbf{z}_k|\mathbf{x}_k)$ is the measurement model. The dynamical model describes the stochastic dynamics of the system, and is used in the prediction step in the filtering process. The measurement model characterizes the density of measurements given the state. Figure 4.1 illustrates the state space model concept. The black arrows represents the dynamical model, the blue represents the measurement model, while the absence of arrows between the measurements reflects the conditional independence between measurements. The Markov property is also illustrated since arrows connect neighboring states only.

By making the same assumptions as in Särkkä (2013) and Bar-Shalom and Li (1995), the filtering density can be

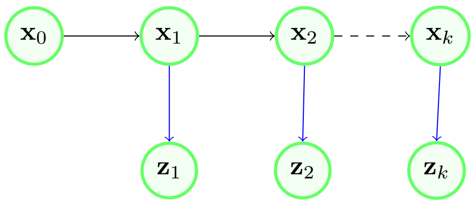


Figure 4.1: The state space model concept.

written as

$$p(\mathbf{x}_k|\mathbf{Z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{Z}_{1:k-1})}, \tag{4.2}$$

where the predicted density is determined by the Chapman-Kolmogorov equation (Särkkä, 2013),

$$p(\mathbf{x}_k|\mathbf{Z}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{Z}_{1:k-1})d\mathbf{x}_{k-1}. \tag{4.3}$$

So, the posterior estimate of the state at time $k-1$ is the prior for the state at time k. The Bayesian filtering process therefore consists of two steps: Predict using (4.3), and update using (4.2). Figure 4.2 shows the conceptual idea behind this predict-update logic.

In order to simplify Bayesian filtering and improve computational efficiency, we adopt the KF theory (Kalman, 1960). We assume Gaussian measurement likelihood and predictive distributions, and linear dynamic and measurement models. With these simplifications, we can derive a closed-form expression of the Gaussian posterior state density. We will denote a Gaussian density function evaulated at $\mathbf{x}$ by $\phi(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$, and a Gaussian random variable with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ by $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. The state space model can thus be written as

$$\mathbf{x}_k = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{q}_{k-1}, \qquad \mathbf{q}_{k-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k-1}), \tag{4.4}$$

$$\mathbf{z_k} = \mathbf{H}_k\mathbf{x}_k + \mathbf{r}_{k-1}, \qquad \mathbf{r}_{k-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_{k-1}). \tag{4.5}$$

However, in our case the transition and covariance matrices will be constant, since they are modelling objects moving according to Newtons second law in discretized time. So $\mathbf{A}_k = \mathbf{A}, \mathbf{H}_k = \mathbf{H}, \mathbf{Q}_k = \mathbf{Q}$ and $\mathbf{R}_k = \mathbf{R}$, resulting in a probabilistic model (Särkkä, 2013) given by

$$p(\mathbf{x}_k|\mathbf{x}_{k-1}) = \phi(\mathbf{x}_k; \mathbf{A}\mathbf{x}_{k-1}, \mathbf{Q}), \tag{4.6}$$

$$p(\mathbf{z}_k|\mathbf{x}_k) = \phi(\mathbf{z}_k; \mathbf{H}\mathbf{x}_k, \mathbf{R}), \tag{4.7}$$

where

$$\mathbf{A} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix} \frac{1}{3}\Delta t^3 & \frac{1}{2}\Delta t^2 \\ \frac{1}{2}\Delta t^2 & \Delta t \end{bmatrix} \sigma_q^2, \quad \mathbf{H} = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} \sigma_r^2 \end{bmatrix}.$$

Inserting this probabilistic model into the Bayesian filtering equations, the resulting densities become

$$p(\mathbf{x}_k|\mathbf{Z}_{1:k-1}) = \phi(\mathbf{x}_k; \boldsymbol{\mu}_{k|k-1}, \mathbf{P}_{k|k-1}), \tag{4.8}$$

$$p(\mathbf{x}_k|\mathbf{Z}_{1:k}) = \phi(\mathbf{x}_k; \boldsymbol{\mu}_{k|k}, \mathbf{P}_{k|k}), \tag{4.9}$$

$$p(\mathbf{z}_k|\mathbf{Z}_{1:k-1}) = \phi(\mathbf{x}_k; \mathbf{H}\boldsymbol{\mu}_{k|k-1}, \mathbf{S}_k), \tag{4.10}$$

where the prediction step is given by

$$\boldsymbol{\mu}_{k|k-1} = \mathbf{A}\boldsymbol{\mu}_{k-1|k-1}, \tag{4.11}$$

$$\mathbf{P}_{k|k-1} = \mathbf{A}\mathbf{P}_{k-1|k-1}\mathbf{A}^T + \mathbf{Q}, \tag{4.12}$$

and the corresponding update yields

$$\bar{\mathbf{z}}_{k|k-1} = \mathbf{H}\boldsymbol{\mu}_{k|k-1}, \tag{4.13}$$

$$\mathbf{S}_k = \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^T + \mathbf{R}, \tag{4.14}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}^T(\mathbf{S}_k)^{-1}, \tag{4.15}$$

$$\boldsymbol{\mu}_{k|k} = \boldsymbol{\mu}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \bar{\mathbf{z}}_{k|k-1}), \tag{4.16}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{H}\mathbf{P}_{k|k-1}. \tag{4.17}$$

4.2 The multi object measurement model

We can now introduce the various concepts that constitute the MOT framework. In Fredriksen (2022), these concepts were studied in the case of a constant number of objects to track, and the theory presented in this thesis also makes the assumption of a constant number of objects. We realize that this is somewhat limiting, but our focus in this thesis is the practical implementation of a tracking algorithm allowing for time-varying number of objects, and not the rigorous theoretical framework for this. Therefore, we present the theory with this assumption ($n_k = n \ \forall k$), and incorporate mechanisms allowing for time-varying number of objects in the JPDA algorithm. First, we develop

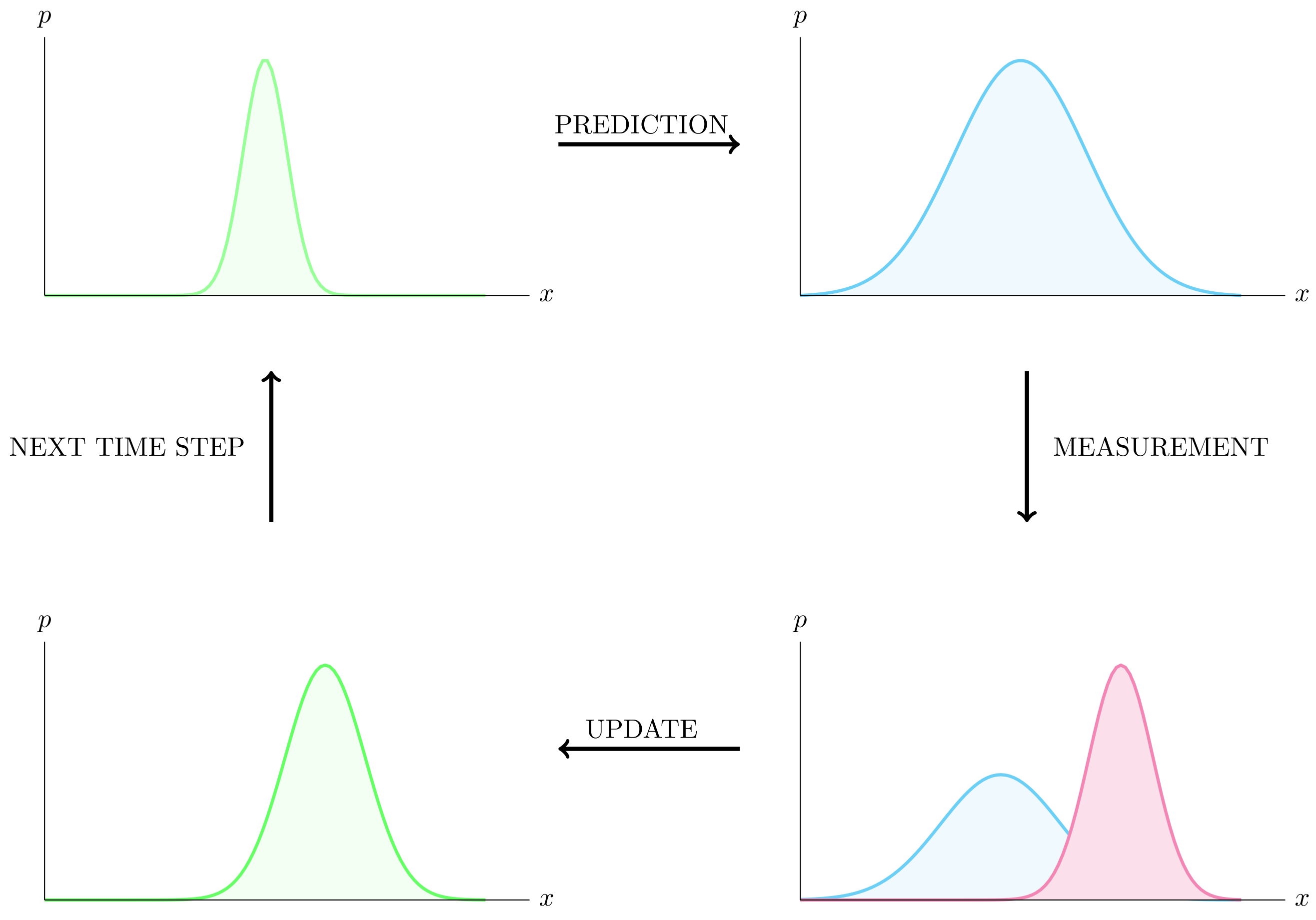


Figure 4.2: Illustration of the conceptual idea behind Bayesian filtering.

a comprehensive multi object measurement model that includes clutter measurements, object detections and missed detections. In MOT, we make use of the object states matrix

$$\mathbf{X}_k = [\mathbf{x}_k^1, ..., \mathbf{x}_k^n] \in \mathbb{R}^{2\times n}, \tag{4.18}$$

which will contain the position and velocity of the objects at every time step k. Note that at time k there are n columns in $\mathbf{X}_k$, which implies that there are n objects in the physical area in which we are tracking our targets, commonly known as the Field of View (FOV) V. Furthermore, we assume that the object states are independent of each other at time $k = 0$, so that their initial state density $p(\mathbf{X}_0) = p(\mathbf{x}_0^1, ..., \mathbf{x}_0^n)$ can be factorized as

$$p(\mathbf{X}_0) = \prod_{i=1}^{n} p^i(\mathbf{x}_0^i). \tag{4.19}$$

With the Gaussian assumption, the prior density becomes

$$p(\mathbf{X}_0) = \prod_{i=1}^{n} \phi(\mathbf{x}_0^i; \mu_0^i, \mathbf{P}_0^i). \tag{4.20}$$

We assume that each object i is detected with a detection probability denoted by $P^D(\mathbf{x}_k^i)$, and in this thesis this quantity will be constant for all objects, i.e. $P^D(\mathbf{x}_k^i) = P^D, \forall i, k$. If object i is detected, it will generate a measurement from

$$p(\mathbf{o}_k^i|\mathbf{x}_k^i) = g_k(\mathbf{o}_k^i|\mathbf{x}_k^i), \tag{4.21}$$

where

$$\mathbf{o}_k^i = \mathbf{H}\mathbf{x}_k^i + \mathbf{r}_k, \qquad \mathbf{r}_k \sim \mathcal{N}(0, \mathbf{R}), \tag{4.22}$$

so that

$$g_k(\mathbf{o}_k^i|\mathbf{x}_k^i) = \phi(\mathbf{o}_k^i; \mathbf{H}\mathbf{x}_k^i, \mathbf{R}). \tag{4.23}$$

We define the *observation matrix* as

$$\mathbf{O}_k = [\mathbf{O}_k^1, ..., \mathbf{O}_k^n], \tag{4.24}$$

where

$$\mathbf{O}_k^i = \begin{cases} \mathbf{o}_k^i & \text{with probability } P^D, \\ [] & \text{with probability } 1 - P^D. \end{cases} \tag{4.25}$$

Clutter are the measurements that do not originate from any of the objects in the FOV at a specific time. We define the clutter detection matrix as

$$\mathbf{C}_k = [\mathbf{c}_k^1, ..., \mathbf{c}_k^{m_k^c}], \tag{4.26}$$

with $|\mathbf{C}_k| = m_k^c$. Having accounted for both detections and missed detections, in addition to clutter measurements, we can construct the observed measurement matrix as $\mathbf{Z}_k = \Pi(\mathbf{O}_k, \mathbf{C}_k)$. Here Π is the random column permuter. We can use the Poisson Point Process (PPP) as a probabilistic model for clutter, and model the number of clutter detections at time k as a Poisson random variable with

$$m_k^c \sim \text{Poisson}(\lambda V). \tag{4.27}$$

Given m_k^c, the clutter points $\mathbf{c}_k^1, ..., \mathbf{c}_k^{m_k^c}$ are i.i.d uniformly distributed over V. The PPP is parametrized by using an *intensity function* $\lambda_c(\mathbf{c}) \geq 0$. This intensity function can also be represented as a combination of a rate given by $\bar{\lambda} = \int \lambda_c(\mathbf{c}) d\mathbf{c}$ and a spatial probability density function given by $f_c(\mathbf{c}) = \frac{\lambda_c(\mathbf{c})}{\bar{\lambda}_c}$. This means that

$$\lambda_c(\mathbf{c}) = \begin{cases} \lambda & \text{if } \mathbf{c} \in V, \\ 0 & \text{otherwise,} \end{cases} \tag{4.28}$$

and

$$\bar{\lambda}_c = \lambda V, \qquad f_c(\mathbf{c}) = \begin{cases} \frac{1}{V} & \text{if } \mathbf{c} \in V, \\ 0 & \text{otherwise.} \end{cases} \tag{4.29}$$

By utilizing the Probability Mass Function (PMF) of a Poisson random variable and assuming i.i.d. clutter detections, we can formulate the clutter model as follows:

$$\begin{aligned} p(\mathbf{C}_k) = p(\mathbf{C}_k, m_k^c) = p(m_k^c)p(\mathbf{C}_k|m_k^c) &= \text{Poisson}(m_k^c; \bar{\lambda}_c) \prod_{i=1}^{m_k^c} f_c(\mathbf{c}_k^i) \\ &= \frac{\exp\{-\bar{\lambda}_c\}\bar{\lambda}_c^{m_k^c}}{m_k^c!} \prod_{i=1}^{m_k^c} \frac{\lambda_c(\mathbf{c}_k^i)}{\bar{\lambda}_c} = \frac{\exp\{-\bar{\lambda}_c\}}{m_k^c!} \prod_{i=1}^{m_k^c} \lambda_c(\mathbf{c}_k^i). \end{aligned} \tag{4.30}$$

DA is the mechanism that relates an object to a specific measurement at a specific time step. We define the DA variable θ_k as

$$\theta_k = [\theta_k^1, ..., \theta_k^i, ..., \theta_k^n], \tag{4.31}$$

where

$$\theta_k^i = \begin{cases} j & \text{object } i \text{ is associated to measurement } j, \\ 0 & \text{object } i \text{ is undetected.} \end{cases} \tag{4.32}$$

Since we have n objects at time k, to ensure consistent DA, we must introduce a set of rules that defines the notion of a *valid* DA. Let Θ_k be the set of valid DAs at time k. For every $\theta_k \in \Theta_k$, it must hold that

1. Every object is either detected or undetected:

$$\theta_k^i \in \{0, ..., m_k\}, \forall i \in \{1, ..., n\}, \tag{4.33}$$

   where $m_k = |\mathbf{Z}_k|$.

2. Each pair of objects cannot share the same measurement:

$$\forall i, i' \in \{1, ..., n\} \text{ s.t. } i \neq i', \text{ if } \theta_k^i \neq 0 \text{ and } \theta_k^{i'} \neq 0 \implies \theta_k^i \neq \theta_k^{i'}. \tag{4.34}$$

These constraints ensures that at most n measurements gets associated at time k. We can express the observation matrix for object i at time k as

$$\mathbf{O}_k^i = \begin{cases} \mathbf{z}_k^{\theta_k^i} & \text{if } \theta_k^i \neq 0, \\ [] & \text{if } \theta_k^i = 0. \end{cases} \tag{4.35}$$

Those measurements that are clutter, are the measurements $\mathbf{z}_k^j$ that satisfy the following condition:

$$j \in \{1, ..., m_k\} \text{ s.t. } \nexists i \in \{1, ..., n\} \text{ s.t. } \theta_k^i = j. \text{ Abbreviation: } j : \nexists \theta_k^i = j. \tag{4.36}$$

By letting $m_k^o = |\mathbf{O}_k| = \sum_{i:\theta_k^i \neq 0} 1$, the number of clutter measurements can be calculated as $m_k^c = |\mathbf{C}_k| = m_k - m_k^o$. We can express the probability of detecting a specific set of m^o object as

$$\prod_{i:\theta_k^i = 0} (1 - P^D) \prod_{i:\theta_k^i \neq 0} P^D. \tag{4.37}$$

Furthermore, the probability of having m^c clutter measurements is given by $\text{Poisson}(m^c; \bar{\lambda_c})$. Omitting the time indices for brevity, we can thus formulate the prior probability for θ and m as

$$p_{\theta,m|\mathbf{X}}(\theta, m|\mathbf{X}) = \prod_{i:\theta^i = 0} (1 - P^D) \prod_{i:\theta^i \neq 0} P^D \cdot \text{Poisson}(m^c; \bar{\lambda_c}) \cdot \frac{1}{\binom{m}{m^o} m^o!}, \tag{4.38}$$

where the last factor is the probability of having a given association at a given time step, which is equivalent to 1 divided by the number of ways to select m^o detections and associate them to the corresponding objects. Using the law of total probability, we can express the measurement likelihood as the sum over all valid DAs of the product between the association conditioned measurement density and the association prior, given by

$$p_{\mathbf{Z}|\mathbf{X}}(\mathbf{Z}|\mathbf{X}) = \sum_{\theta \in \Theta} p_{\mathbf{Z},\theta|\mathbf{X}}(\mathbf{Z}, \theta|\mathbf{X}) = \sum_{\theta \in \Theta} p_{\mathbf{Z}|\mathbf{X},\theta,m}(\mathbf{Z}|\mathbf{X}, \theta, m) p_{\theta,m|\mathbf{X}}(\theta, m|\mathbf{X}). \tag{4.39}$$

If we assume that the measurements are independent when conditioning on θ and m, the conditional measurement model can be expressed as

$$p_{\mathbf{Z}|\mathbf{X},\theta,m}(\mathbf{Z}|\mathbf{X}, \theta, m) = \prod_{j:\nexists \theta^i = j} f_c(\mathbf{z}^j) \prod_{i:\theta^i \neq 0} g(\mathbf{z}^{\theta^i}|\mathbf{x}^i). \tag{4.40}$$

The full multi-object measurement likelihood at a specific time can thus be written as

$$
\begin{aligned}
p_{\mathbf{Z}|\mathbf{X}}(\mathbf{Z}|\mathbf{X}) &= \sum_{\theta \in \Theta} p_{\mathbf{Z},\theta|\mathbf{X}}(\mathbf{Z},\theta|\mathbf{X}) && (4.41)\\
&= \sum_{\theta \in \Theta} p_{\mathbf{Z}|\mathbf{X},\theta,m}(\mathbf{Z}|\mathbf{X},\theta,m) p_{\theta,m|\mathbf{X}}(\theta,m|\mathbf{X}) && (4.42)\\
&= \sum_{\theta \in \Theta} \left(\prod_{j:\nexists \theta^i = j} f_c(\mathbf{z}^j) \prod_{i:\theta^i \neq 0} g(\mathbf{z}^{\theta^i}|\mathbf{x}^i) \right) \left(\prod_{i:\theta^i=0} (1-P^D) \prod_{i:\theta^i_k \neq 0} P^D \cdot \text{Poisson}(m^c;\bar{\lambda_c}) \frac{m^c!}{m!} \right) && (4.43)\\
&= \sum_{\theta \in \Theta} \frac{\exp\{-\bar{\lambda_c}\}}{m!} \prod_{j:\nexists \theta^i = j} \lambda_c(\mathbf{z}^j) \prod_{i:\theta^i=0} (1-P^D) \prod_{i:\theta^i \neq 0} P^D g(\mathbf{z}^{\theta^i}|\mathbf{x}^i) && (4.44)\\
&= \sum_{\theta \in \Theta} \frac{\exp\{-\bar{\lambda_c}\}}{m!} \prod_{j=1}^{m} \lambda_c(\mathbf{z}^j) \prod_{i:\theta^i=0} (1-P^D) \prod_{i:\theta^i \neq 0} \frac{P^D g(\mathbf{z}^{\theta^i}|\mathbf{x}^i)}{\lambda_c\left(\mathbf{z}^{\theta^i}\right)} && (4.45)\\
&\propto \sum_{\theta \in \Theta} \prod_{i:\theta^i=0} (1-P^D) \prod_{i:\theta^i \neq 0} \frac{P^D g(\mathbf{z}^{\theta^i}|\mathbf{x}^i)}{\lambda_c\left(\mathbf{z}^{\theta^i}\right)}. && (4.46)
\end{aligned}
$$

4.3 Posterior density

The posterior density at a given time step can be expressed as

$$
p(\mathbf{X}_k|\mathbf{Z}_{1:k}) \propto p(\mathbf{X}_k) p(\mathbf{Z}_{1:k}|\mathbf{X}_k) = \sum_{\theta_k \in \Theta_k} p(\mathbf{X}_k) p(\mathbf{Z}_{1:k},\theta_k|\mathbf{X}_k). \tag{4.47}
$$

However, it is more practical to express this density as a weighted sum of densities, on the form

$$
p(\mathbf{X}_k|\mathbf{Z}_{1:k}) = \sum_{\theta_k \in \Theta_k} w_{\theta_k} p_{\theta_k}(\mathbf{X}_k), \tag{4.48}
$$

where $w_{\theta_k} = \Pr(\theta_k|\mathbf{Z}_{1:k-1})$ is a PMF, while $p_{\theta_k}(\mathbf{X}_k) = p(\mathbf{X}_k|\theta_k,\mathbf{Z}_{1:k})$ is a Probability Density Function (PDF). Observe that w_{θ_k} is the posterior probability of the DA hypothesis θ_k at time k, while $p_{\theta_k}(\mathbf{X}_k)$ is the posterior of the state matrix $\mathbf{X}_k$ conditioned on the DA hypothesis and sequence of measurements up until time k, $\mathbf{Z}_{1:k}$. Thus, summing up all these conditional posteriors over all valid DAs, the structure of the posterior density is

$$
p(\mathbf{X}_k|\mathbf{Z}_{1:k}) = \sum_{\theta_k \in \Theta_k} \Pr(\theta_k|\mathbf{Z}_{1:k-1}) p(\mathbf{X}_k|\theta_k,\mathbf{Z}_{1:k}). \tag{4.49}
$$

Now, since we at each filter recursion will use the previous posterior (a mixture density) as prior, the predicted density will also be a mixture, and we will derive the general expression of the posterior density in the case of a mixture prior, which is the most general case. The posterior will consist of a weighted sum over every possible valid DA sequence up until time k, of densities corresponding to these DA sequences. So we can write the posterior as

$$
p_{\mathbf{X}_k|\mathbf{Z}_{1:k}}(\mathbf{X}_k|\mathbf{Z}_{1:k}) = \sum_{\theta_{1:k} \in \Theta_{1:k}} w^{\theta_{1:k}} p^{\theta_{1:k}}(\mathbf{X}_k), \tag{4.50}
$$

where the notation $\sum_{\theta_{1:k} \in \Theta_{1:k}} = \sum_{\theta_1 \in \Theta_1} \cdots \sum_{\theta_k \in \Theta_k}$ constitutes a compact form of expressing the sequence of k sums over each valid DA at every time step. Suppose that the mixture prior density has the following structure

$$
p(\mathbf{X}) = \sum_h w^h p^h(\mathbf{X}) = \sum_h Pr(h) p(\mathbf{X}|h), \tag{4.51}
$$

where h is some prior DA hypothesis. Then the posterior density becomes a mixture density with one component for every combination of the hypothesis h and the DA θ, and it can be expressed as

$$\begin{aligned} p(\mathbf{X}|\mathbf{Z}) &\propto p(\mathbf{Z}|\mathbf{X})p(\mathbf{X}) && (4.52)\\ &= \left(\sum_{\theta\in\Theta} p(\mathbf{Z},\theta|\mathbf{X})\right)\left(\sum_h w^h p^h(\mathbf{X})\right) && (4.53)\\ &= \sum_h \sum_{\theta\in\Theta} w^h p(\mathbf{Z},\theta|\mathbf{X})p^h(\mathbf{X}) && (4.54)\\ &= \sum_h \sum_{\theta\in\Theta} w^h \tilde{w}^{\theta|h} \frac{p(\mathbf{Z},\theta|\mathbf{X})p^h(\mathbf{X})}{\tilde{w}^{\theta|h}} && (4.55)\\ &= \sum_h \sum_{\theta\in\Theta} \tilde{w}^{h,\theta} p^{h,\theta}(\mathbf{X}). && (4.56) \end{aligned}$$

By introducing the normalization factor $\tilde{w}^{\theta|h} = \int p(\mathbf{Z},\theta|\mathbf{X})p^h(\mathbf{X})d\mathbf{X} = p(\mathbf{Z},\theta|h)$ we have normalized each individual density conditioned on the combination of h and θ. However the resulting weights $\tilde{w}^{h,\theta}$ corresponding to these densities must be normalized as well to arrived at a normalized mixture posterior. This is done by dividing by the sum of all weights resulting from all combinations of h and θ, so that

$$p(\mathbf{X}|\mathbf{Z}) = \sum_h \sum_{\theta\in\Theta} w^{h,\theta} p^{h,\theta}(\mathbf{X}) = \sum_h \sum_{\theta\in\Theta} \Pr(h,\theta)p(\mathbf{X}|h,\theta), \tag{4.57}$$

with

$$w^{h,\theta} = \frac{\tilde{w}^{\theta|h} w^h}{\sum_h \sum_{\theta\in\Theta} \tilde{w}^{\theta|h} w^h}. \tag{4.58}$$

It can be shown that the multi-object posterior density at time k can be expressed as

$$p_{k|k}(\mathbf{X}_k) = p(\mathbf{X}_k|\mathbf{Z}_{1:k}) \propto \sum_{\theta_{1:k}\in\Theta_{1:k}} \tilde{w}_{k|k}^{\theta_{1:k}} p_{k|k}^{\theta_{1:k}}(\mathbf{X}_k), \tag{4.59}$$

where

$$\begin{aligned} \tilde{w}_{k|k}^{\theta_{1:k}} &= \prod_{i=1}^{n}\prod_{t=1}^{k} \tilde{w}^{\theta_t^i|\theta_{1:t-1}^i}, && (4.60)\\ p_{k|k}^{\theta_{1:k}}(\mathbf{X}_k) &= \prod_{i=1}^{n} p_{k|k}^{i,\theta_{1:k}^i}(\mathbf{x}_k^i). && (4.61) \end{aligned}$$

The product of un-normalized weights, calculated for each object and DA at each time step, yields the un-normalized weight $\tilde{w}_{k|k}^{\theta_{1:k}}$ corresponding to the sequence of DAs $\theta_1,\ldots,\theta_k$. The posterior for this specific DA sequence is then obtained by multiplying the individual posterior object densities. Upon normalization, the resulting posterior distribution becomes

$$p_{k|k}(\mathbf{X}_k) = \sum_{\theta_{1:k}\in\Theta_{1:k}} w_{k|k}^{\theta_{1:k}} p_{k|k}^{\theta_{1:k}}(\mathbf{X}_k) = \sum_{\theta_1\in\Theta_1} \cdots \sum_{\theta_k\in\Theta_k} \Pr[\theta_{1:k}|\mathbf{Z}_{1:k}]p(\mathbf{X}_k|\theta_{1:k},\mathbf{Z}_{1:k}), \tag{4.62}$$

with

$$w_{k|k}^{\theta_{1:k}} = \frac{\tilde{w}_{k|k}^{\theta_{1:k}}}{\sum_{\theta'_{1:k}}} \tilde{w}_{k|k}^{\theta'_{1:k}} = \frac{\prod_{i=1}^{n}\prod_{t=1}^{k} \tilde{w}^{\theta_t^i|\theta_{1:t-1}^i}}{\sum_{\theta'_1}\cdots\sum_{\theta'_k}\prod_{i=1}^{n}\prod_{t=1}^{k} \tilde{w}^{\theta_t^{i'}|\theta_{1:t-1}^{i'}}}. \tag{4.63}$$

4.4 Predicting the multi-object density

In the recursive Bayesian filtering scheme, the posterior at time step $k-1$ is used as a prior in the recursion at time step k. Assume that the posterior at time $k-1$ is expressed as a mixture density given by

$$p(\mathbf{X}_{k-1}|\mathbf{Z}_{1:k-1}) = \sum_{\theta_{1:k-1}\in\Theta_{1:k-1}} w_{k-1|k-1}^{\theta_{1:k-1}} p_{k-1|k-1}^{\theta_{1:k-1}}(\mathbf{X}_{k-1}), \tag{4.64}$$

where

- $w_{k-1|k-1}^{\theta_{1:k-1}} = P[\theta_{1:k-1}|\mathbf{Z}_{1:k-1}]$ represents the probability of the DA hypothesis sequence $\theta_{1:k-1}$ given the observed measurements up to time $k-1$.
- $p_{k-1|k-1}^{\theta_{1:k-1}}(\mathbf{X}_{k-1}) = \prod_{i=1}^{n} p_{k-1|k-1}(\mathbf{x}_{k-1}^{i}|\theta_{1:k-1}^{i}, \mathbf{Z}_{1:k-1})$ is the product of the conditional posterior densities of the n objects conditioned on the DA sequence $\theta_{1:k-1}$ at time step $k-1$.

The transition density can be written

$$p_k(\mathbf{X}_k|\mathbf{X}_{k-1}) = p_k(\mathbf{x}_k^1, ..., \mathbf{x}_k^n|\mathbf{x}_{k-1}^1, ..., \mathbf{x}_{k-1}^n). \tag{4.65}$$

By assuming that the object states evolve in time independently of each other, we can factorize (4.65) to obtain a simpler expression for the predicted density, so

$$p_k(\mathbf{X}_k|\mathbf{X}_{k-1}) = \prod_{i=1}^{n} \pi_k(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i). \tag{4.66}$$

Let h be the DA hypothesis sequence from which we predict the object states at the next time step. By the application of the Chapman-Kolmogorov equation, we can write

$$\begin{aligned}
p(\mathbf{X}_k|\mathbf{Z}_{1:k-1}) &= \int p(\mathbf{X}_k|\mathbf{X}_{k-1})p(\mathbf{X}_{k-1}|\mathbf{Z}_{1:k-1})d\mathbf{X}_{k-1} &(4.67)\\
&= \int p(\mathbf{X}_k|\mathbf{X}_{k-1}) \left[\sum_h w_{k-1|k-1}^h p_{k-1|k-1}^h(\mathbf{X}_{k-1})\right] d\mathbf{X}_{k-1} &(4.68)\\
&= \sum_h w_{k-1|k-1}^h \int p(\mathbf{X}_k|\mathbf{X}_{k-1})p_{k-1|k-1}^h(\mathbf{X}_{k-1})d\mathbf{X}_{k-1} &(4.69)\\
&:= \sum_h w_{k|k-1}^h p_{k|k-1}^h(\mathbf{X}_k). &(4.70)
\end{aligned}$$

This is nothing else but the weighted sum of the standard density predictions over all valid DA sequences, and the weights remain unchanged after prediction. With the linear and Gaussian assumption, we have that

$$p(\mathbf{X}_{k-1}|\mathbf{Z}_{1:k-1}) = \sum_h w_{k-1|k-1}^h \prod_{i=1}^{n} \phi\left(\mathbf{x}_{k-1}^i; \boldsymbol{\mu}_{k-1|k-1}^{i,h}, \mathbf{P}_{k-1|k-1}^{i,h}\right), \tag{4.71}$$

$$p_k(\mathbf{X}_k|\mathbf{X}_{k-1}) = \prod_{i=1}^{n} \phi\left(\mathbf{x}_k^i; \mathbf{A}\mathbf{x}_{k-1}^i, \mathbf{Q}\right), \tag{4.72}$$

so that the predicted density becomes

$$p(\mathbf{X}_k|\mathbf{Z}_{1:k-1}) = \sum_h w_{k|k-1} \prod_{i=1}^{n} \phi\left(\mathbf{x}_k^i; \boldsymbol{\mu}_{k|k-1}^{i,h}, \mathbf{P}_{k|k-1}^{i,h}\right), \tag{4.73}$$

with

- $w_{k|k-1}^h = w_{k-1|k-1}^h$,
- $\boldsymbol{\mu}_{k|k-1}^{i,h} = \texttt{predict}(\boldsymbol{\mu}_{k-1|k-1}^{i,h})$,
- $\mathbf{P}_{k|k-1}^{i,h} = \texttt{predict}(\mathbf{P}_{k-1|k-1}^{i,h})$,

where the function `predict()` performs the usual KF prediction on the state vector and corresponding covariance matrix, for object i and the given DA sequence h, as explained in (4.11) and (4.12).

4.5 Data association as optimization problem

The posterior density of the state of n objects is a mixture density. The increase of the number of mixture components occur at every time step, due to the presence of unknown DAs. In particular, assume we have n objects and m measurements at time k, where $m^o \in \{0, 1, ..., \min(m, n)\}$ measurements originate from objects. There are $\binom{n}{m^o}$ ways to choose m^o objects from n objects, $\binom{m}{m^o}$ ways to choose m^o detections from m detections and $m^o!$ ways to assign

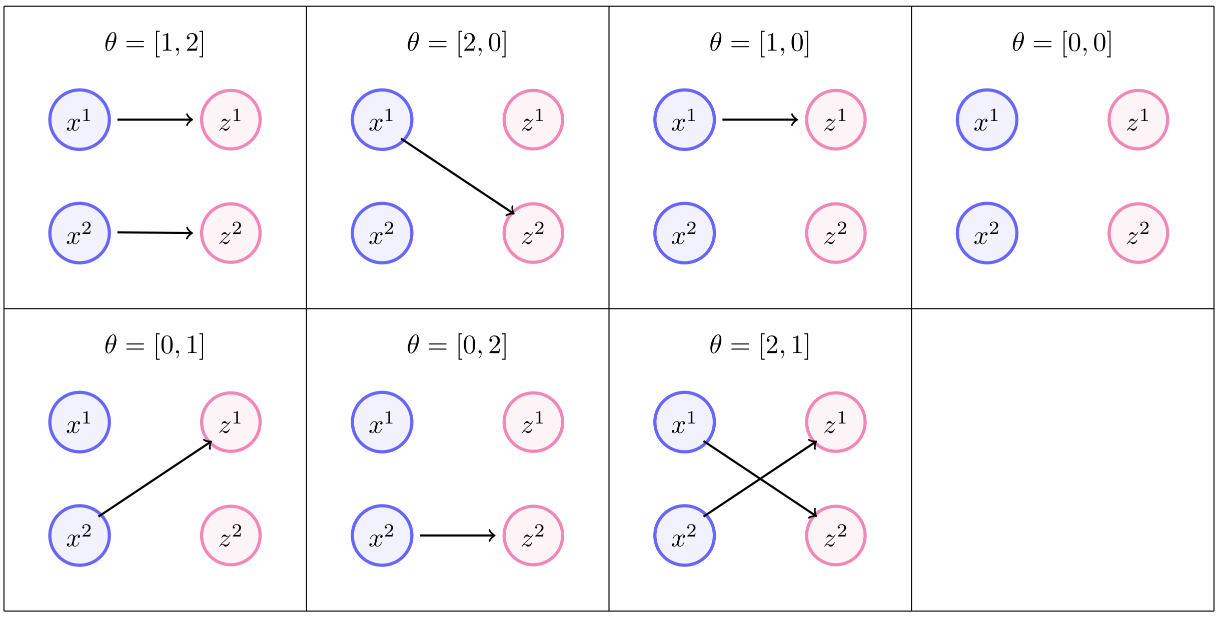


Figure 4.3: Illustration of DAs between two objects and two measurements.

the chosen detections and the chosen objects, since the order of assignment matters. The total number of ways to associate m measurements to n objects is then

$$N_A(m,n) := \sum_{m^o=0}^{\min(m,n)} \binom{n}{m^o}\binom{m}{m^o} m^o!. \tag{4.74}$$

The expression above only accounts for one time step. Assuming that the initial prior mixture density has N_0 components, the number of mixture components at time k becomes

$$N_0 \prod_{t=1}^{k} N_A(m_t, n). \tag{4.75}$$

We now illustrate the extreme rate by which the number of mixture components grow when we increase both number of objects involved and propagate in time. Assume that we track a constant number of objects for $k = 3$ time steps, and that $m_1 = 4$, $m_2 = 6$ and $m_3 = 7$. Assuming that the initial prior has $N_0 = 1$ component, we get that for

- $n = 1$ object: $(1 + 4)(1 + 6)(1 + 7) = 280$ posterior mixture components,
- $n = 2$ objects: $N_A(4, 2)N_A(6, 2)N_A(7, 2) = 21 \cdot 43 \cdot 57 = 51471$ components,
- $n = 3$ objects: $N_A(4, 3)N_A(6, 3)N_A(7, 3) = 73 \cdot 229 \cdot 358 = 5984686$ components.

Figure 4.3 illustrates a toy example for the number of valid DAs we can have with 2 objects and 2 measurements at a given time step. Given the explosive growth of mixture components, measures to reduce this growth must be implemented to preserve an acceptable amount of computational resources being spent in a tracking application.

In particular, we aim to find a subset of DA hypotheses $\tilde{\Theta}_k \subset \Theta_k$ such that $|\tilde{\Theta}_k| << |\Theta_k|$. Since the DA variable length is equal to the number of objects at a given time step, the hypothesis reduction problem can be posed as a 2D assignment problem, where we compute an optimal subset of Θ_k such that some cost is minimized.

The 2D assignment problem is a commonly known combinatorial optimization problem (Pierskalla, 1968). We have a set of n workers denoted as $w_1, ..., w_n$ and a set of m tasks denoted as $t_1, ..., t_m$. Each worker must be assigned to exactly one task, and each task can be assigned to only one worker. There is a cost associated with every worker-task assignment pair, and the objective is the minimization of the total cost by identifying the optimal set of worker-task pairs. The $n \times m$ cost matrix takes form

$$L = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1m} \\ c_{21} & c_{22} & \dots & c_{2m} \\ \vdots & \vdots & & \vdots \\ c_{n1} & c_{n2} & \dots & c_{nm} \end{bmatrix}, \tag{4.76}$$

where $L_{ij} = c_{ij}$ represents the cost of assigning worker i to task j. The $n \times m$ assignment matrix has elements according to the following rule:

$$A_{ij} = \begin{cases} 1 & \text{if } w^i \text{ is assigned to } t^j, \\ 0 & \text{otherwise.} \end{cases} \tag{4.77}$$

The total cost of assigning n workers can now be computed as

$$C = \sum_{i=1}^{n} \sum_{j=1}^{m} A_{ij} L_{ij} = \text{trace}(A^T L). \tag{4.78}$$

The 2D assignment problem can thus be formulated as

$$\min \text{trace}(A^T L), \qquad \text{subject to } A_{ij} \in \{0,1\} \ \forall i,j, \quad \sum_{j=1}^{m} A_{ij} = 1 \ \forall i, \quad \sum_{i=1}^{n} A_{ij} = 1 \ \forall j. \tag{4.79}$$

As derived in Section 4.3, the multi-object density for a given time step can be expressed as

$$p(\mathbf{X}) \propto \sum_{h} \sum_{\theta \in \Theta} \tilde{w}^{\theta|h} w^h p^{h,\theta}(\mathbf{X}). \tag{4.80}$$

Let h be some DA hypothesis sequence. The optimal DA $\theta^* \in \Theta$ must possess the highest weight when compared to the weights of all other DAs in the same set Θ at a specific time step. That is,

$$\tilde{w}^{\theta^*|h} \geq \tilde{w}^{\theta|h} \forall \theta \in \Theta. \tag{4.81}$$

The objective in question is to find this optimal DA

$$\theta^* = \arg\max_{\theta \in \Theta} \tilde{w}^{\theta|h} = \arg\max_{\theta \in \Theta} \prod_{i=1}^{n} \tilde{w}^{\theta^i|h}. \tag{4.82}$$

We apply a logarithmic transformation to the weights for computational conveniences. The maximization of the logarithm of the weights will yield the same optimum as the maximization of the weights directly. So

$$\theta^* = \arg\max_{\theta \in \Theta} \log \left(\prod_{i=1}^{n} \tilde{w}^{\theta^i|h} \right) \tag{4.83}$$

$$= \arg\min_{\theta \in \Theta} \sum_{i=1}^{n} -\log \left(\tilde{w}^{\theta^i|h} \right). \tag{4.84}$$

The shape of the assignment matrix A in this case will be $n \times (m+n)$, if we assume that at a given time step we are tracking n objects, and have m detections and n misdetections. The assignment matrix takes values according to the following logic:

- If object i is detected, we have that $\theta^i = j \implies A^{i,j} = 1$,
- If object i is not detected, we have that $\theta^i = 0 \implies A^{i,m+i} = 1$.

This gives rise to the following general structure

$$A = \left[\begin{array}{cccc|cccc} A^{1,1} & A^{1,2} & \dots & A^{1,m} & A^{1,m+1} & 0 & \dots & 0 \\ A^{2,1} & A^{2,2} & \dots & A^{2,m} & 0 & A^{2,m+2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ A^{n,1} & A^{n,2} & \dots & A^{n,m} & 0 & 0 & \dots & A^{n,m+n} \end{array} \right]. \tag{4.85}$$

Here, the left $n \times m$ sub-matrix correspond to objects being detected with some corresponding association, while the diagonal of the right $n \times n$ sub-matrix correspond to objects being misdetected. Every object must either be detected or misdetected, and any detection or misdetection can be assigned to either no object or one object, that is

$$\sum_{j} A^{i,j} = 1 \quad \forall i, \tag{4.86}$$

$$\sum_{i} A^{i,j} \in \{0,1\} \quad \forall j. \tag{4.87}$$

By using the derived measurement likelihood in (4.46), we can express the un-normalized weights for a given DA hypothesis h as

$$\tilde{w}^{\theta^i|h} = \int p(\mathbf{Z}, \theta^i|\mathbf{x}^i)p^{i,h}(\mathbf{x}^i)d\mathbf{x}^i = \begin{cases} \int (1-P^D)p^{i,h}(\mathbf{x}^i)d\mathbf{x}^i & \text{if } \theta^i = 0, \\ \int \frac{P^D g_k(\mathbf{z}^{\theta^i}|\mathbf{x}^i)}{\lambda_c(\mathbf{z}^{\theta^i})}p^{i,h}(\mathbf{x}^i)d\mathbf{x}^i & \text{if } \theta^i \neq 0. \end{cases} \tag{4.88}$$

Furthermore, we have that

$$\ell^{i,0,h} = \log\left(\int (1-P^D)p^{i,h}(\mathbf{x}^i)d\mathbf{x}^i\right), \tag{4.89}$$

$$\ell^{i,j,h} = \log\left(\int \frac{P^D g_k(\mathbf{z}^j|\mathbf{x}^i)}{\lambda_c(\mathbf{z}^j)}p^{i,h}(\mathbf{x}^i)d\mathbf{x}^i\right), \tag{4.90}$$

are the log-likelihoods of misdetecting $\mathbf{x}^i$ and of associating state $\mathbf{x}^i$ to the measurement $\mathbf{z}^j$, respectively. Assuming linear transition models and Gaussian state densities, we can write

$$\ell^{i,0,h} = \log\bigl(1-P^D\bigr), \tag{4.91}$$

$$\ell^{i,j,h} = \log\left(\frac{P^D V}{\bar{\lambda}_c}\right) - \frac{1}{2}\log\left(\det\left(2\pi\mathbf{S}^{i,h}\right)\right) - \frac{1}{2}\left(\mathbf{z}^j - \hat{\mathbf{z}}^{i,h}\right)^T\left(\mathbf{S}^{i,h}\right)^{-1}\left(\mathbf{z}^j - \hat{\mathbf{z}}^{i,h}\right). \tag{4.92}$$

Here $p^{i,h}(\mathbf{x}^i) = \phi\left(\mathbf{x}^i; \boldsymbol{\mu}^{i,h}, \mathbf{P}^{i,h}\right)$ denotes the prior density for the state of object i given a DA hypothesis h, while $\hat{\mathbf{z}}^{i,h} = \mathbf{H}\boldsymbol{\mu}^{i,h}$ and $\mathbf{S}^{i,h} = \mathbf{H}\mathbf{P}^{i,h}\mathbf{H}^T + \mathbf{R}$ are the predicted measurement and innovation covariance for object i respectively, given the DA hypothesis h. The cost associated with the DA θ is then given by

$$\sum_{i=1}^{n} -\log\Bigl(\tilde{w}^{\theta^i|h}\Bigr) = \sum_{i=1}^{n}\sum_{j=1}^{m} A^{i,j}(-\ell^{i,j,h}) + \sum_{i'=1}^{n} A^{i',m+i'}(-\ell^{i',0,h}). \tag{4.93}$$

We define the cost matrix as

$$L^h = \left[\begin{array}{cccc|cccc} -\ell^{1,1,h} & -\ell^{1,2,h} & \dots & -\ell^{1,m,h} & -\ell^{1,0,h} & \infty & \dots & \infty \\ -\ell^{2,1,h} & -\ell^{2,2,h} & \dots & -\ell^{2,m,h} & \infty & -\ell^{2,0,h} & \dots & \infty \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -\ell^{n,1,h} & -\ell^{n,2,h} & \dots & -\ell^{n,m,h} & \infty & \infty & \dots & -\ell^{n,0,h} \end{array}\right], \tag{4.94}$$

where the left $n \times m$ submatrix corresponds to the costs of object detections, while the diagonal $n \times n$ matrix represents the misdetection costs. The off-diagonal entries of this submatrix is set to infinity to ensure unique object assignments. The cost can now be expressed more compactly as

$$\sum_{i=1}^{n} -\log\Bigl(\tilde{w}^{\theta^i|h}\Bigr) = \sum_{i=1}^{n}\sum_{j=1}^{m+n} A^{i,j}L^{i,j,h} = \operatorname{trace}(A^T L^h). \tag{4.95}$$

With a slight modification to (4.79), we can write the optimal assignment problem as

$$\begin{gathered} \min \operatorname{trace}(A^T L), \quad \text{subject to } A_{ij} \in \{0,1\},\ \forall i,j \in \{1,...,n\} \times \{1,...,n+m\}, \\ \sum_{j=1}^{n+m} A^{i,j} = 1\ \ \forall i \in \{1,...,n\}, \quad \sum_{i=1}^{n} A^{i,j} \in \{0,1\}\ \ \forall j \in \{1,...,n+m\}. \end{gathered} \tag{4.96}$$

There are a number of algorithms that solves this kind of optimization problem. For finding the unique best solution θ^* the Hungarian (Kuhn, 1955) and Auction (Bertsekas, 1979) algorithm can be used. If we on the other hand want to find the set of the M best solutions, i.e. the solutions $A^{*1}, ..., A^{*M}$ such that

$$\tilde{w}^{\theta^{*1}|h} \geq ... \geq \tilde{w}^{\theta^{*M}|h} \geq \tilde{w}^{\theta|h}, \ \ \forall \theta \in \Theta \setminus \{\theta^{*m}\}_{m=1}^{M}, \tag{4.97}$$

Murty's algorithm (Murty, 1968) can be used for this. We will later see that the JPDA algorithm uses multiple DAs to arrive at informed state estimations, and therefore this approach will be the most relevant in this thesis.

4.6 Gating in MOT

As a pre-processing step even before DA optimization, gating can be used to rule out highly unlikely detections. It works by forming a probabilistic gate around the neighborhood of the predicted measurement that at a certain significance

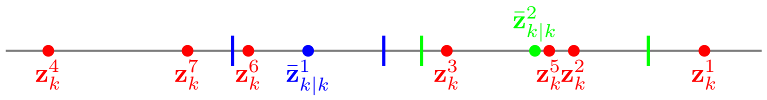


Figure 4.4: Illustration of the gating idea on a one-dimensional domain. The red dots correspond to measurement at time k, while the blue and green are predicted hypotheses, with corresponding gates.

level excludes those measurements falling outside the gate. We do this so that we avoid computing state predictions and covariance matrices for DAs whose weights are essentially zero. Since we are working with measurements of 1D position of objects, the gate will be an interval on the real line. For measurements of higher dimensions, this gate can take other geometrical shapes, depending on the distance measure used. In the Gaussian case this would be an ellipse in 2D and an ellipsoid in 3D, but still just an interval in 1D. Figure 4.4 is an illustration of the gating idea; there are two predicted hypotheses: $h_{k-1} = 1$ and $h_{k-1} = 2$. The gates for these two predicted hypotheses are illustrated by the colored intervals with the predicted measurements in the center. The measurements that are inside either of the gates are kept, which means that measurements $\mathbf{z}^4_k$, $\mathbf{z}^7_k$ and $\mathbf{z}^1_k$ are discarded as clutter at time k.

We found that the log-likelihood of the association between the state of object i and the measurement $\mathbf{z}^j$ under a specific DA hypothesis (sequence) h was given by (4.92). From this we can derive the Mahalanobis distance between the measurement $\mathbf{z}^j$ and the predicted detection of object i under hypothesis h at time step k as

$$d^2_{i,j,h_{k-1}} = \left(\mathbf{z}^j_k - \bar{\mathbf{z}}^{i,h_{k-1}}_{k|k-1}\right)^T \left(\mathbf{S}^{i,h_{k-1}}_k\right)^{-1} \left(\mathbf{z}^j_k - \bar{\mathbf{z}}^{i,h_{k-1}}_{k|k-1}\right), \tag{4.98}$$

where $\mathbf{S}^{i,h_{k-1}}_k$ is the innovation covariance from earlier. Note that the larger this distance is, the smaller the log-likelihood for the respective association will be. Let G be the gating threshold we set to determine the size of the gate. If

$$d^2_{i,j,h_{k-1}} \leq G, \tag{4.99}$$

then $\mathbf{z}^j_k$ is a possible detection of object i. Otherwise we discard that DA under the hypothesis h, and we set $\ell^{i,j,h_{k-1}} = -\infty$. The gating threshold G can be selected by various methods (Wang et al., 2008), like the Chi-square estimation method. If G is sufficiently small, we risk having object measurements fall outside the gate. For a general association value at time k for object i, θ^i_k, this probability can be formulated as

$$P_G = \Pr(d^2_{i,\theta^i_k,h_{k-1}} < G | h_{k-1}, \theta^i_k). \tag{4.100}$$

The Chi-square estimation method suggests that

$$d^2_{i,\theta^i_k,h_{k-1}} | h_{k-1}, \theta^i_k \sim \chi^2(d_z), \tag{4.101}$$

where $d_z = 1$ is the spatial dimension of the measurements. Here, P_G is set to a desired confidence level, and the G can thus be found by using the CDF of $\chi^2(d_z)$.

4.7 The JPDA algorithm

JPDA presents a sophisticated approach to the complex problem of MOT. When it comes to tracking multiple targets simultaneously, as earlier seen, one of the most pressing challenges is the accurate association of measurements with the corresponding targets. The JPDA algorithm tackles this problem by introducing a probabilistic framework that allows for the simultaneous consideration of multiple potential DAs. Unlike other tracking methods that rely on strict associations, like the Global Nearest Neighbor filter (GNN), JPDA (Bar-Shalom & Li, 1995) accounts for uncertainty, and leverages the power of probability to estimate the likelihood of each association, effectively handling the growing complexity of unknown DAs. In this section we describe the algorithm, and the concepts of track initiation and deletion.

We found that the exact normalized posterior density of the states can be expressed as

$$p_{k|k}(\mathbf{X}_k) = \sum_{\theta_{1:k} \in \Theta_{1:k}} w^{1:k}_{k|k} p^{\theta_{1:k}}_{k|k}(\mathbf{X}_k). \tag{4.102}$$

We now wish to approximate this mixture density by a single density given by

$$p^{JPDA}_{k|k}(\mathbf{X}_k) = p^{\beta_{1:k}}_{k|k}(\mathbf{X}_k), \tag{4.103}$$

where $\beta_{1:k}$ is defined to be the sequence of marginal association probabilities

$$\beta_1,\ \ \beta_2|\beta_1,\ \ \beta_3|\beta_2,\beta_1,\ \ \dots\ \ ,\beta_k|\beta_{1:k-1}. \tag{4.104}$$

The JPDA density can be parameterized by the individual object state densities

$$p_{k|k}^{i,\beta_{1:k}}(\mathbf{x}_k^i),\ \ i=1,...,n_k. \tag{4.105}$$

Now we introduce a time-varying number of objects by time-indexing n. The mechanisms of initiation and deletion of objects and how they are incorporated into JPDA are discussed later. Let the marginal probability that object i is associated with measurement $\mathbf{z}_k^j$ at time k be given by

$$\beta_k^{i,j}=\Pr[\theta_k^i=j|\mathbf{Z}_{1:k-1}]=\sum_{\theta_k\in\Theta_k:\theta_k^i=j} w^{\theta_k}\propto\sum_{\theta_k\in\Theta_k:\theta_k^i=j}\tilde{w}^{\theta_k}, \tag{4.106}$$

and the probability that object i was undetected be given by

$$\beta_k^{i,0}=\Pr[\theta_k^i=0|\mathbf{Z}_{1:k-1}]=1-\sum_j\beta_k^{i,j}=\sum_{\theta_k\in\Theta_k:\theta_k^i=0} w^{\theta_k}\propto\sum_{\theta_k\in\Theta_k:\theta_k^i=0}\tilde{w}^{\theta_k}. \tag{4.107}$$

It is clear that the computation of all marginal association probabilities requires the summation over the set of all valid DAs at each time step. As discussed earlier, this often is highly unfeasible, and therefore we can compute the M best associations $\theta_k^{*1:M}$, and approximate the probabilities as a remedy. The probabilities can thus be approximated as

$$\beta_k^{i,j}\approx\frac{\sum_{\theta_k\in\theta_k^{*1:M}:\theta_k^i=j}\tilde{w}^{\theta_k}}{\sum_{\theta_k\in\theta_k^{*1:M}}\tilde{w}^{\theta_k}}. \tag{4.108}$$

The higher we select M, the more accurate the probability estimates will be, at the expense of increased computational demand. In the JPDA update step, we uses as prior the predicted density of the states, given by

$$p_{k|k-1}(\mathbf{X}_k)=\prod_{i=1}^{n_k}p_{k|k-1}^i(\mathbf{x}^i). \tag{4.109}$$

The marginal posterior density for object i i determined by the linear combination of all marginal association probabilities and their corresponding updated density, given by

$$p_{k|k}^i(\mathbf{x}^i)=\beta_k^{i,0}p_{k|k-1}^i(\mathbf{x}^i)+\sum_{j=1}^{m_k}\beta_k^{i,j}p_{k|k-1}^{i,j}(\mathbf{x}^i), \tag{4.110}$$

where we have let $p_{k|k-1}^{i,j}(\mathbf{x}^i)$ denote the posterior density of the state of object i when we use measurement $\mathbf{z}_k^j$ to update the predicted density $p_{k|k-1}^i(\mathbf{x}^i)$. It is clear that (4.110) is a mixture density. Therefore, by applying the method of moments, we merge the density components of the mixture into a single density, and we can write the merged marginal posterior for object i as

$$p_{k|k}^{i,\beta}(\mathbf{x}_k^i)=\texttt{merge}\left(\beta_k^{i,0}p_{k|k-1}^i(\mathbf{x}^i)+\sum_{j=1}^{m_k}\beta_k^{i,j}p_{k|k-1}^{i,j}(\mathbf{x}^i)\right), \tag{4.111}$$

where `merge()` computes linear combinations of the mean vectors and covariance matrices resulting from each of the mixture component densities, so that we end up with one mean vector and one covariance matrix, parameterizing the approximated marginal JPDA posterior for object i. The multi-object JPDA approximated posterior then becomes

$$p_{k|k}^{\beta}(\mathbf{X}_k)=\prod_{i=1}^{n_k}p_{k|k}^{i,\beta}(\mathbf{x}_k^i). \tag{4.112}$$

Algorithm 3 describes the main steps of the JPDA filter. Making the same assumptions for the distributions and transition models as in the KF, the prediction and update steps in Algorithm 3 becomes:

- Prediction:

$$\boldsymbol{\mu}_{k|k-1}^i=\mathbf{A}\boldsymbol{\mu}_{k-1|k-1}^i, \tag{4.113}$$

$$\mathbf{P}_{k|k-1}^i=\mathbf{A}\mathbf{P}_{k-1|k-1}^i\mathbf{A}^T+\mathbf{Q}. \tag{4.114}$$

Algorithm 3 The JPDA filter update

```
for $k = 1, ..., T$ do
    Prediction:
    for $i = 1, ..., n_k$ do
        $p^{i,\beta_{1:k-1}}_{k|k-1}(\mathbf{x}^i_k) = \int p(\mathbf{x}^i_k|\mathbf{x}^i_{k-1}) p^{i,\beta_{1:k-1}}_{k-1|k-1}(\mathbf{x}^i_{k-1}) d\mathbf{x}^i_{k-1}$
    end for
    Update:
    Compute $\beta^{i,j}_k$
    for $i = 1, ..., n_k$ do
        $p^{i,\beta_{1:k}}_{k|k}(\mathbf{x}^i_k) = \texttt{merge}\left(\beta^{i,0}_k p^{i,\beta_{1:k-1}}_{k|k-1}(\mathbf{x}^i) + \sum_{j=1}^{m_k} \beta^{i,j}_k p^{i,\beta_{1:k},j}_{k|k-1}(\mathbf{x}^i)\right)$
    end for
end for
```

- Update mean:

$$\varepsilon^{i,j}_k = \mathbf{z}^j_k - \mathbf{H}\mu^i_{k|k-1}, \tag{4.115}$$

$$\varepsilon_k = \sum_{j=1}^{m_k} \beta^{i,j}_k \varepsilon^{i,j}_k, \tag{4.116}$$

$$\mu^i_{k|k} = \mu^i_{k|k-1} + \mathbf{K}^i_k \varepsilon_k. \tag{4.117}$$

- Update covariance:

$$\bar{\mathbf{P}}^i_k = \mathbf{P}^i_{k|k-1} - \mathbf{K}^i_k \left(\mathbf{H}\mathbf{P}^i_{k|k-1}\mathbf{H}^T + \mathbf{R}\right) \left(\mathbf{K}^i_k\right)^T, \tag{4.118}$$

$$\tilde{\mathbf{P}}^i_k = \mathbf{K}^i_k \left(\left(\sum_{j=1}^{m_k} \beta^{i,j}_k \varepsilon^{i,j}_k \left(\varepsilon^{i,j}_k\right)^T \right) - \varepsilon_k \varepsilon_k^T \right) \left(\mathbf{K}^i_k\right)^T, \tag{4.119}$$

$$\mathbf{P}^i_{k|k} = \beta^{i,0}_k \mathbf{P}^i_{k|k-1} + (1 - \beta^{i,0}_k)\bar{\mathbf{P}}^i_k + \tilde{\mathbf{P}}^i_k, \tag{4.120}$$

with $\mathbf{K}^i_k = \mathbf{P}^i_{k|k-1}\mathbf{H}^T \left(\mathbf{H}\mathbf{P}^i_{k|k-1}\mathbf{H}^T + \mathbf{R}\right)$.

4.7.1 Initiation and deletion

In the world of MOT, keeping tabs on objects as they enter and leave the FOV V is an essential challenge. Since we are working with DAS data from a restricted road segment at Selsbakk, this gives rise to the need for automatic initiation of tracks as vehicles enter the road segment, and the equivalent deletion of tracks as the vehicles passes the road segment. These are functions whose task will be to make sure we keep track of the right object at the right times, especially when objects enter and exit the FOV. Figure 4.5 illustrates the setup for the FOV in our tracking application. V_a and V_b marks the lower and upper spatial bounds for the FOV, and is equivalent to the road segment we receive DAS data from. Furthermore, we denote by δ a certain range into the FOV from the boundaries, in which initiation of new tracks can happen. That is, new tracks can only be initiated in the green areas, which we will call the initiation field. Tracks can exist in both the green and yellow areas. Outside the FOV, namely the red areas in Figure 4.5, there will be no recording of measurements, and therefore no new tracks can be initiated here.

In this application we will base the initiation of new tracks on measurements occurring in the initiation field. For every time step, the initiator will look for measurements falling into either of the initiation fields, and it will establish a set of potential candidate tracks, called *holding* tracks. For every time step, the tracks will be propagated according to the JPDA algorithm with Kalman updates and predictions. If a holding track exceeds a minimum number of valid updates from measurements N_{init} in the initiation field, the track is added to the set of confirmed tracks, and a new track is born. This minimum threshold is a hyperparameter set by the user, and can greatly influence the tracking applications ability to recognize new tracks. This will also be influenced by how large we set δ. The wider we set the initiation field, the more chances the tracker will have to initiate new tracks, while the risk of creating tracks from clutter also increases.

Every new track needs a prior density for the object state. Therefore, as a reasonable solution, every new holding track will have prior state density given by

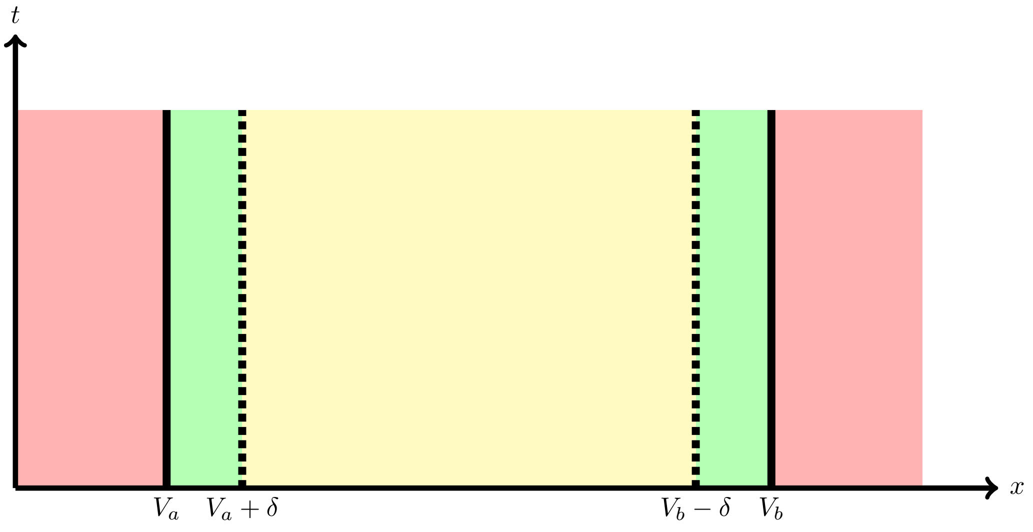


Figure 4.5: 1D FOV and initiation field (green).

$$\mathbf{x}_k = \begin{cases} \begin{bmatrix} \texttt{mean}(\mathbf{Z}_k) \\ v \end{bmatrix}, & \text{if } \mathbf{Z}_k \in [V_a, V_a+\delta], \\ \begin{bmatrix} \texttt{mean}(\mathbf{Z}_k) \\ -v \end{bmatrix}, & \text{if } \mathbf{Z}_k \in [V_b-\delta, V_b], \end{cases} \quad \text{and} \quad \mathbf{P}_k = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix}. \tag{4.121}$$

Here, $\texttt{mean}(\mathbf{Z}_k)$ denotes the average positional measurement from the set of measurements inside either initiation field at time k, while v, σ_1 and σ_2 are values set by the user based on e.g. knowledge about the typical kinetic behavior of objects in the FOV. Typically $|\mathbf{Z}_k| \sim 1$ for a vehicle moving with large distance to neighboring vehicles, and given that the clutter rate is reasonably low.

When it comes to the deletion of tracks, there are various approaches one can take. In this application we have chosen to delete tracks based on the following conditions:

- If $\boldsymbol{\mu}_{k|k-1}[1] < V_a$ or $\boldsymbol{\mu}_{k|k-1}[1] > V_b$, delete the track.
- If $\text{trace}(\mathbf{P}_{k|k-1}) > \sigma_{thres}$, delete the track.

So, if the predicted position of the object falls outside the FOV or that the predicted covariance of the state exceeds a certain threshold, the object track is deleted. Similarly to N_{init}, σ_{thres} is chosen by the user, and can greatly affect the trackers ability to maintain tracks during periods of few updates, i.e. in areas with a low measurement density, if not chosen appropriately.

4.8 Simulation example

In this section we will based on a simulation setup demonstrate the importance of appropriate choices of essential hyperparameters in the tracking application. The findings in this section will influence the hyperparameter choices made in the next chapter, when we track real vehicles from real DAS data. Table 4.1 lists the different parameters related to the generation of ground truth trajectories. Furthermore, Table 4.2 shows the parameter configuration for the generation of measurements based on the ground truth trajectories. Table 4.3 contains the parameter choices related to initiation and deletion of tracks.

4.8.1 Process noise covariance Q

We have used a certain value for σ_q for the generation of ground truth object trajectories, and the kinetic behavior of the objects are directly influenced by this parameter. The higher we set σ_q, the more variability the dynamical system will experience, leading to higher fluctuations in velocity, and decreased predictability in both position and velocity. We aim to simulate vehicles driving at near constant speeds with an overall high state predictability. Therefore, it is reasonable to select a low value for σ_q in the generation of ground truth data.

Table 4.1: Parameter configuration related to ground truth generation

Parameter Name	Value	Explanation
σ_q	0.05	Process noise parameter used to generate ground truth.
T	200	Number of simulation steps.
V_a	0	Start of the FOV.
V_b	500	End of the FOV.
$birth\_rate$	0.05	Birth rate for new objects.
$death\_probability$	0	Probability of object death.
Δt	1	Time step size.

Table 4.2: Parameter configuration related to measurement generation

Parameter Name	Value	Explanation
$\bar{\lambda}$	2	Global clutter rate for the simulation.
σ_r	0.05	Measurement noise parameter.
P^D	0.90	Probability of detection for the tracking system.

When it comes to the construction of a Kalman predictor of the dynamical system, it might be beneficial to use a different value for σ_q. In particular, choosing a slightly higher σ_q (compared to the one used in generation of ground truth data) could help the filter deal with factors like imperfect modeling, external disturbances or other uncertainties that aren't fully captured in the idealized ground truth. In other words, a higher σ_q in the Kalman predictor allows the filter to be more flexible and adapt to unexpected situations, leading to a more robust estimation process. That said, if the noise level is set too high, the filter will have less confidence in the predictions. It might become too flexible, and adapts too quickly to changes in the system as it expects larger variations. This might lead to overfitting, meaning that the filter mistakes short-term fluctuations for actual changes in the underlying system. Overfitting can make the filter start tracking clutter, and therefore eventually loose track of the object. Since we are working with a covariance-based deleter, using a too high σ_q might cause a track to be deleted as consequence of an update based on a noisy measurement, leading to a huge increase in prediction uncertainty. So while the filter thought the measurement was clutter and deleted the track, in reality it was a noisy object detection.

On the other hand, when σ_q is set too low, the filter has a higher confidence in its predictions, since it assumes that the actual system follows the predicted model. The filter therefore becomes less responsive to sudden changes or accelerations in the system dynamics. So, a too low σ_q can lead to underfitting, meaning that the filter adapts too slow to actual changes in the system, since it is less influenced by the measurements. Therefore, selecting an appropriate value for σ_q in the Kalman predictor is crucial for the tracking performance.

4.8.2 Measurement noise covariance R

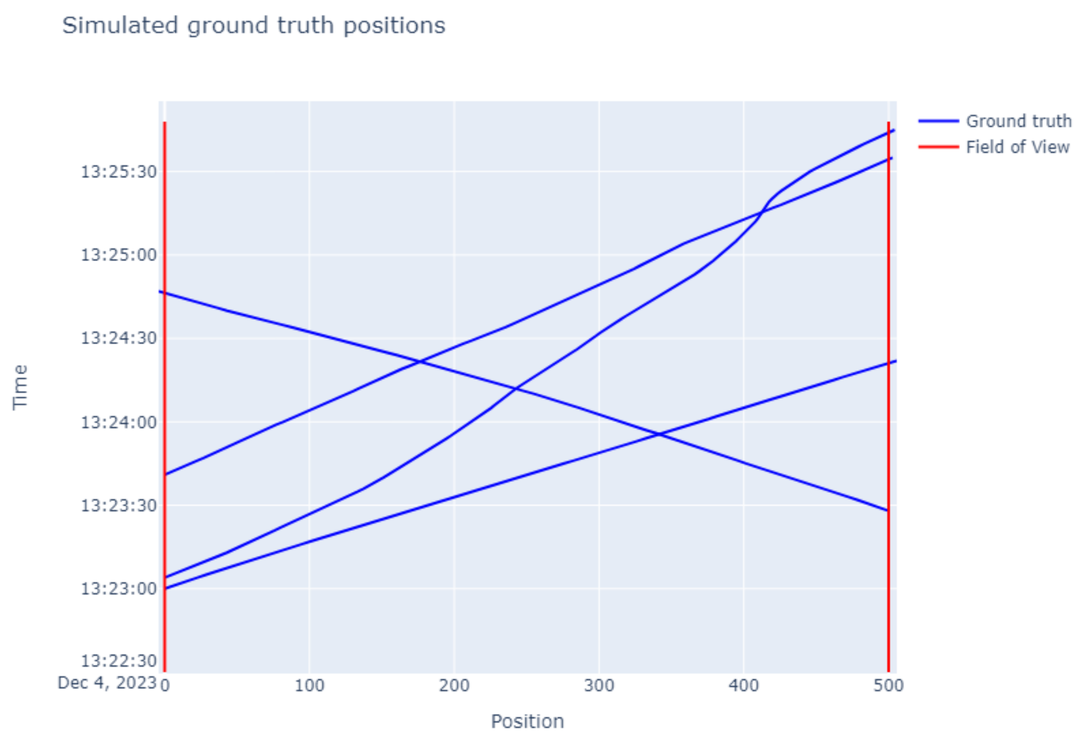
The measurement noise covariance parameter σ_r measures the trust the filter has in the measurements. We used a certain value for σ_r in the generation of measurements, and since we assume that we have a decent sensor quality, this noise level was set fairly low, so that the object detections don't deviate too much from the ground truth positions.

When it comes to the construction of a Kalman updater, it is important to choose a σ_r that reflects the level of noise that is observed in the measurements. If σ_r is set too high, it means that the filter has less confidence in the measurements, making it rely more on the predictions. So the filter inclines towards trusting its internal model, rather than noisy sensor data. However, if set excessively high, the filter might ignore or downplay the significance of measurements. With a high σ_r the Kalman gain becomes small, and thus the corrections to predictions the measurements make becomes smaller the higher σ_r is set.

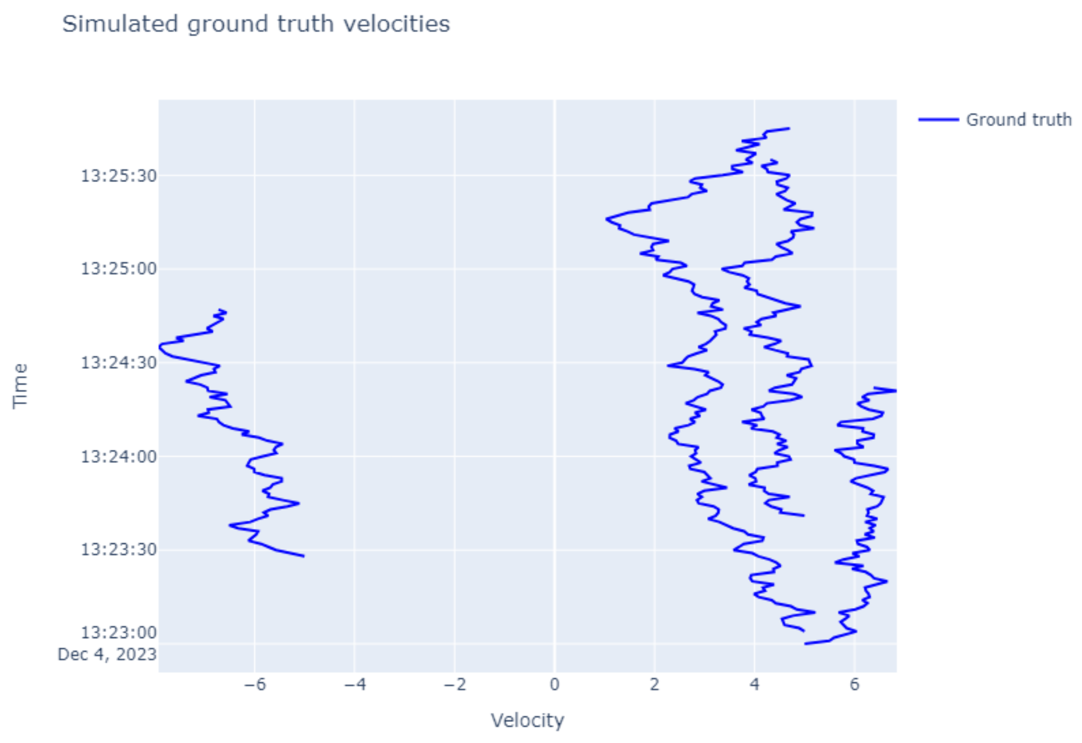
A low σ_r implies higher confidence in the measurements. The filter becomes more willing to adjust its predictions based on the incoming measurements. A too low σ_r can make the filter oscillate between measurements, and essentially ignore the predictions. In other words, the filter might produce less stable and more erratic state estimates due to the increased impact of measurement noise.

In the following, we show the practical consequences in terms of tracking performance when we choose a too high or too low σ_q (with a reasonable σ_r). After experimenting with different values of σ_r, we found that the tracking performance was very little affected by this parameter, in comparison to when we adjusted σ_q, so we omit plots for these scenarios. A possible reason for this is that the measurements and clutter are generated independently of each other. This does not however mean that it shouldn't be carefully tuned for optimal tracking performance in the next chapter, where the measurements and clutter will have higher dependence.

Figure 4.6 shows the simulated object positions and velocities when using the parameter setting from Table 4.1, together with the FOV. Figure 4.7 shows the simulated measurements based on the ground truth object states using



(a) Ground truth object positions.



(b) Ground truth object velocities.

Figure 4.6: Simulated ground truth object states based on the parameter setting from Table 4.1.

Table 4.3: Parameter configuration related to initiation and deletion of tracks

Parameter Name	Value	Explanation
δ	20	Initiation range.
σ_{thres}	50	Covariance threshold for the predicted covariance matrix.
N_{init}	5	Minimum number of updates in a holding track for it to be confirmed as a sure track.
σ_1	2	Prior position covariance during initiation.
σ_2	2	Prior velocity covariance during initiation.
v	5	Prior velocity mean during initiation.

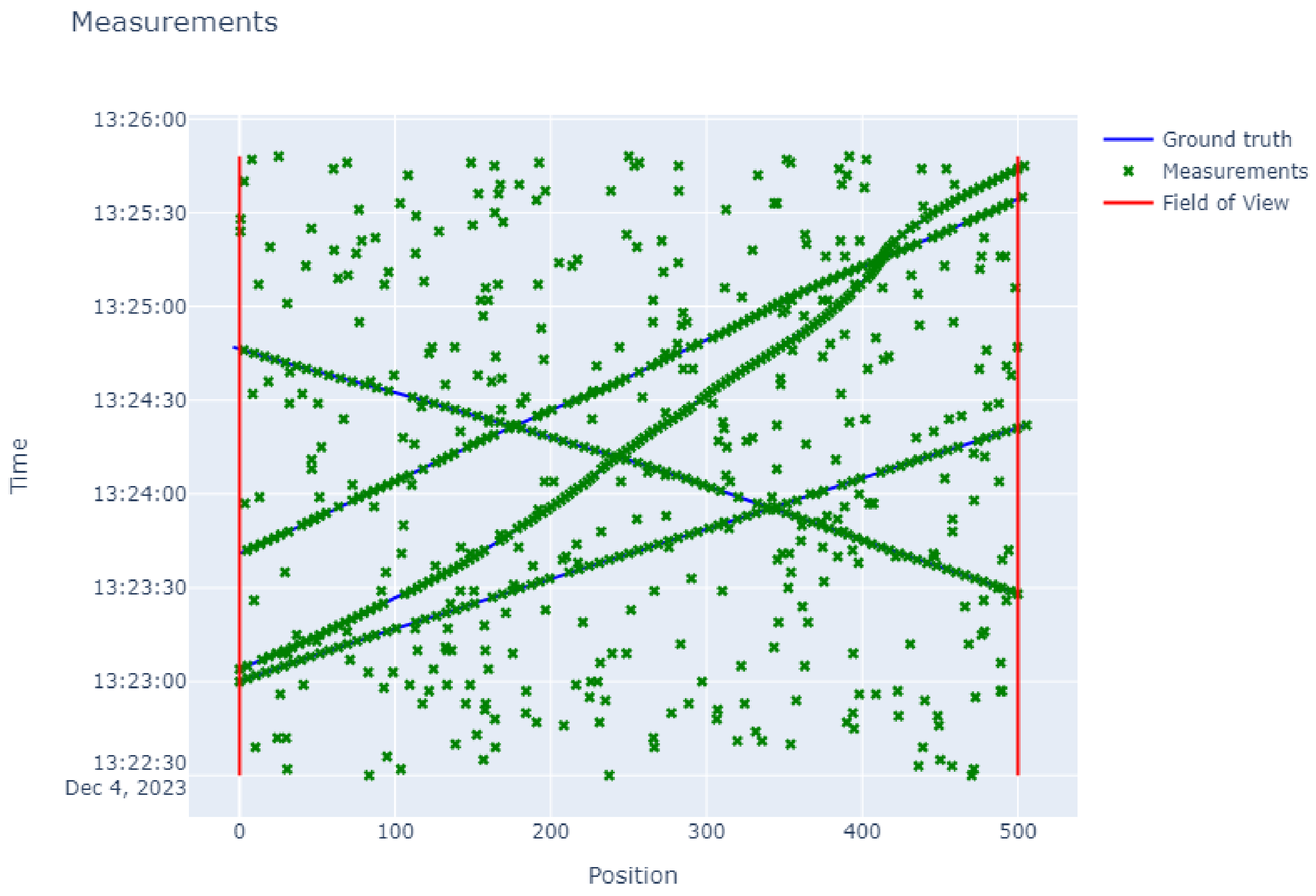


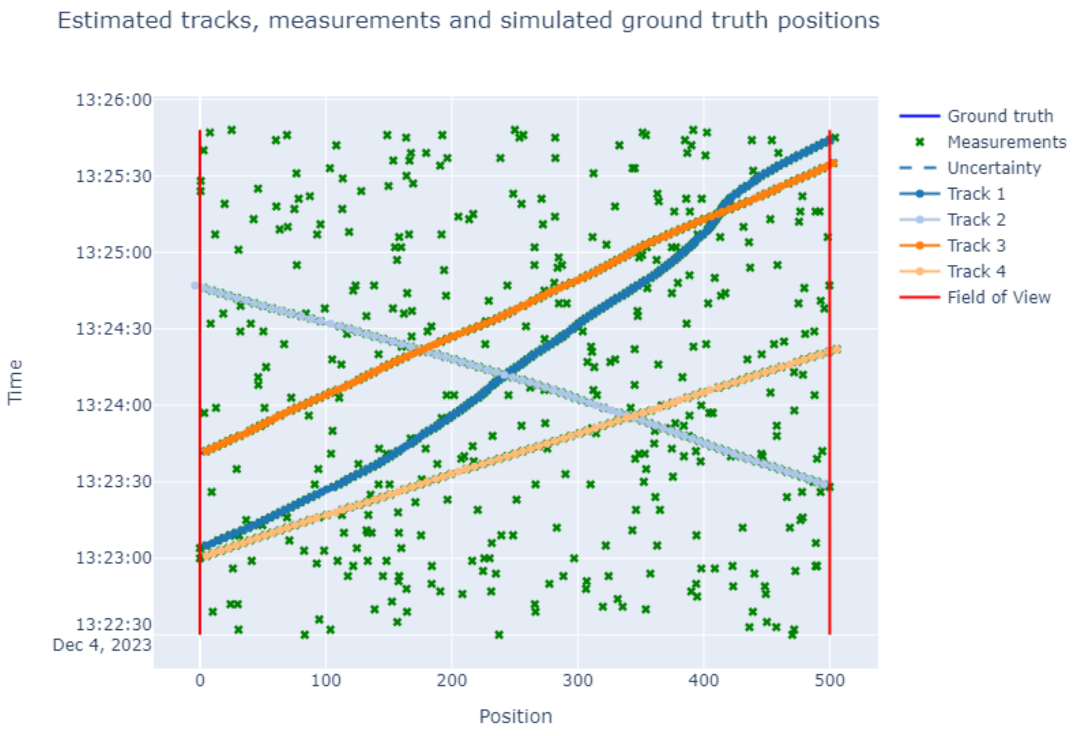
Figure 4.7: Generated measurements based on the parameter setting in Table 4.2.

the parameter setting from Table 4.2. In addition to the measurements corresponding to true object detections, one can observe occasional gaps between true detections along the true object trajectories, representing missed detections. Outside the immediate neighborhood of the true trajectories, there are uniformly distributed clutter detections.

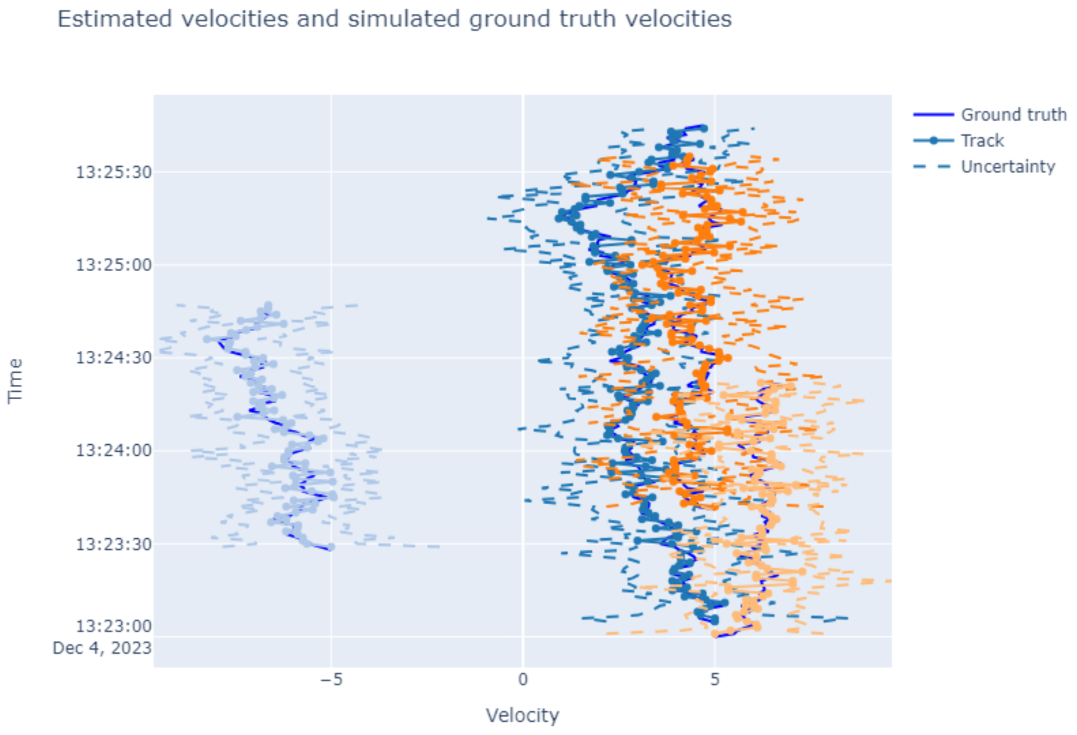
Figure 4.8 displays the JPDA estimates of the object positions and velocities when using $\sigma_q = 1$. This is quite a larger value than the one in the true states, but it seems to estimate the states quite well despite this. In fact, we looked at the case with a Kalman predictor using the same process noise level as in the true states, but the filter showed small tendencies towards the case in which the filter trusted its own predictions too much. Therefore, when comparing the tracking performance of this process noise level to a slightly higher one, the latter gave better results.

In Figure 4.9 the effects of an inappropriate choice of the process noise is clearly displayed; The Kalman predictor here is using a process noise of $\sigma_q = 0.001$. Since this is much lower than the noise level in the true object states, the filter trusts its own predictions too much and fails to adapt to changes in the system dynamics. This is clearly shown by the nearly constant-velocity predictions of essentially all of the initiated tracks. Another aspect of this is that the filter initiates many more tracks than actually exist in the given time interval. This is because the predicted covariance of the object state rarely reaches the deletion threshold because of the high confidence in its own predictions, which means that tracks can be propagated a long time before the filter covariance reaches the deletion threshold, leading to an inaccurate total count of objects.

On the contrary, Figure 4.10 shows the consequence of using a too high process noise. Here the Kalman predictor is using a process noise $\sigma_q = 100$. Since this is much higher than the level in the true object states, the filter has little confidence in its predictions, and adapts too quickly and hard since it expects large variations in the dynamical system. As mentioned earlier, the filter is prone to overfitting in this situation. This is indeed the case, especially if we look at the velocity estimates. Observe also how much faster the covariance grows for each time step. The filter deletes the track very early because of these high covariances, caused by either a missed detection (filter prediction) or a filter update from a noisy measurement.

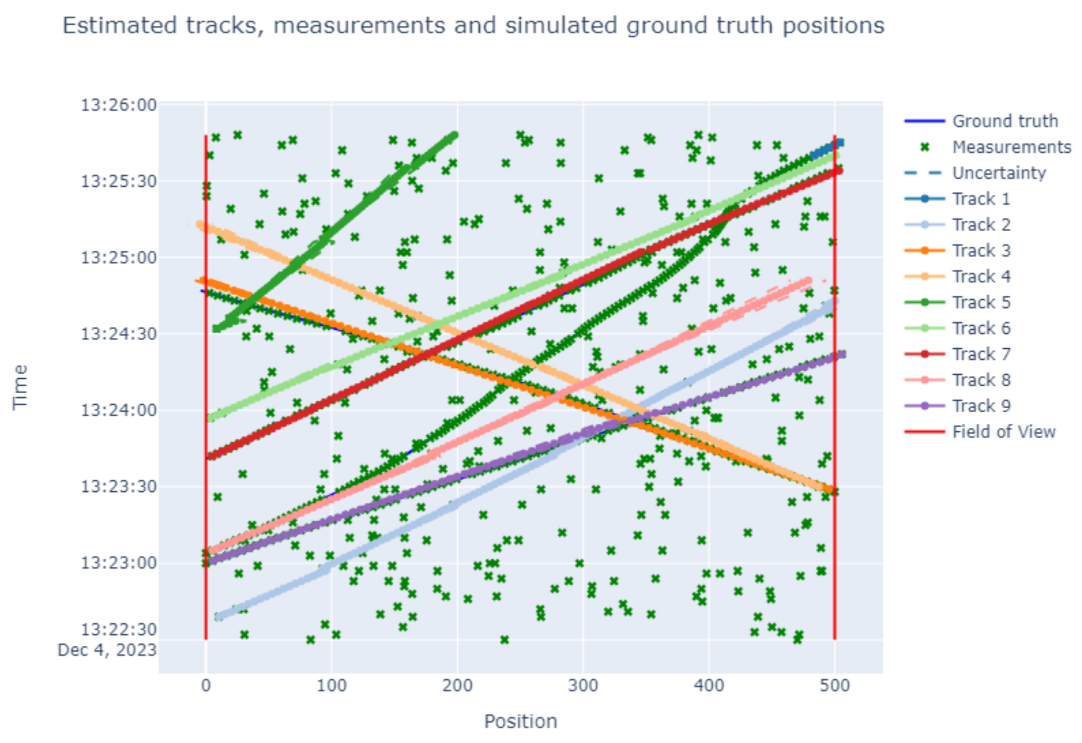


(a) Estimated object positions.

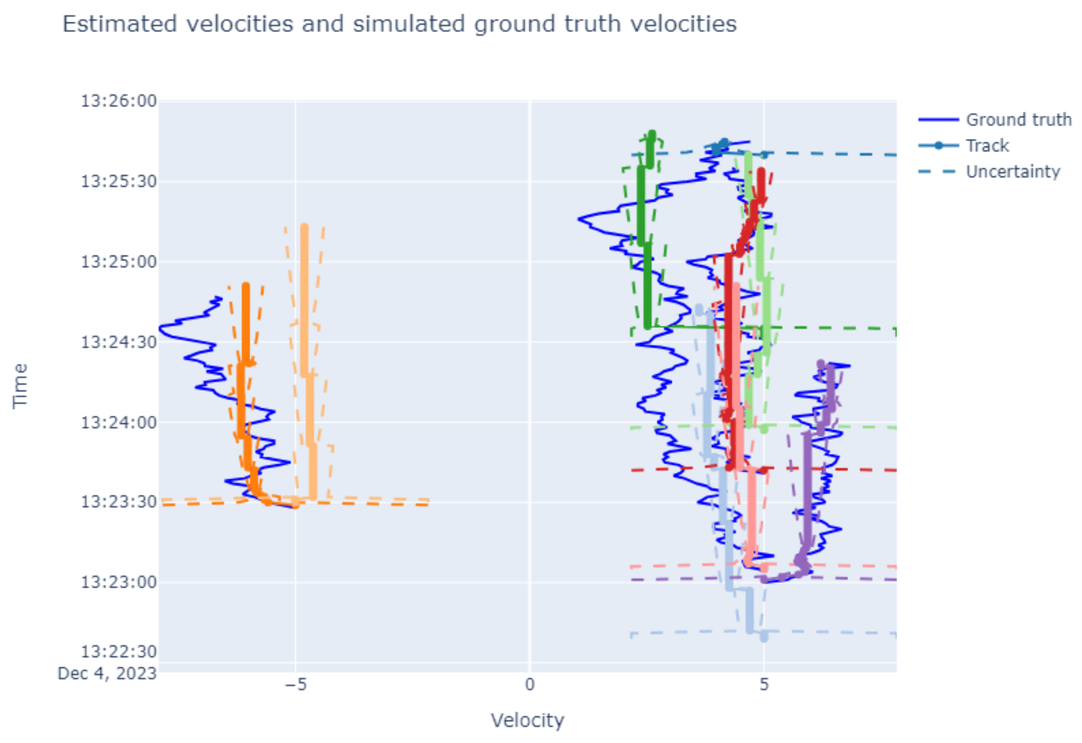


(b) Estimated object velocities.

Figure 4.8: JPDA-estimated object states using a Kalman predictor with process noise $\sigma_q = 1$.

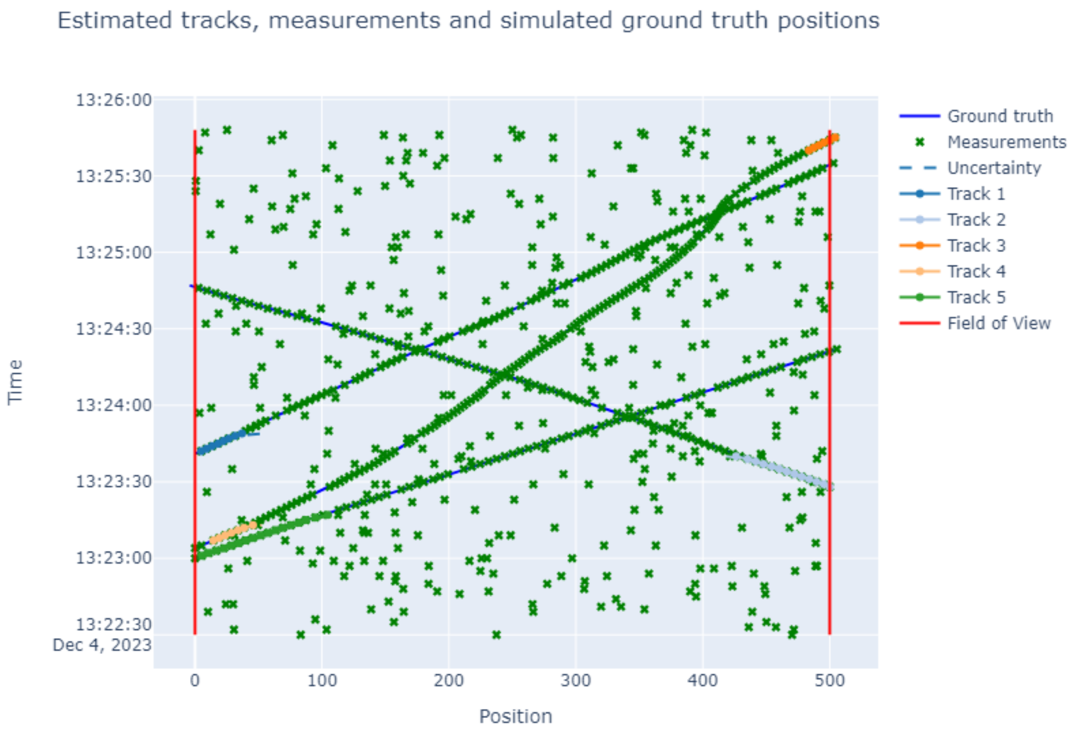


(a) Estimated object positions.

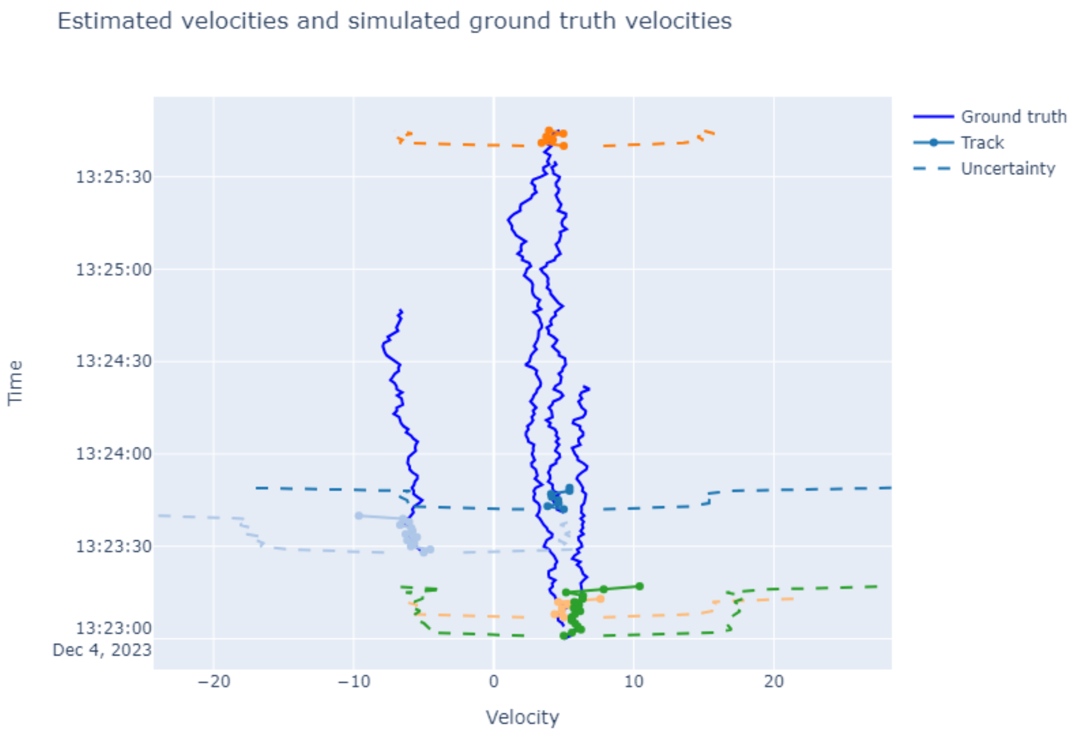


(b) Estimated object velocities.

Figure 4.9: JPDA-estimated object states using a Kalman predictor with process noise $\sigma_q = 0.001$.



(a) Estimated object positions.



(b) Estimated object velocities.

Figure 4.10: JPDA-estimated object states using a Kalman predictor with process noise $\sigma_q = 100$.

Chapter 5

Tracking vehicles from DAS data

In the previous chapter we examined the KF's behavior in a simulation scenario, with the objective of seeing the impact the adjustment of process noise covariance might have on the tracking performance. These insights will be taken into account when we now create a tracker for vehicles based on picks of DAS data originating from the road and railway at Selsbakk. The primary objectives in this chapter is to report the results from using JPDA tracking on the DAS signal picks, unveil traffic patterns and perform vehicle and train counting based on the state estimates. In addition, we aim to use the signal pick amplitude information to estimate the probabilities that the object we are tracking is a train or a car.

Tracking objects that might exhibit different dynamics (acceleration, constant velocity etc.) at random times, not to mention the fact that the tracked objects can be of different classes (cars and trains), raises the question whether one transition model can accurately describe the various dynamic behaviors, or system modes. From a general perspective, this is unlikely. Cars and trains often move at different speeds, and trains likely display a more conservative velocity pattern compared to cars. This is because cars must adjust the speed according to the neighboring cars, speed zones etc. leading to more acceleration. This kind of switch in object dynamics can generally not be described by a single transition model.

Dingler (2022) presents the theory behind the Interactive Multiple Model (IMM) method. Here the idea is that each object can be modeled by multiple transition models, and can switch between models randomly in time, according to what model describes the current object dynamics best. Therefore, two questions arises:

- According to which method does the system behave?
- If so, and when is there a switch between models?

Dingler (2022) starts by presenting what is referred to as *first generation* methods, which are methods that only resolves the first question. The Autonomous Multiple Model (AMM) assumes that the optimal state estimation can be achieved by only one (unknown) model. This is solved by applying r parallel KFs that provides a state estimate for each model i. Then, by the law of total expectation one finds the optimal estimate, weighing each individual state estimate with the posterior probability that an optimal single estimate is obtained with model i. This probability can be found by the use of Bayes' theorem, and a likelihood model consisting of a probability distribution that describes whether to use model i. This likelihood model is often Gaussian, with the mean equal to the predicted measurement, and covariance matrix equal to the residual covariance from the i-th KF. It is expected that the Bayes recursion converges to 1 for the correct (or best describing) model.

Furthermore, the IMM is what can be referred to as *second generation* methods. The derivations are similar to the first generation methods, but it differs in the derivation of the probability that model i is the model that provides the best estimate to the object's current dynamics. Instead of using the previous posterior probability as a prior in the current time step, it uses a linear combination (law of total expectation) of all the r previous posterior probabilities. The weights here are transition probabilities p_{ji}, quantifying the probability that the system switches from model j to model i at the next time step. This is a Markov chain, with states corresponding to the various modes the object can behave according to. For our case, one could assume that one object could work according to one of these modes:

$$S_k = \begin{cases} \texttt{Car, constant velocity,} \\ \texttt{Car, accelerating,} \\ \texttt{Train, constant velocity,} \\ \texttt{Train, accelerating,} \end{cases} \tag{5.1}$$

with corresponding transition and measurement matrices, i.e. $\mathbf{A}_k = \mathbf{A}(S_k), \mathbf{H}_k = \mathbf{H}(S_k), \mathbf{Q}_k = \mathbf{Q}(S_k)$ and $\mathbf{R}_k = \mathbf{R}(S_k)$. However, to limit the scope of this thesis, we decided to proceed with one transition model, both for object type and kinematic behavior. We will see that this has consequences for the quality of the tracking in some scenarios, but that it suffices for the majority of the cases. So, the assumptions in this thesis are:

- Cars and trains can be modeled with the same transition model. However, they are considered two different classes upon which the KF is conditioned.
- Cars and trains are assumed to move with close to constant velocity.
- Measurements only include noisy observations of the object position (signal picks), and the amplitude of the picks is left as an external attribute.

This means that unlike in the IMM paper, we do not use the positional measurements in the calculation of the probability that an object is in a certain mode (S_k), but rather use the signal pick amplitudes with Bayes' theorem "independently" of the Kalman filtering to decide whether the object is a car or a train. However, the classification scheme is not completely independent from the MOT scheme in the sense that the signal pick amplitudes used in Bayes' theorem are those corresponding to the measurements falling inside the object gate in the MOT scheme. This is discussed in more detail in the next section.

5.1 Object classification using fiber strain amplitude

The MOT framework presented in the previous chapter only takes into account the measurement of the objects *position*, and not external data such as signal amplitude. So, we want to add functionality to the tracking application that based on the amplitudes of the measurements being used to update tracks of the objects, can quantify the likelihood that the object is a car or a train. We define the *class* variable as

$$\Gamma = [\gamma^1, ..., \gamma^i, ..., \gamma^{n_k}], \tag{5.2}$$

where

$$\gamma^i = \begin{cases} 1 & \text{if object } i \text{ is a car,} \\ 2 & \text{if object } i \text{ is a train.} \end{cases} \tag{5.3}$$

Then the prior probability for object i is given by

$$P(\gamma^i = l) = \pi_l^i, \quad s.t. \quad \sum_{l=1}^{2} \pi_l^i = 1. \tag{5.4}$$

These prior probabilities are set using the logged data presented in Chapter 2, with $\pi_1^i = \pi_1 \; \forall i$ equal to the proportion of logged cars, and equivalently, $\pi_2^i = \pi_2 \; \forall i$ equal to the proportion of logged trains. Given this classification of objects, one could create new transition models based on the values of γ as mentioned in the introduction of the chapter, but in this thesis we assume that both cars and trains follow the same physical dynamics, and therefore the same transition model. So, assuming a time and class invariant transition model, the multi-object predictive density can be written

$$p_k(\mathbf{X}_k | \mathbf{X}_{k-1}, \Gamma = L) = \prod_{i=1}^{n_k} \phi(\mathbf{x}_k^i; \mathbf{A}\mathbf{x}_{k-1}^i, \mathbf{Q}). \tag{5.5}$$

The prediction and update steps in the KF remain the same, conditioned on the class variable γ^i. We are interested in using the amplitudes of the measurements $\{\mathbf{z}_k^1, ..., \mathbf{z}_k^{m_k^{DA}}\}$, which is a subset of all measurements at time k. This subset corresponds to the measurements falling inside at least one of the n_k measurement gates, and thus can be considered a detection candidate. Let the amplitude of the measurement $\mathbf{z}_k^j$ be denoted by $\mathbf{y}_k^j$, so that $\mathbf{Y}_{1:k}$ denotes the sequence of measurement amplitudes up to time k. We define a likelihood model for new measurement amplitudes as

$$p(\mathbf{y}_k^j | \gamma^i = l) = \phi(\mathbf{y}_k^j; \alpha_l, r_l^2). \tag{5.6}$$

Here α_1 and α_2 are the empirical averages over the logged amplitudes caused by cars and trains respectively. Similarly r_1^2 and r_2^2 are the empirical variances of the logged amplitudes caused by cars and trains respectively. It is reasonable to assume that new measurement amplitudes will be normally distributed around α_1 if the measurements originate from a car, or around α_2 if the measurements originate from a train. Indeed, by looking at the distribution of all amplitudes recorded the 31st of August 2021 in Figure 5.1, we can observe that the distribution is bimodal and fairly symmetrically bell-shaped around the modes. This reflects the fact that we have mainly two classes of objects: cars and trains. Figure 5.1a shows the distribution of the processed signals, i.e. the signal picks that are used as measurements in the JPDA algorithm. Obviously it will contain data only above the minimum amplitude threshold a. Figure

5.1b shows the distribution of all signal amplitudes, i.e. the amplitudes without having gone through Algorithm 2. Comparing Figure 5.1a with Figure 5.1b, we see that the car signal and train signal distributions become much more apparent once we extract signal picks. We expected there to be a local maxima around α_1 also in the unprocessed signals, but we see that this is not the case. Since the majority of the signal amplitudes are of lower log-RMS value (less than -9), we omitted those from the histogram as they correspond to insignificant events or background noise.

The Gaussian densities displayed in Figure 5.1a are the empirical amplitude densities based on the mean and variance of the logged data, scaled with an appropriate factor. The amplitudes corresponding to cars seem to nicely align with the empirical amplitude density (truncated at a), as expected. We would expect the amplitude threshold a to be lower than α_1, but we saw in the sensitivity analysis conducted in Chapter 3 that $a = -8.8$ was the most appropriate threshold. So, a lower a would increase the number of false positives, i.e. tracks based on clutter measurements. In addition, the logged amplitudes are relatively sparse, which in turn gives a somewhat inaccurate estimate of the true mean of car amplitudes. Looking at the empirical amplitude density for trains, we see that it is slightly shifted to the left when comparing to the number of train signal picks. However, this is almost certainly due to the fact that we only have 3 logged trains, and therefore it is highly likely that α_2 is quite different from the true mean amplitude. But since it is significantly different from α_1, we believe it will not affect classification performance too much. So, given the likelihood model defined in (5.6), the updated class probabilities for object i are defined by Bayes' formula as

$$P(\gamma^i = l|\mathbf{Y}_{1:k}) = \pi^i_{k,l} \propto \left\{ \sum_{j=1}^{m_k^{DA}} \beta_k^{i,j} p(\mathbf{y}_k^j|\gamma^i = l) \right\} \pi^i_{k-1,l}, \tag{5.7}$$

where $\beta_k^{i,j}$ are the same marginal association weights used in the JPDA algorithm. The previous probability that object i is of class l is updated with a weighted sum of amplitudes evaluated in the likelihood model for class l. Since the marginal association weights reflects the probability of a measurement being a detection of the object, it is reasonable to assume a similar relationship between the signal amplitude and the object. If a measurement-object pair has a very low association weight, we expect the measurement to be far away from the true object position. It is natural that in turn the measurement amplitude is lower, which if valued too high could lead to a misclassification of the object. In summary, we believe using the same weights when summing over the different amplitude likelihoods as the ones used in the JPDA update is a reasonable approach.

5.1.1 Simulation example

To demonstrate the performance of the proposed classification method, we conduct a simulation example. We generate ground truth trajectories of both cars and trains using the hyperparameter setup from the simulation example in the previous chapter. The order of the simulated vehicle passings, along with their start time is displayed in Table 5.1. In addition to the generation of positional measurements of the objects and Poisson clutter, we generate "amplitudes" of each measurement. This is done by sampling from the following Gaussian densities:

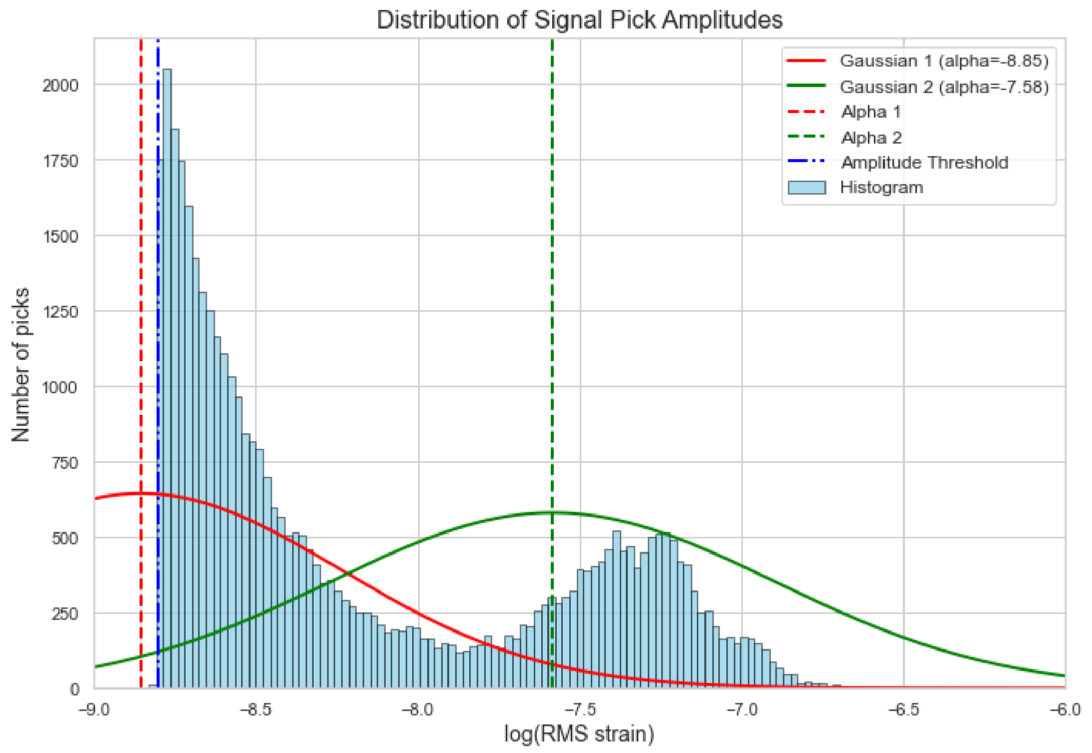
$$\mathbf{y}_k^j \sim \begin{cases} \mathcal{N}(-8.8, 1^2) & \text{if } \theta_k^i = j \text{ and } \gamma^i = 1, \\ \mathcal{N}(-7, 1.2^2) & \text{if } \theta_k^i = j \text{ and } \gamma^i = 2, \\ \mathcal{N}(-10, 1^2) & \text{if } \theta_k^i = 0, \end{cases} \tag{5.8}$$

depending on whether the measurement $\mathbf{z}_k^j$ is associated to any object, and if so, the class assigned to the object. Next, assume that the vehicle classes and true amplitude distributions are unknown, and that we are only able to observe a small subset of the amplitudes each object is causing. For example, we observe a small number of trains and cars passing at a specific location within the FOV, and we are able to record one amplitude per object. This means we will have small samples of amplitudes of each class, and can use the empirical mean and variance of these samples to construct Gaussian likelihood models for new amplitudes. We simulate this situation by sampling 10 samples from $\mathcal{N}(-8.8, 1^2)$ corresponding to car amplitudes, and 3 samples from $\mathcal{N}(-7, 1.2^2)$ corresponding to train amplitudes. We calculate the empirical mean and variance of each of the samples, resulting in the following empirical Gaussian likelihood models to be used in Bayes formula:

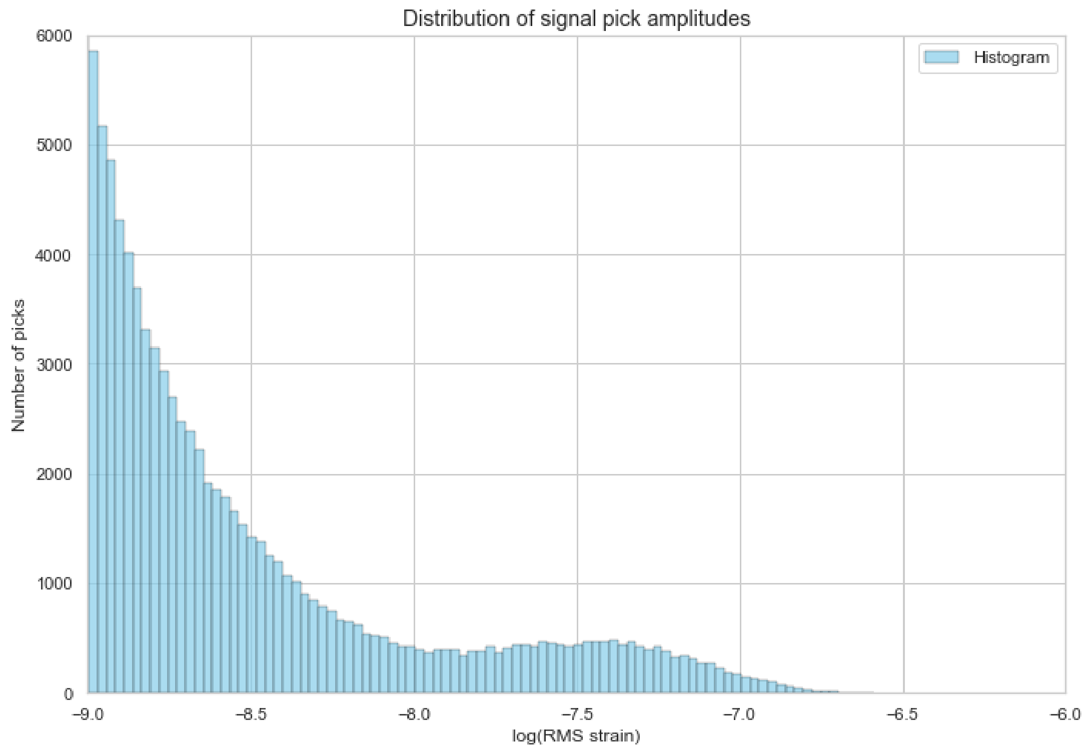
$$p(\mathbf{y}_k^j|\gamma^i = 1) = \phi(\mathbf{y}_k^j; -8.813, 1.391^2), \tag{5.9}$$

$$p(\mathbf{y}_k^j|\gamma^i = 2) = \phi(\mathbf{y}_k^j; -8.209, 1.244^2). \tag{5.10}$$

Figure 5.2 shows the estimated tracks of each object. We see that for one of the objects, JPDA looses track of the object, and the track gets deleted too early. But overall, the tracking is sufficiently accurate. Figure 5.3 shows the same estimated tracks, but with the car probability at each estimated position. Comparing the true vehicle type and start time for each object from Table 5.1 to the probability displayed at each track in Figure 5.3, we see that the Bayes probability updater is able to correctly identify the true object class for all the simulated objects (at least



(a) Distribution of signal pick amplitudes, i.e. the amplitudes after the application of the signal extraction algorithm (Algorithm 2).



(b) Distribution of all signal amplitudes (unprocessed)

Figure 5.1: Distribution of signal amplitudes and empirical amplitude densities for cars and trains based on the logged data.

when using cutoff probability 0.5). This is despite the fact that we are using empirical likelihood models based on a very small amplitude sample, and the certainty in the probability estimates only become higher when we increase the empirical sample size. However, for some of the objects, the classifier struggles to identify the correct class for a while, but eventually ”makes up its mind”. This is especially true for the object (true object class is car) starting at 13:18:38, where the classifier for a long while classifies the object as a train. Just at the end of the track, it increases the probability of being a car. This behavior is to be expected, given the scarcity of the empirical samples and high variances in both the true and empirical amplitude distributions.

Table 5.1: Simulated car and train passings.

Start time	**Vehicle type**
2024-01-08 13:15:22	train
2024-01-08 13:15:55	car
2024-01-08 13:17:04	train
2024-01-08 13:17:31	car
2024-01-08 13:18:38	car
2024-01-08 13:18:54	train
2024-01-08 13:19:01	train
2024-01-08 13:19:20	train
2024-01-08 13:19:26	train
2024-01-08 13:19:29	car
2024-01-08 13:19:32	car

5.2 Tracking and classification during dense traffic

5.2.1 Example I (with logged events)

The parameters used for the JPDA setup are reported in Table 5.2, 5.3 and 5.4. Since the dynamics of the objects we are tracking can be assumed to maintain a fairly constant velocity, a process noise value of 1 is reasonable. This was also determined through a range of experiments with different values, and we found that σ_q^2 in the range from 0.25 to 3 gave reasonable results. When it comes to the measurement noise, we expect a high degree of noise, due to the clustering of fiber strain values. It can also be observed directly from the signal picks that they are way more noisy compared to the simulated measurements in the simulation example. We choose $\sigma_r^2 = 15$, but we found that σ_r^2 in the range of 5 to 30 also gave satisfactory results. The choice of these noise parameters is always a balance, and different parameter values might work better in a different tracking scenario. We indeed saw this phenomenon when tracking on different times of the day, and therefore slight modifications to the parameter values were made, but within the mentioned intervals. This suggests for a future task that the implementation of an adaptive process and measurement noise matrix could be beneficial. The parameters related to the prior state in the initiation of new objects were determined through experimentation and observation of the strain data. An initial velocity of 36 km/h is reasonable since the road segment is divided into both a 30 and a 50 speed limit zone. A positional variance of 10 m is reasonable given the high noise in measurements, and a lower variance in velocity is reasonable since the objects are expected to move with close to constant velocity. The initiation rate of 60m is quite wide given the rather short extent of the FOV, but necessary in our case since there are not so many measurements per object. The deletion threshold on the covariance and the minimum number of updates are determined through experimentation. The FOV however was determined by inspecting the data, and evaluating the quality and frequency of signal picks in different areas to arrive at a feasible FOV.

We now apply the signal extraction algorithm presented in Algorithm 2 and the JPDA algorithm to DAS data recorded from approximately 14:24 to 14:34 the 31st of August 2021. Figure 5.4 shows the FOV and the measurements (signal picks) used in the JPDA algorithm. Naturally, only the measurements inside the FOV are used for tracking.

Table 5.2: Parameter configuration related to transition model and FOV

Parameter Name	**Value**	**Explanation**
σ_q^2	1	Process noise parameter.
T	3000	Number of time steps.
V_a	3963 m	Start of the FOV.
V_b	4167 m	End of the FOV.
Δt	0.2	Time step size.

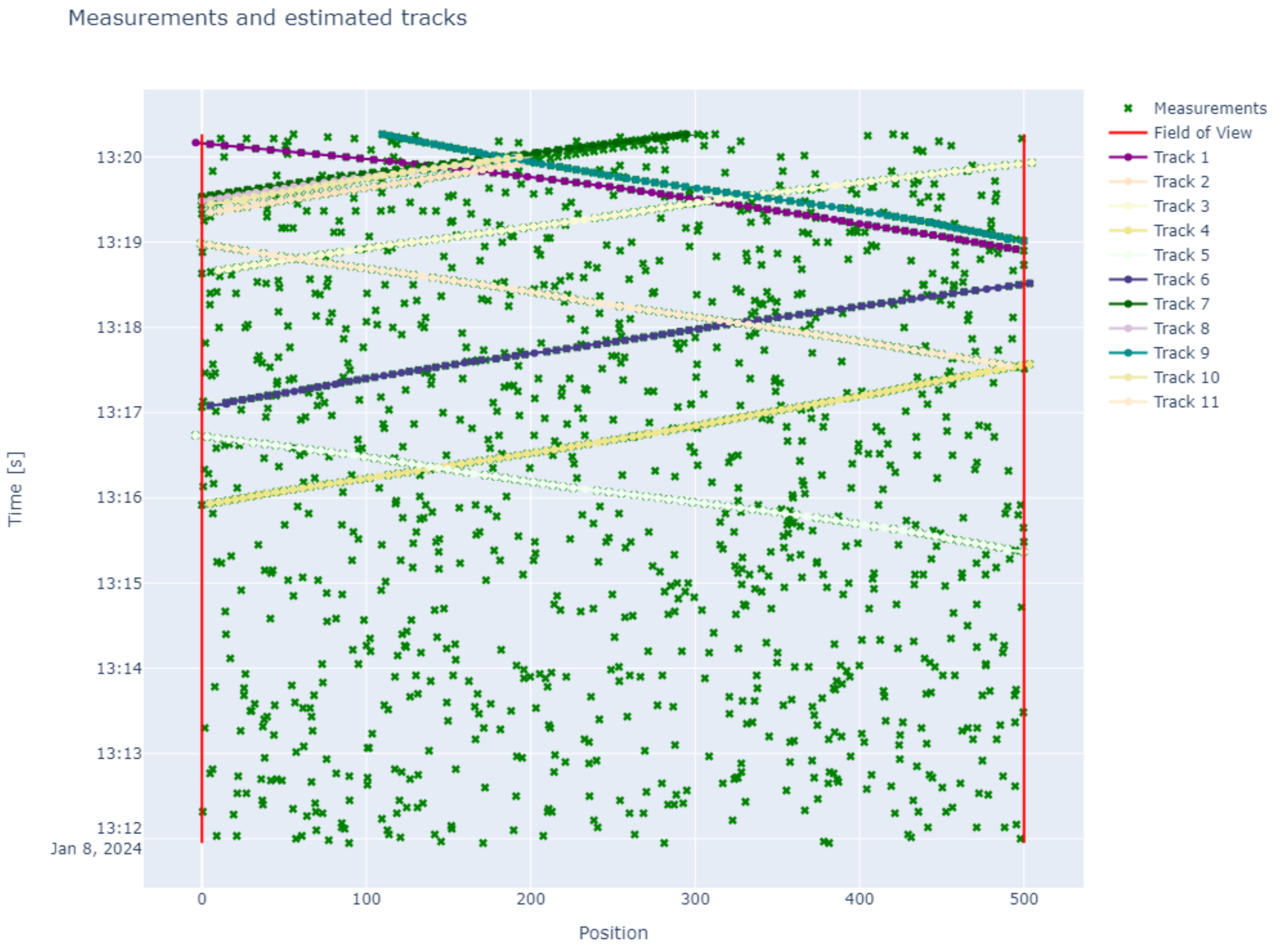


Figure 5.2: Simulated object measurements, clutter, and JPDA-estimated tracks of the object positions.

Table 5.3: Parameter configuration related to measurement model

Parameter Name	**Value**	**Explanation**
σ_r^2	15	Measurement noise parameter.
P^D	0.95	Probability of detection for the tracking system.

Table 5.4: Parameter configuration related to initiation and deletion of tracks

Parameter Name	**Value**	**Explanation**
δ	60m	Initiation range.
σ_{thres}	150	Covariance threshold for the predicted covariance matrix.
N_{init}	5	Minimum number of updates in a holding track for it to be confirmed as a sure track.
σ_1	10	Prior position covariance during initiation.
σ_2	2	Prior velocity covariance during initiation.
v	36 km/h	Prior velocity mean during initiation.

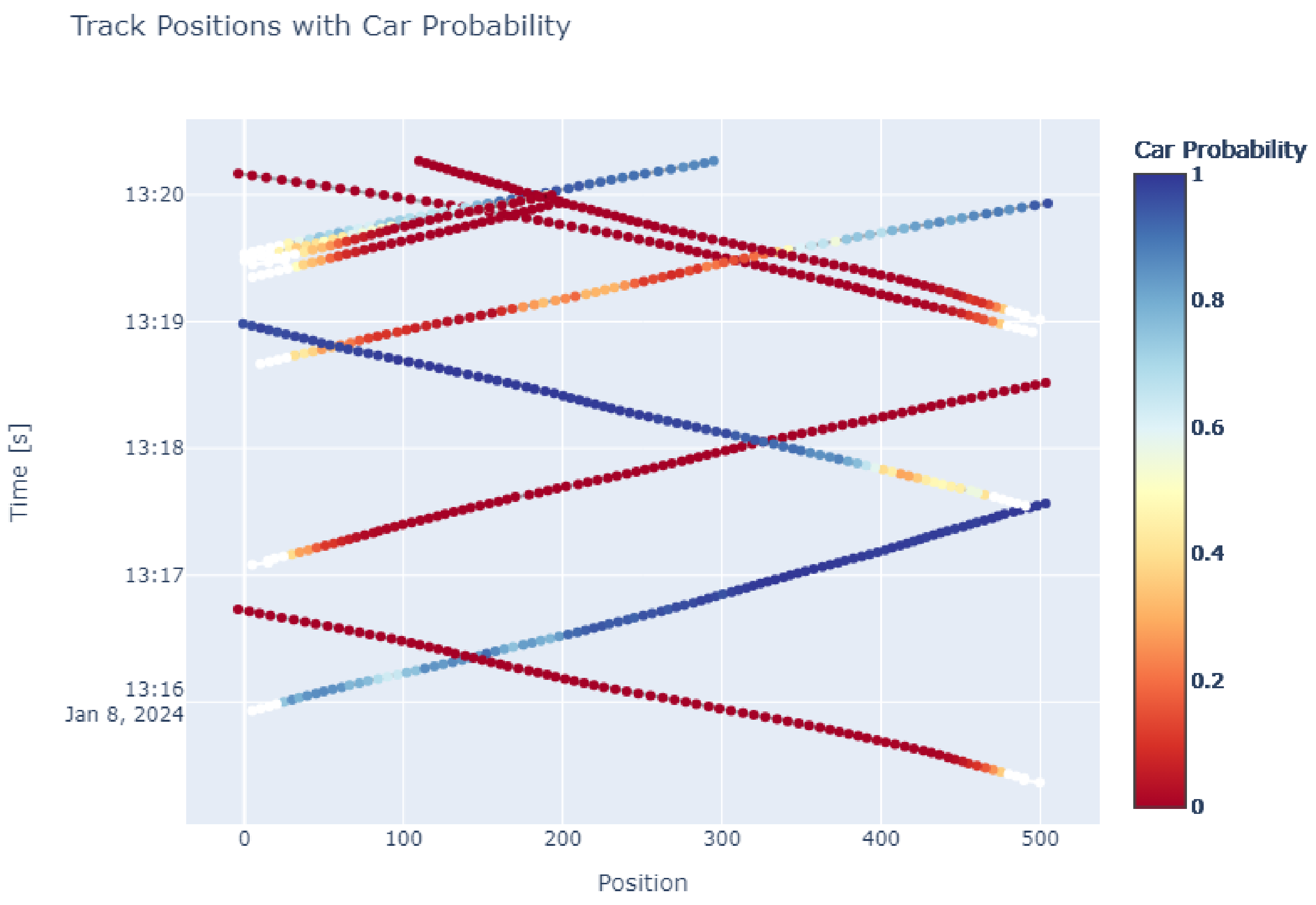


Figure 5.3: Estimated car probabilities at each estimated position for each simulated object.

As can be observed, at times where the traffic is denser, or where we experience more strain on the cable, the signal picks tend to be more noisy and even redundant, potentially affecting the tracking quality. This is particularly true from approximately 14:31 to 14:34. Also, at some times the fiber strain is quite weak, resulting in fewer signal picks, as can be seen between 14:29 and 14:30. Figure 5.5 shows the resulting tracks after applying the JPDA algorithm on the measurements. As can be observed, the tracker creates 19 tracks for the given time interval. All tracks are initiated at either side of the FOV, and are terminated when reaching the other side, or if the covariance of the estimated state exceeds the defined limit. Visually we observe that the tracks in most of the cases successfully follows the object movements, with very few exceptions. The algorithm effectively recognizes clutter measurements, by assigning those hypotheses a low weight, even in the more densely cluttered areas. Figure 5.6 shows the heatmap of the strain values, estimated tracks and the logged events discussed in Chapter 2. This figure gives an even better impression of the performance of the JPDA tracker. By comparing the logged events to the estimated tracks, one can observe that the JPDA algorithm detects and accurately tracks 19 out of the 20 vehicles and trains passing in this 10 minute time interval. Even in the cases where two cars move really close together, the algorithm is able to detect the two cars, and track them independently. We observe that the vehicle the algorithm missed, around time 14:31:30, is due to few and noisy measurements, and a highly cluttered neighborhood. It is not expected that the algorithm would be able detect this object, as it apparently had few or no measurements in the initiation field, and it was close to other cars driving both directions.

The road segment at Selsbakk is shown in Figure 5.7. Here, we observe that the road is divided into two speed zones. The green lines indicate a speed limit of 30 km/h, while the blue line indicates a limit of 50 km/h. So, there is a change in allowed speed around the Selsbakk train station. Therefore, we should expect beforehand to see this velocity change in the estimated object velocities resulting from the JPDA algorithm. Figure 5.8 shows the estimated velocities of the vehicles and trains driving north, that is, towards Marienborg Station. Each track is provided with an upper and lower uncertainty limit of two standard deviations. We observe that all tracks are initiated at negative 36 km/h. We see that after around 5 seconds, many tracks are corrected to a lower velocity, reflecting the fact that the objects remain in the 30 km/h speed zone. At later times, we observe that some tracks gradually increase the speed above 30, all though this trend is not as apparent as we expected. All in all, the velocity pattern for these tracks are well within expectation in terms of staying below the upper speed limit, and it seems that the drivers are respecting the speed limits. Looking at Figure 5.9 however, this velocity change pattern is much more visible. Here we see that almost all of the track velocities are updated to a higher speed within the first 3-4 seconds, reflecting the fact that the cars are in the 50 km/h speed zone. After around 7-8 seconds, the majority of the cars decrease their velocity to around 30 km/h, some above and some below. This is a clear trend, and proof that the created tracks actually gives accurate information regarding the kinematic properties of the cars tracked. However, we can observe that the uncertainty bounds for each track is quite high, and rapidly increasing proportionally to the number of consecutive prediction steps.

When it comes to classifying the tracked objects as car or train, we can observe the results from applying the method described in Section 5.1 to the amplitudes of the measurements used for tracking in Figure 5.10. Here, we observe the same tracks as reported earlier, but colorized according to the probability of an object being a car at each track time step. We see that the majority of the tracks have a high probability of being a car. The white points on the tracks are estimated positions during the initiation phase, and therefore probabilities were not calculated here. Comparing Figure 5.10 with Figure 5.6, the fact that the track "turns" red (i.e. low probability of being a car) is consistent with the train that is moving at approximately 14:31. This means that the method correctly identifies the train, based on the fiber strain fingerprint this train is leaving on the cable. We can also observe that the track moving north at approximately 14:26 for a while has a high train probability, but that it changes its mind at the end, and ends with a high car probability. Looking at Figure 5.6, we see that this track indeed belongs to a car. Having observed this, it seems reasonable to consider the resulting probabilities at the end of each track to use for classification.

5.2.2 Example II (without logged events)

Now we consider a different time window, from approximately 11:20 to 11:30, which is also a time with relatively dense traffic. As in Example I, the measurements used for tracking are reported in Figure 5.11. The estimated tracks are presented in Figure 5.12, and we observe that 16 tracks are created. We see that also here the JPDA algorithm successfully follows the movements of most objects. However, looking at Figure 5.13, we see that it makes more mistakes compared to Example I: Track 15 at approximately 11:27 starts tracking a car driving north right before a southbound train passes. We see that track 15 starts tracking clutter, and actually shifts direction, and gets deleted due to high state covariance. Looking at Figure 5.12 again, we see that this unfortunate tracking is because there are no more measurements for this track to be updated by, at the left most part of the FOV, so it thinks the object has turned around, since there are clutter measurements present in the opposite direction. Another mistake is done at approximately 11:29. Tracks 7 and 12 are initiated, and according to the JPDA algorithm there are two objects moving northbound close to each other, when in reality there is only one, which can be confirmed visually by looking at Figure 5.13. This is likely because of the wide initiation field we are using, and that clutter points near the object

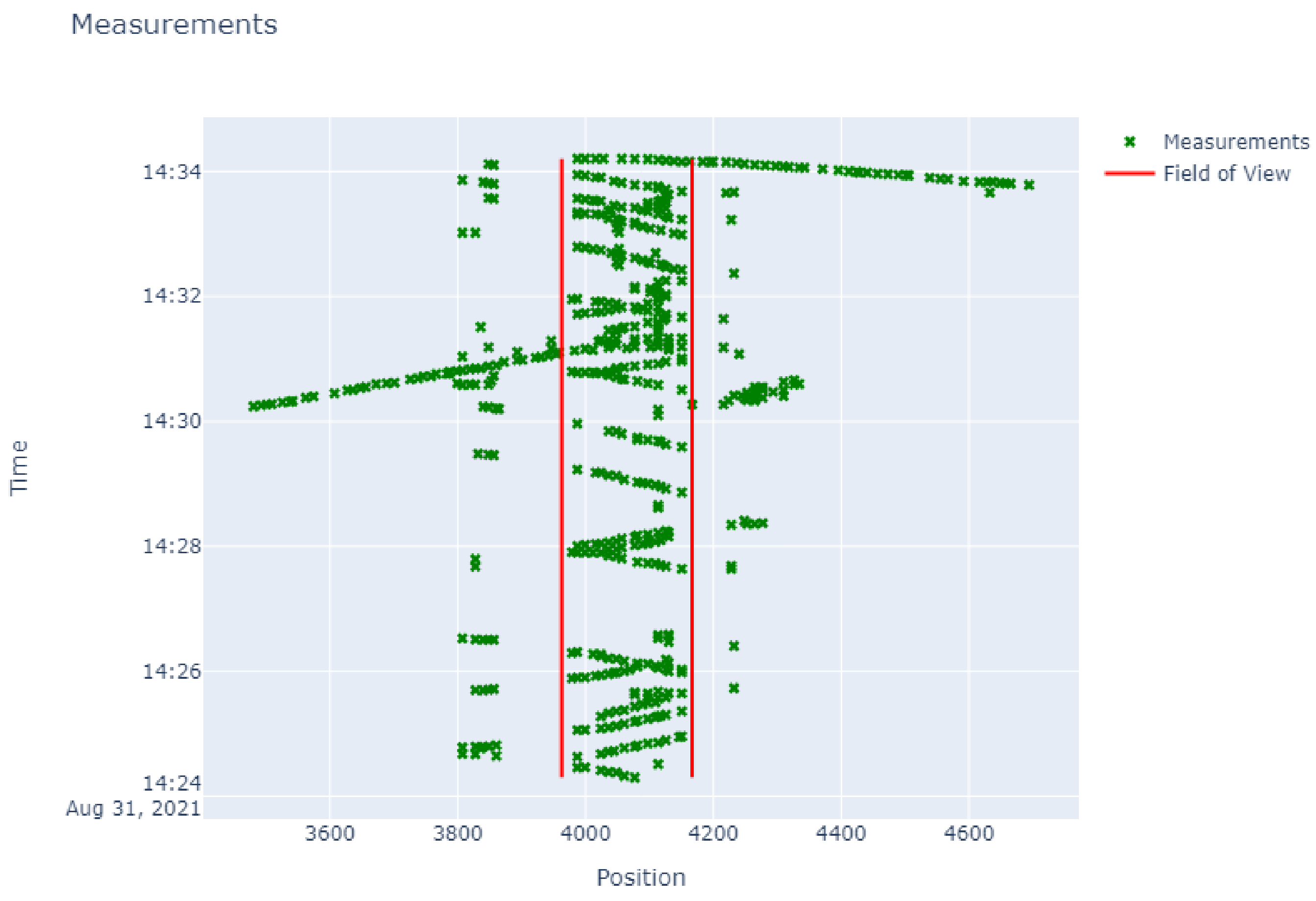


Figure 5.4: Signal picks used for tracking during the time window from 14:24 to 14:34 the 31st of August 2021. The distance unit is meters from Marienborg station.

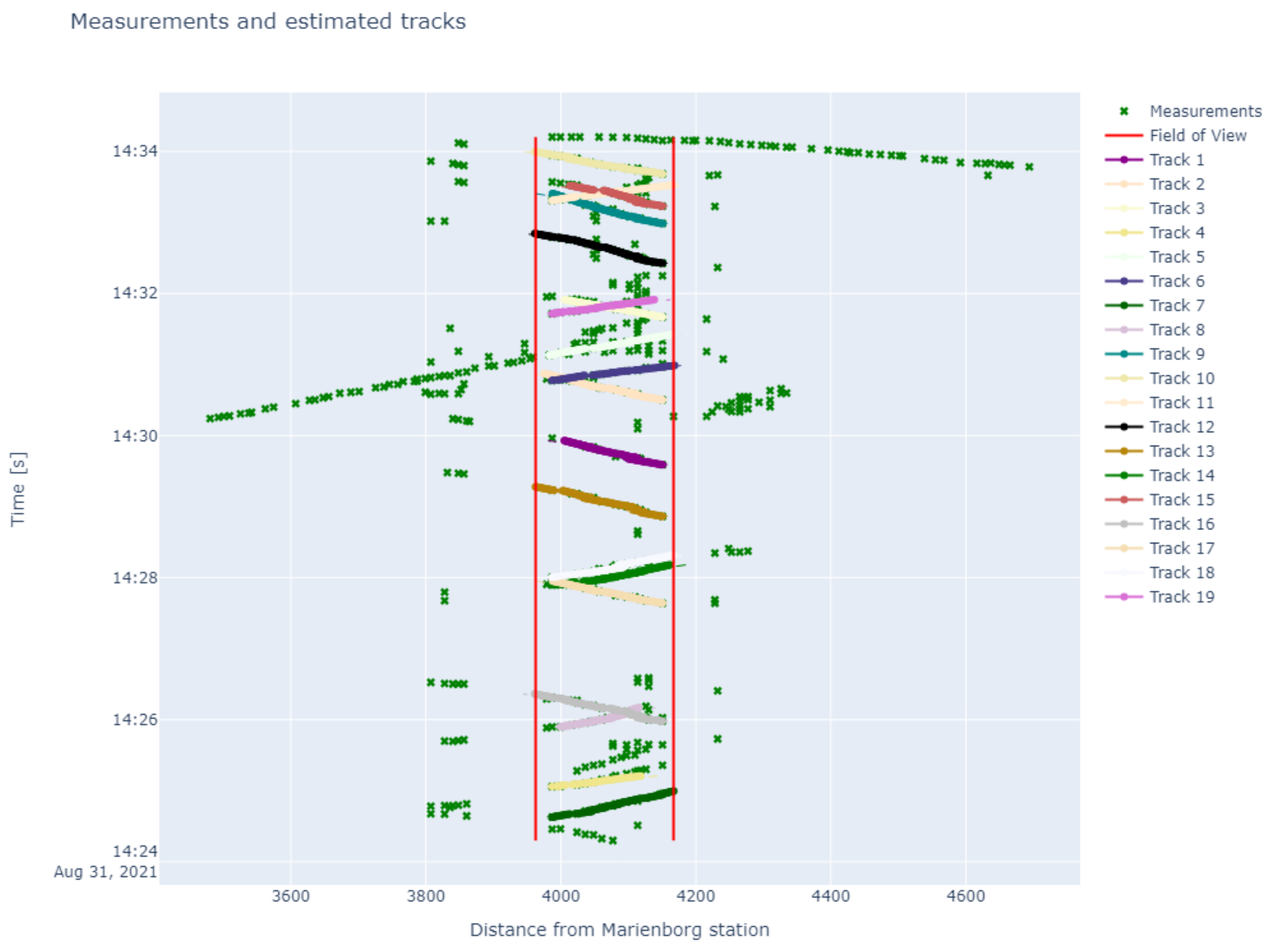


Figure 5.5: Signal picks and estimated positions of tracked cars and trains during the time window from 14:24 to 14:34 the 31st of August 2021.

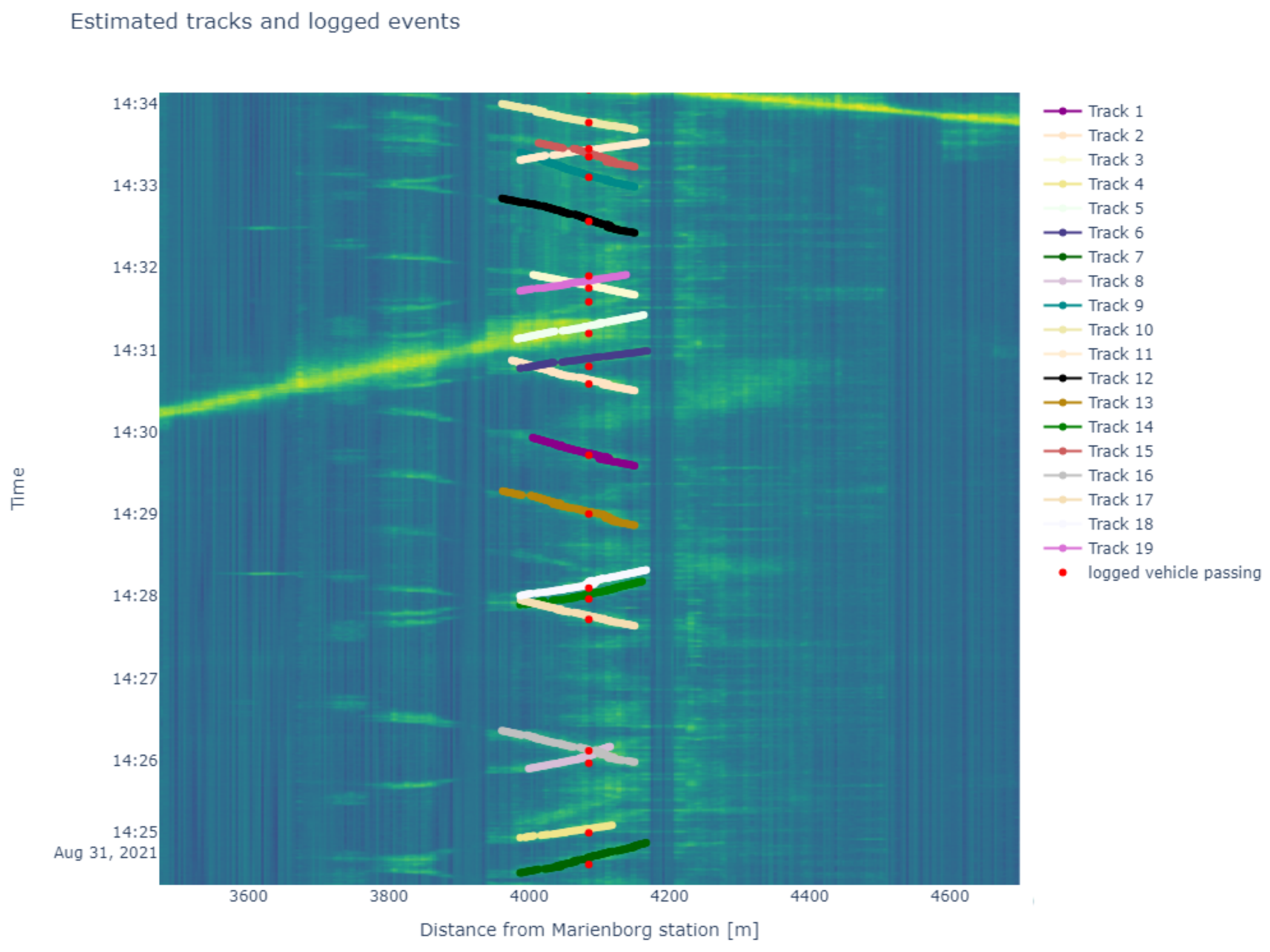


Figure 5.6: Heat map of fiber strain values, logged events and estimated positions of tracked cars and trains during the time window from 14:24 to 14:34 the 31st of August 2021.

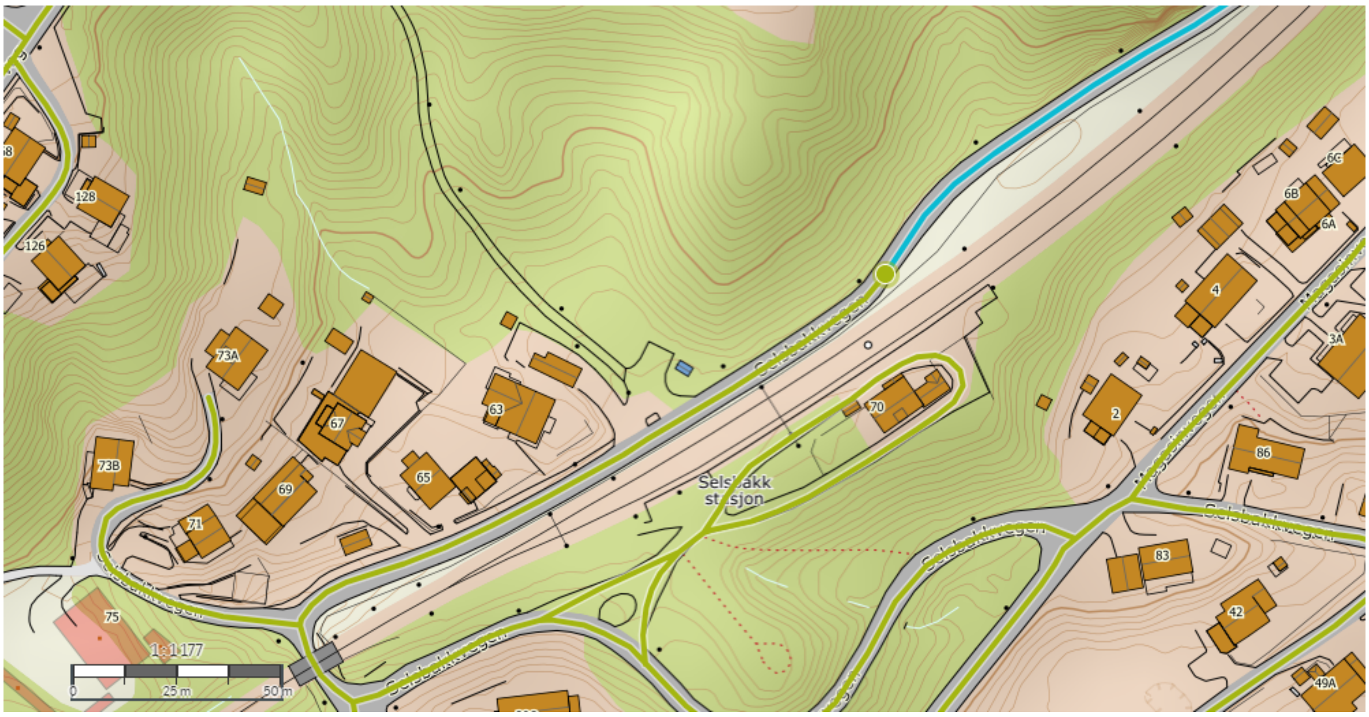


Figure 5.7: Map of the road at Selsbakk, with indicated speed zones (green is 30 km/h, blue is 50 km/h). Taken from Statens Vegvesen veikart (URL: https://vegkart.atlas.vegvesen.no/#kartlag:topo4/@268998,7037395,16/hva:!(category~(type~enum~id~2021)id~105)~)

were erroneously used in the initiation of an additional track. These mistakes are expected and acceptable, when considering the reasons why the algorithm failed in these particular scenarios. As with any scientific method applied to a real world problem, a certain error rate is inevitable. This is particularly true in this application, where we have a high number of tunable hyperparameters.

When it comes to classifying the tracked objects as car or train, we can observe the results in Figure 5.14. Also here, the method correctly identifies the train, and assigns high car probability to almost all cars. However, it seems like it believes that track 13 is a train, when in reality it is a car, when looking at Figure 5.13. Also, for some tracks between 11:27 and 11:30, it is not very confident that the tracked object is a car at some positions, but it corrects its suspicion at the end of these tracks. This is completely reasonable, since a slightly higher amplitude value can shift the updated probability obtained from Bayes' rule very fast. Even if the likelihood models developed based on the logged data are somewhat uncertain (due to the few number of logged amplitudes), they seem to suffice for our simple binary classification purpose.

Furthermore, Figure 5.15 and 5.16 shows the estimated velocities of cars and trains in the mentioned time window. Comparing with the equivalent figures in Example I, we see very similar velocity trends. Also here, the majority of the cars driving south are decreasing their velocity when entering the 30 km/h speed zone, which happens around 7-8 seconds into the tracking period. For the cars driving north, there is not such a clear trend in the velocity pattern, which was somehow not expected. The exception is the velocity estimate of track 15, that obtains a negative velocity at around 13 seconds, and should be completely disregarded since it does not track a real object.

5.3 Tracking and classification during less traffic

We have explored the performance of the JPDA algorithm in the scenario of frequent and dense traffic. We will now consider a time window in which there are fewer vehicles passing the FOV. Figure 5.17 shows the measurements and estimated tracks of vehicles at Selsbakk in the time window from 19:39 to 19:48 the 31st of August 2021. Since it is later on the day, there is less traffic, and the algorithm initiates 6 tracks in this time window. Inspecting the heat map in Figure 5.18, we can visually confirm that the tracks follow the movement of all the vehicles accurately. Figure 5.19 shows that all objects are classified as cars, which is also the most probable case when looking at the heatmap.

After having seen how the JPDA algorithm performs in a variety of traffic scenarios, we can safely say that it does a good job identifying and tracking most objects moving inside the FOV, as long as a reasonable amount of measurements is associated with each object throughout the FOV. Another important aspect for the successful tracking is that objects entering the FOV has an initial velocity somewhat similar to that assumed in the prior object

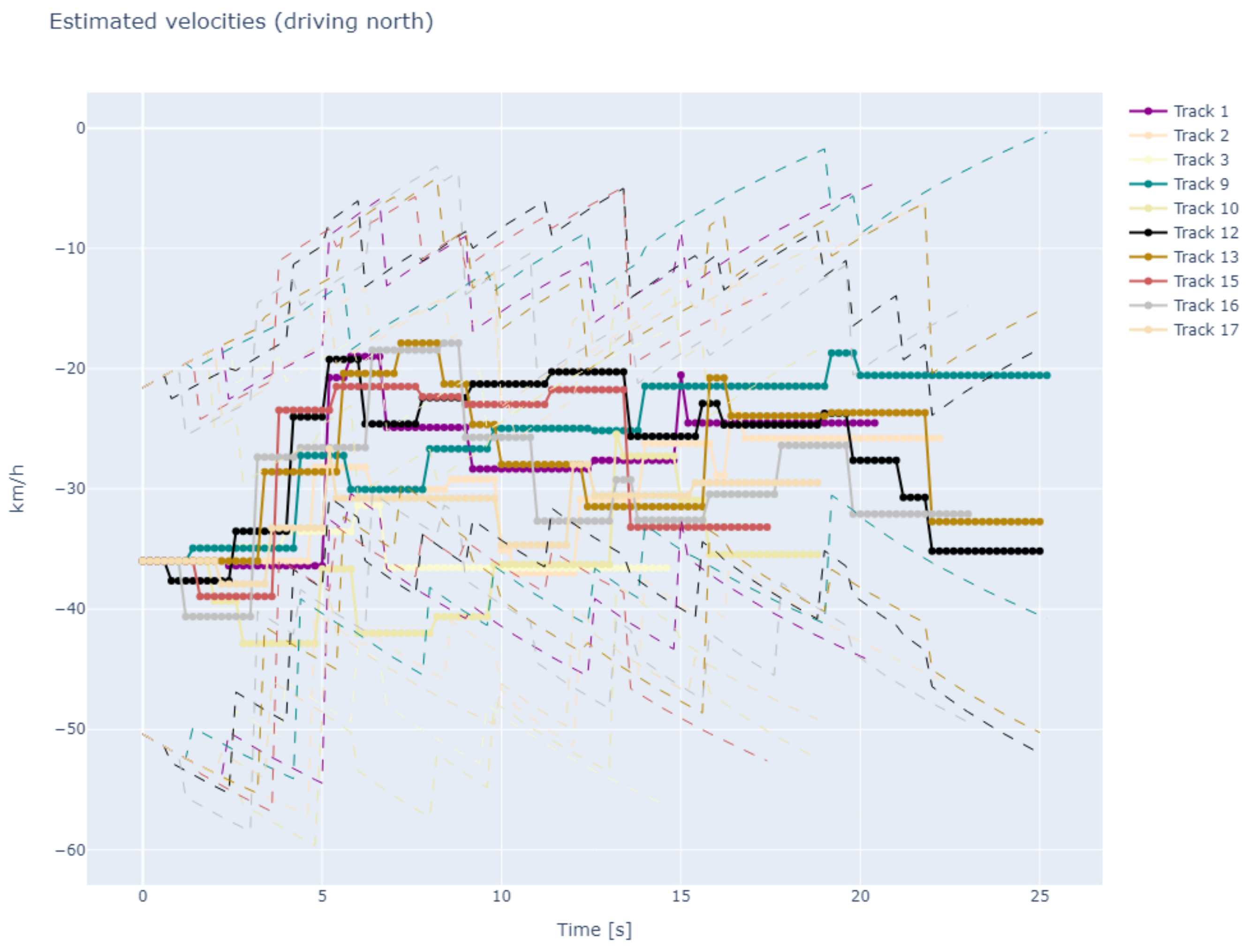


Figure 5.8: Estimated velocities of the tracked cars and trains driving north (towards Marienborg station) during the time window from 14:24 to 14:34 the 31st of August 2021.

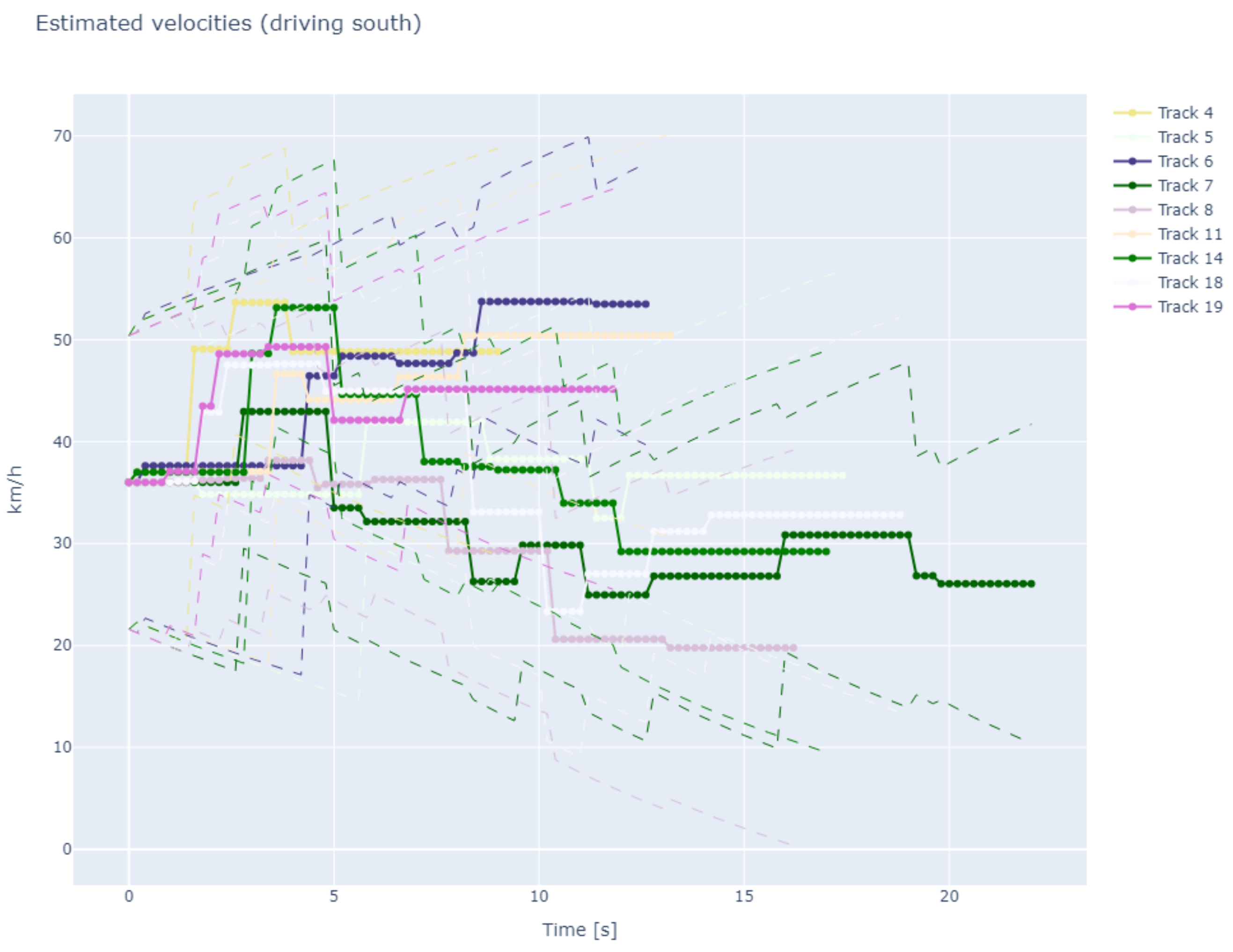


Figure 5.9: Estimated velocities of the tracked cars and trains driving south (away from Marienborg station) during the time window from 14:24 to 14:34 the 31st of August 2021.

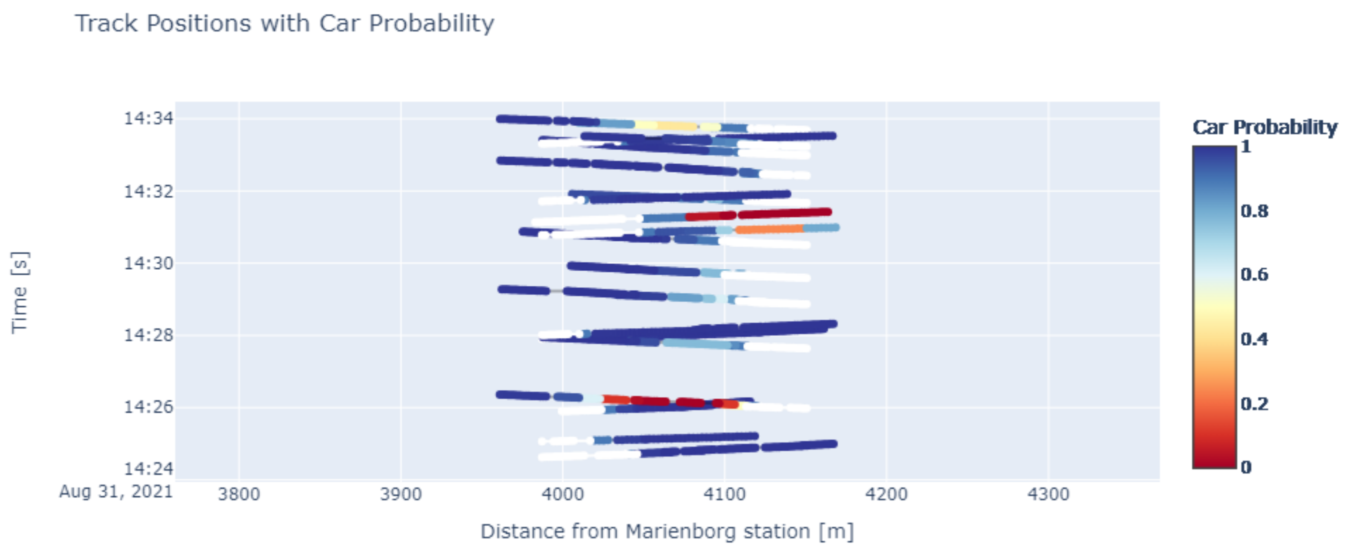


Figure 5.10: Car probability at each estimated position for all tracks during the time window from 14:24 to 14:34 the 31st of August 2021.

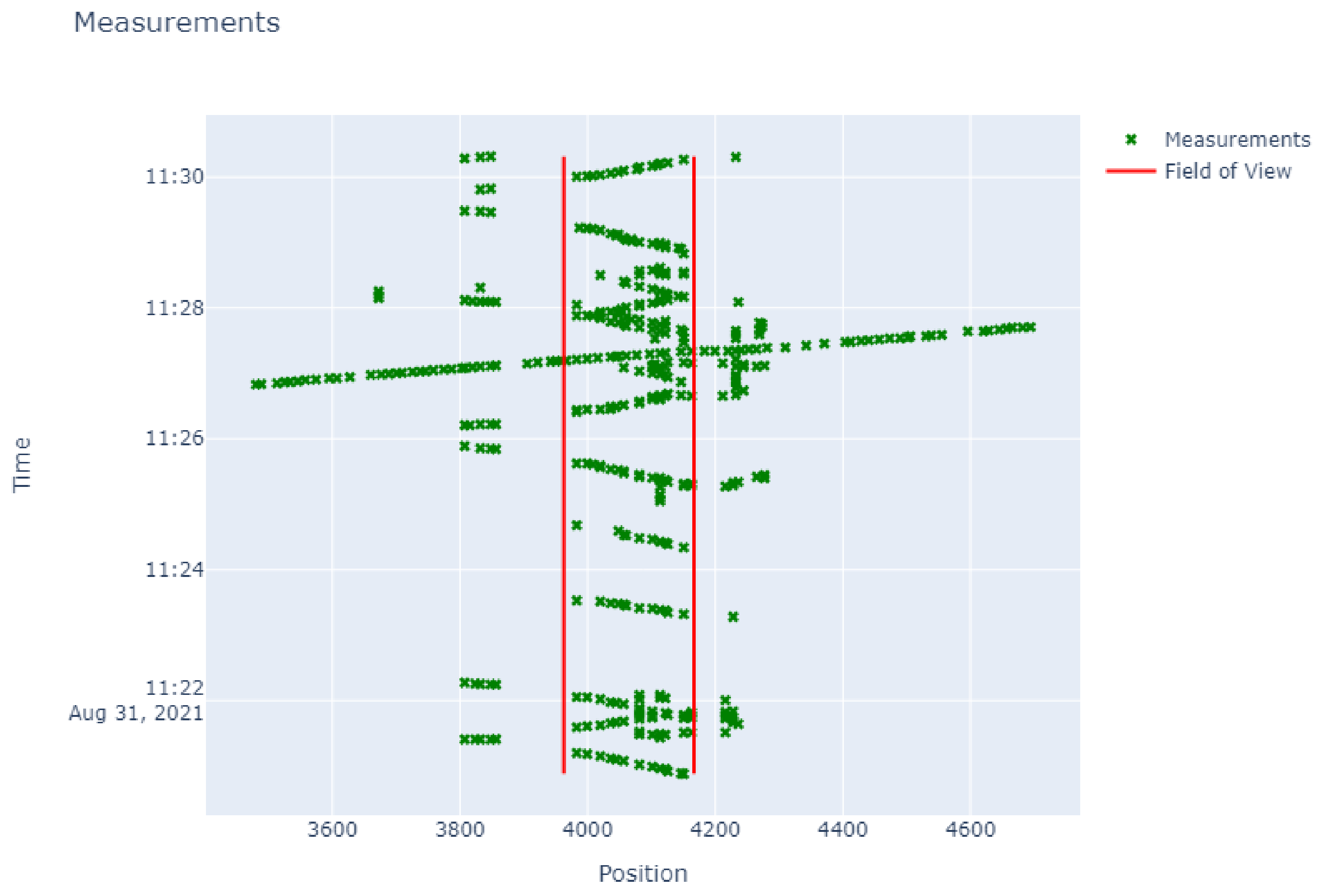


Figure 5.11: Signal picks used for tracking during the time window from 11:20 to 11:30 the 31st of August 2021. The distance unit is meters from Marienborg station.

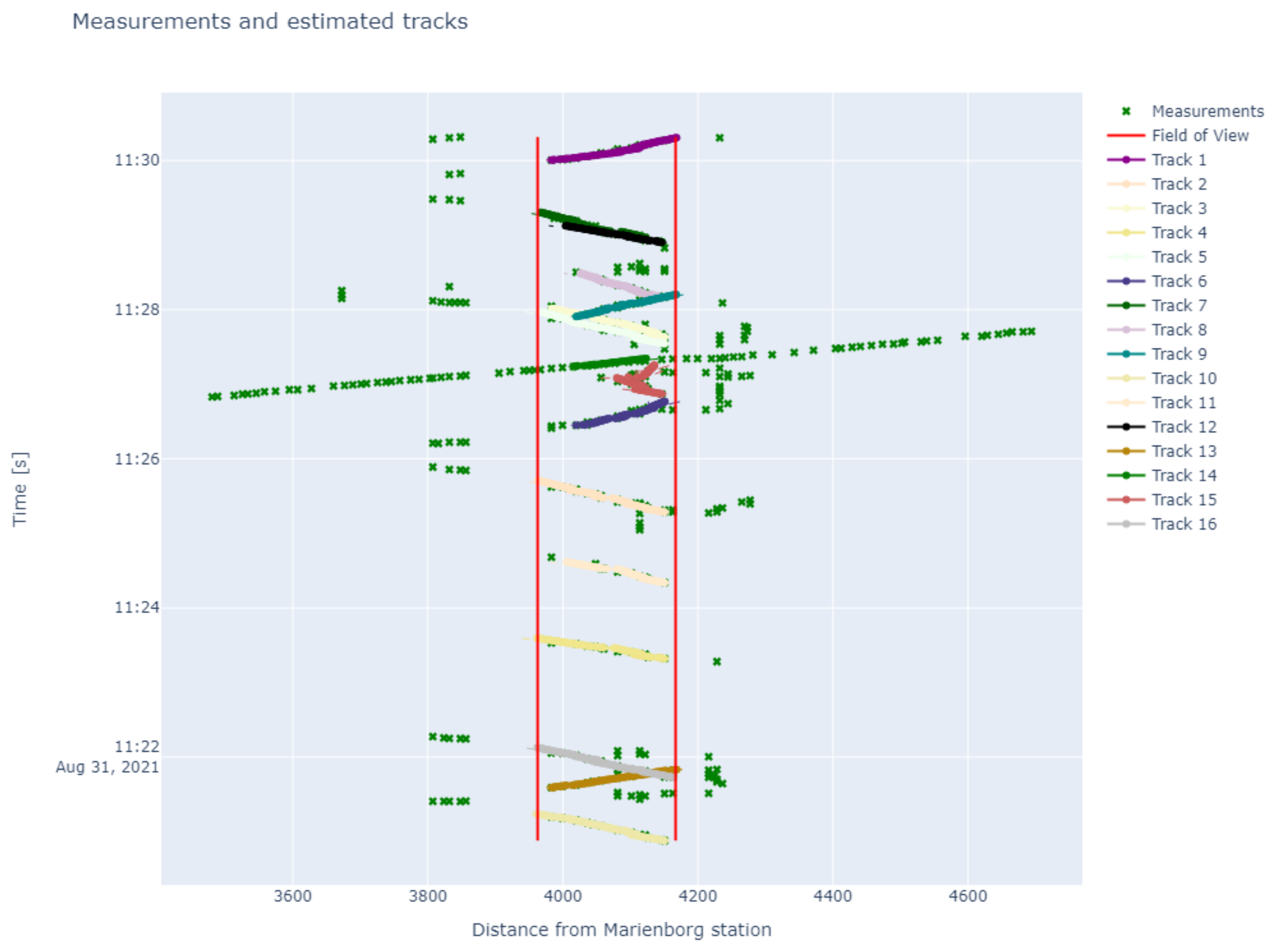


Figure 5.12: Signal picks and estimated positions of tracked cars and trains during the time window from 11:20 to 11:30 the 31st of August 2021

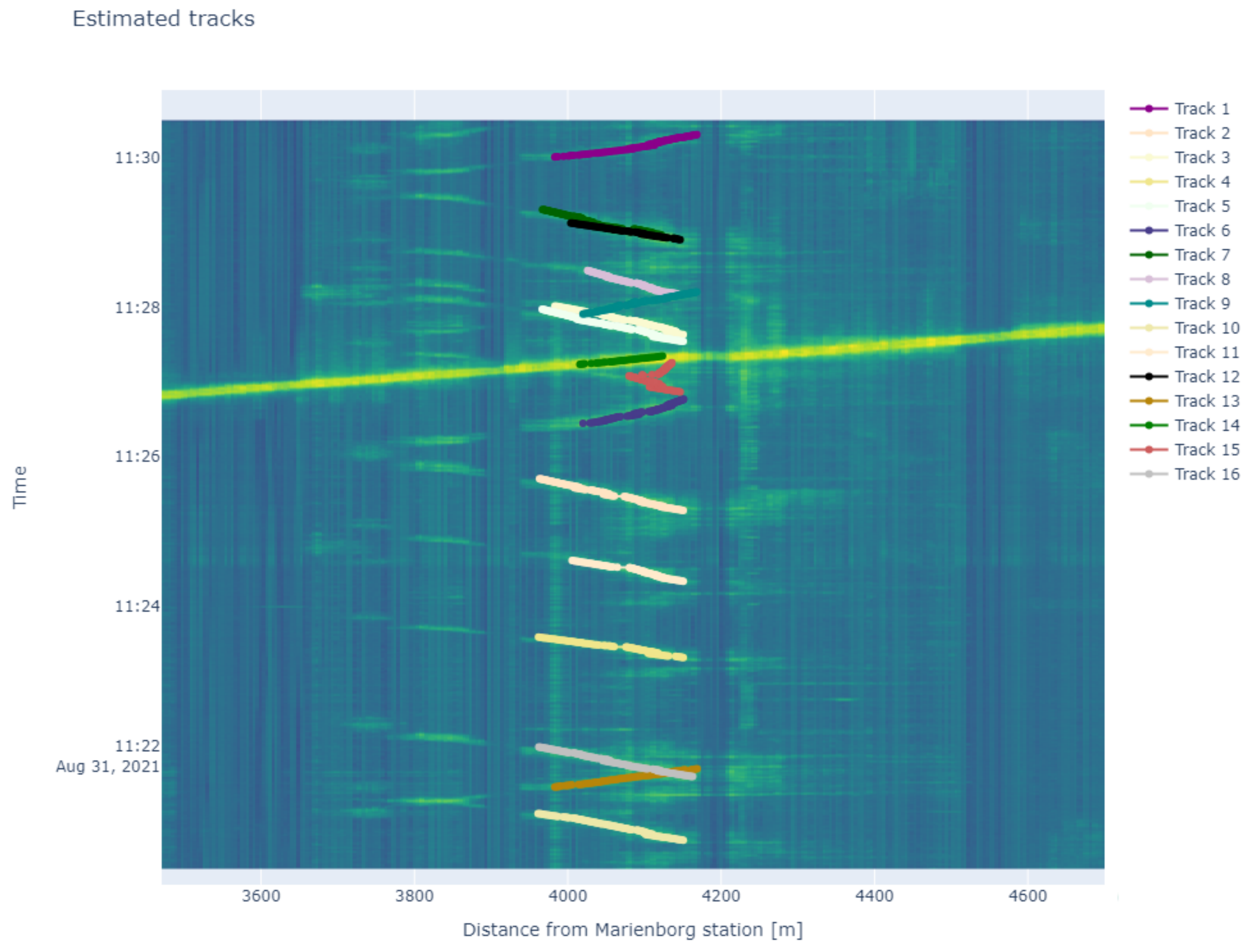


Figure 5.13: Heat map of fiber strain values, logged events and estimated positions of tracked cars and trains during the time window from 11:20 to 11:30 the 31st of August 2021

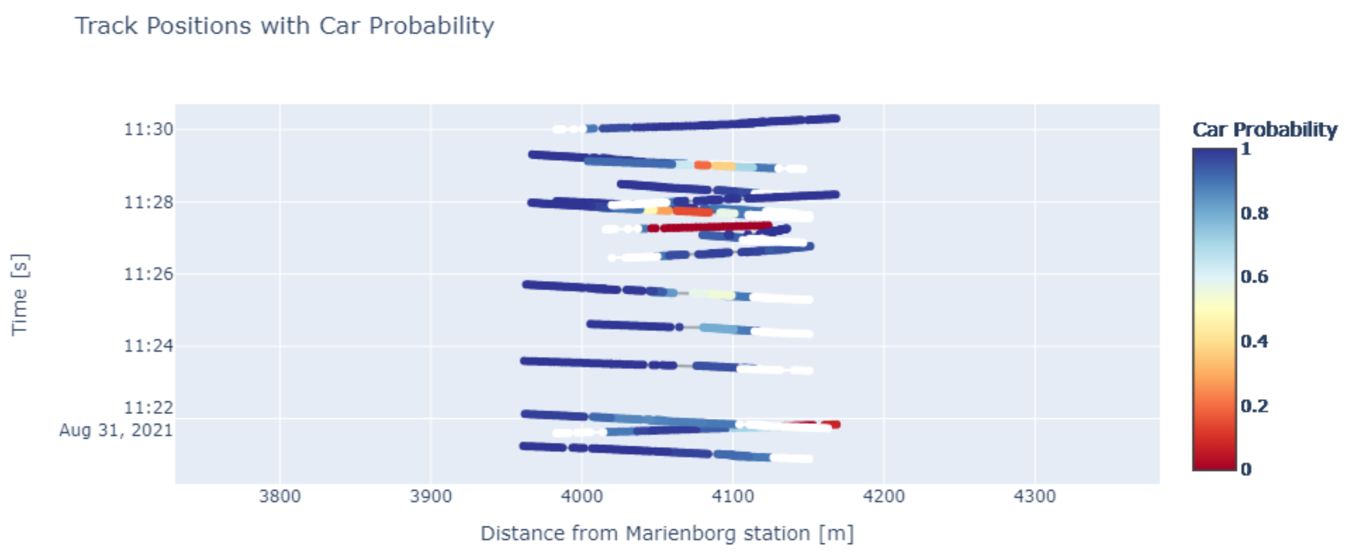


Figure 5.14: Car probability at each estimated position for all tracks during the time window from 11:20 to 11:30 the 31st of August 2021.

state. Since V_a and V_b are located in 30 km/h and 50 km/h speed zones respectively, one could argue that an entry point dependent initial velocity should have been used. This is something that could be considered in future work, but we consider a uniform initial velocity sufficient for this thesis. The general pattern that can be observed here is that the algorithm performs very good when there is little traffic, vehicles are far apart and when their initial velocity is close to the assumed prior velocity. Once the traffic gets more dense, with both long trains and many cars driving past each other at the same time, the fiber strain pattern gets more complex, and the algorithm occasionally run into problems like incorrect data association and missing measurements.

5.4 Tracking traffic during the entire day

We now run the algorithm through data corresponding to the entire day of 31st of August 2021. Figure 5.20 shows the number of tracked cars and trains each half hour during the entire day. As expected the traffic increases significantly from the morning hours and during the daytime, and decreases again around 18:30. The histogram shows that there are many more cars than trains, which is also expected. The traffic seems to peak from 12:30 to 14:00, which is somewhat unexpected, since we expected to see peaks around the rush hours, i.e. during the morning and afternoon. Furthermore, Figure 5.21 and 5.22 shows the average estimated velocities in every half hour during the entire day, for cars and trains respectively. Looking at the average velocities for cars the immediate trend that can be seen is that the average speed is close to the speed limit 30 km/h. This is especially true for north-going vehicles, which start in the 30 km/h speed limit zone and transitions into the 50 zone. Interestingly enough one can observe that the average south-going average speed is slightly higher than 30 in large parts of the day. This makes sense, since many vehicles passing from a higher speed zone into a lower one typically enters with a slightly higher velocity than the one indicated in the lower speed zone. Comparing average speeds of cars with trains, we see that the velocities of cars are more stable than the average train velocities. This means that the train velocities vary greatly throughout the day, which can be a sign that the tracking algorithm is failing on some objects. We would expect the average velocities of trains to be a bit higher than reported, and more stable than what can be seen from the figure.

Another thing worth mentioning is that the counts are highly sensitive to the choice of N_{init}. When we decrease this parameter, the counts in all time windows increases, since we allow for shorter tracks to be consider a true track. Similarly, when we increase N_{init}, we see a decrease in the count in each time window, since we require the tracks to be updated more frequently. However, the general traffic pattern throughout the day remain similar for several values of N_{init}. Since we found through visual inspection and experimentation that a good value for N_{init} was 5, we used this value also for counting tracks during the entire day.

Looking at the night hours, there is a somewhat strange number of trains passing in some time windows. We investigated this further, and discovered that the JPDA algorithm believes that there are 13 objects moving in both directions, which can be seen in Figure 5.23. We can observe that in reality this is a long freight train causing strong strain signals. This is likely one of the reasons for the unstable and somewhat unexpected average train velocities

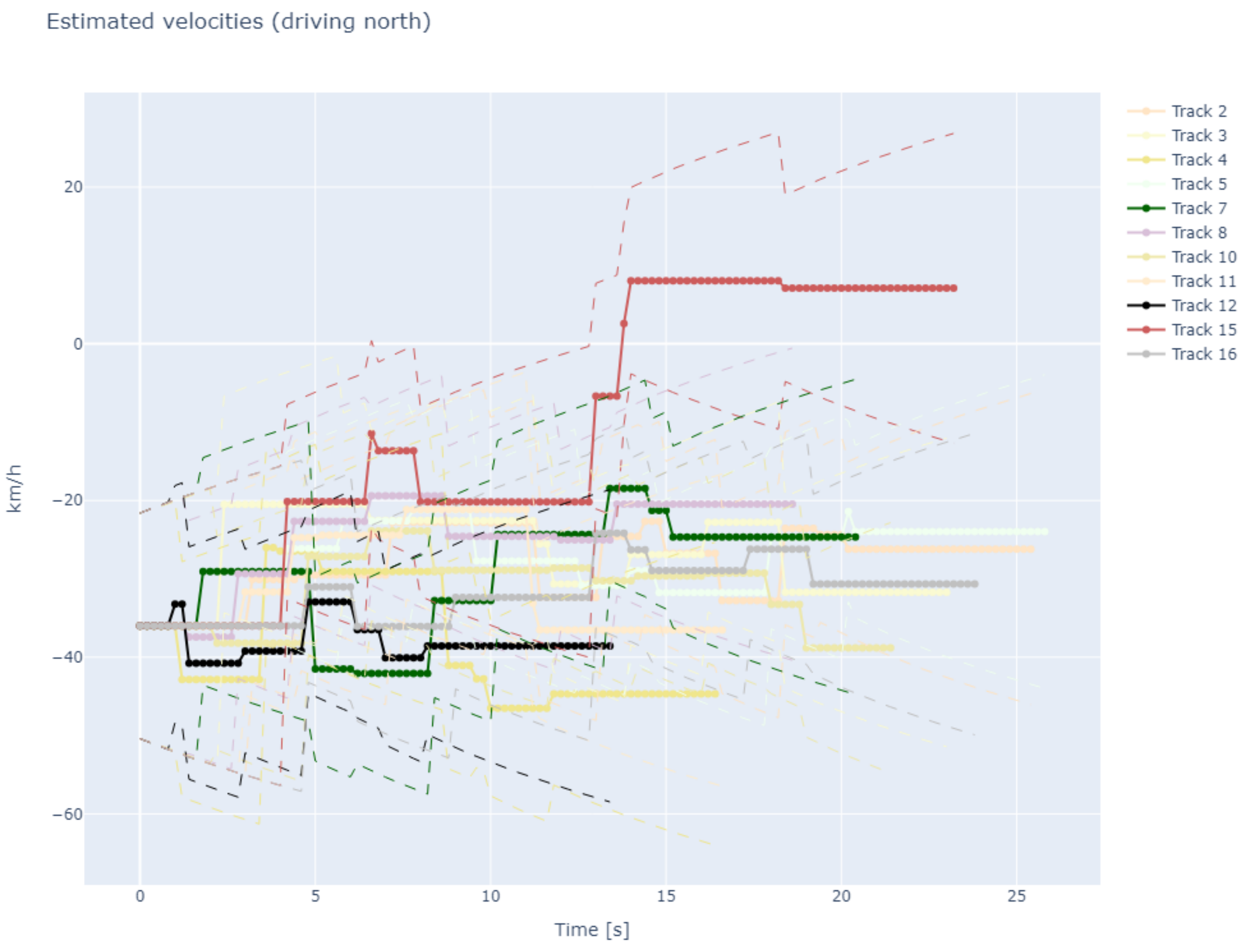


Figure 5.15: Estimated velocities of the tracked cars and trains driving north (towards Marienborg station) during the time window from 11:20 to 11:30 the 31st of August 2021.

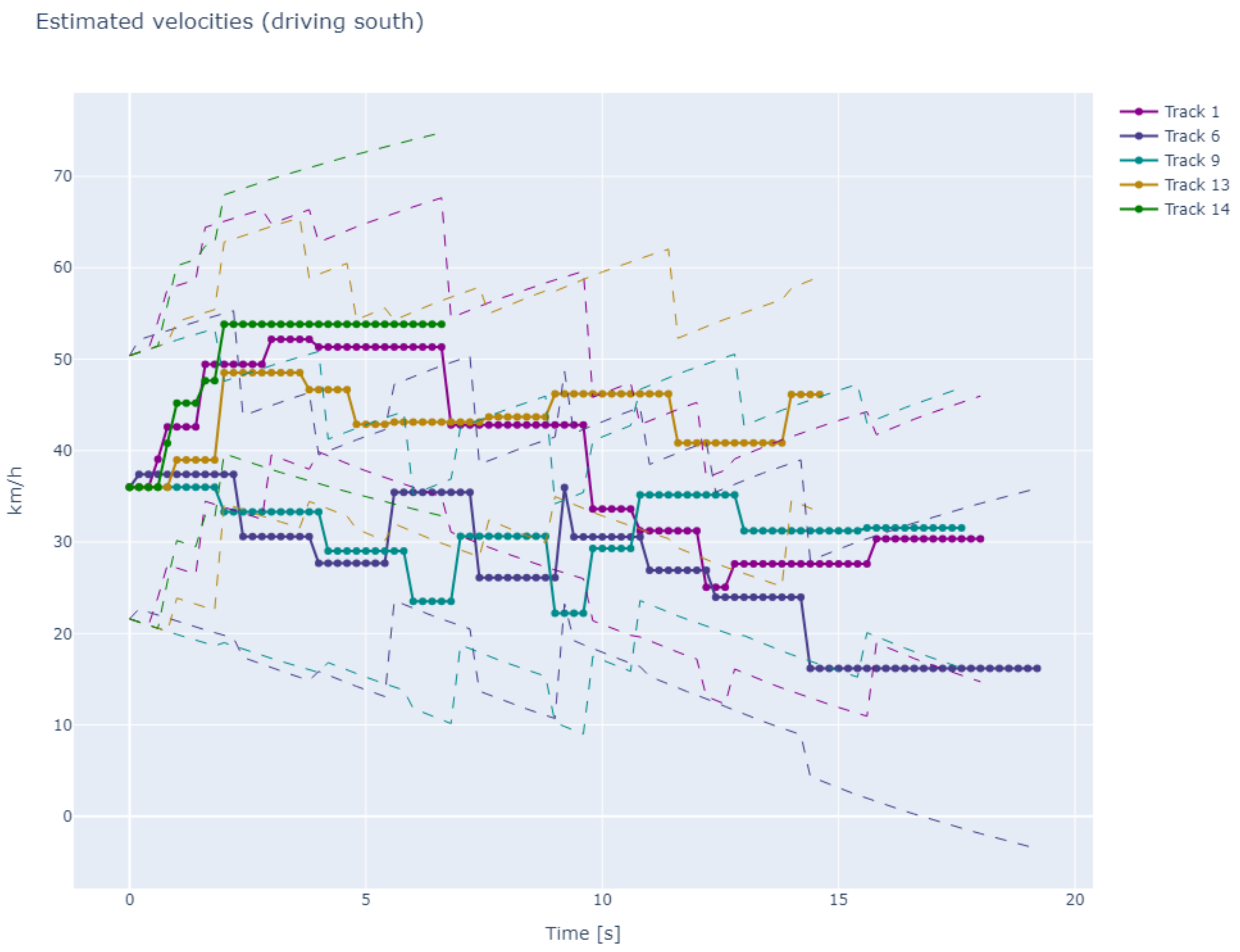


Figure 5.16: Estimated velocities of the tracked cars and trains driving south (away from Marienborg station) during the time window from 11:20 to 11:30 the 31st of August 2021

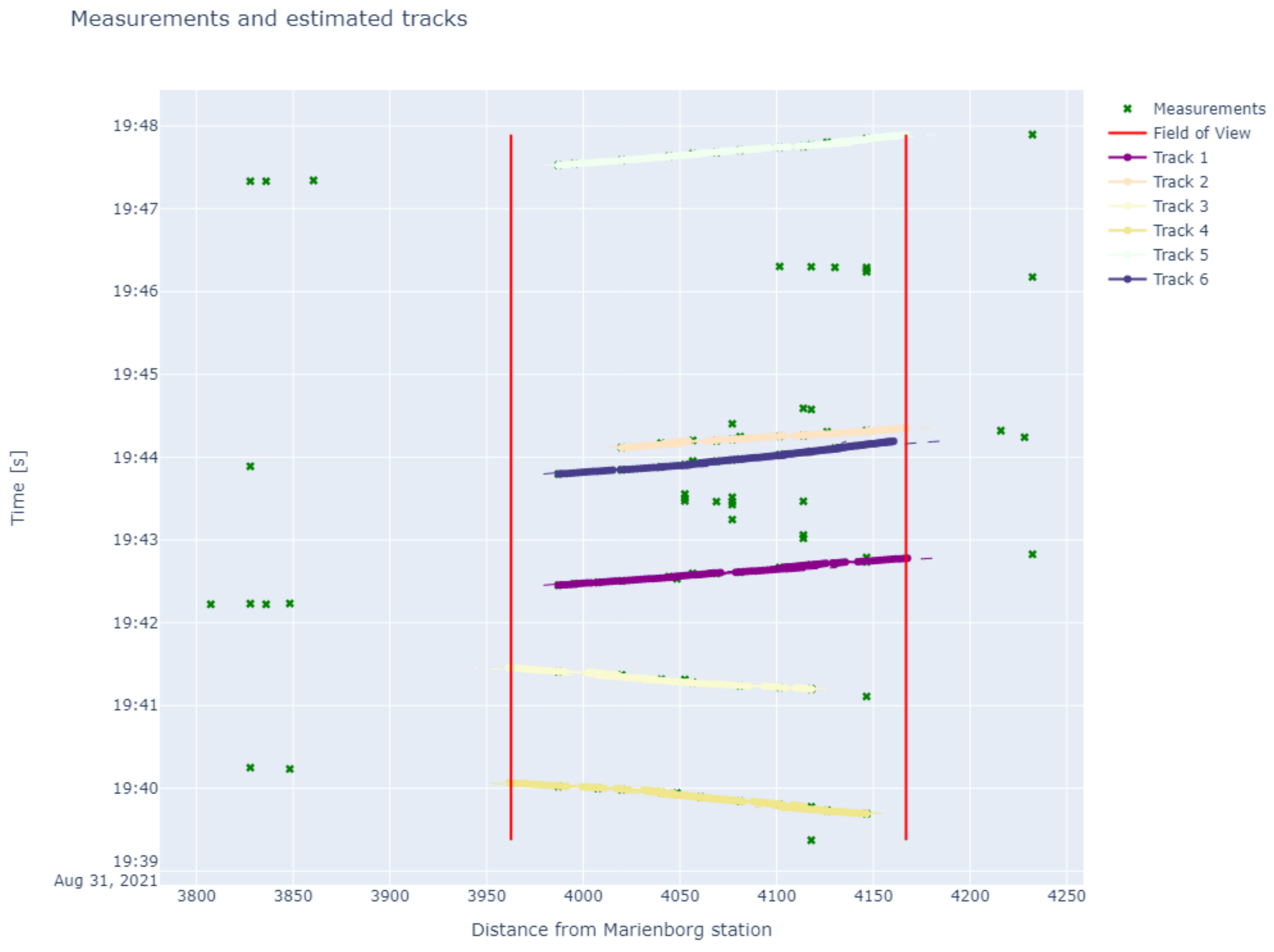


Figure 5.17: Signal picks and estimated positions of tracked cars and trains during the time window from 19:39 to 19:48 the 31st of August 2021

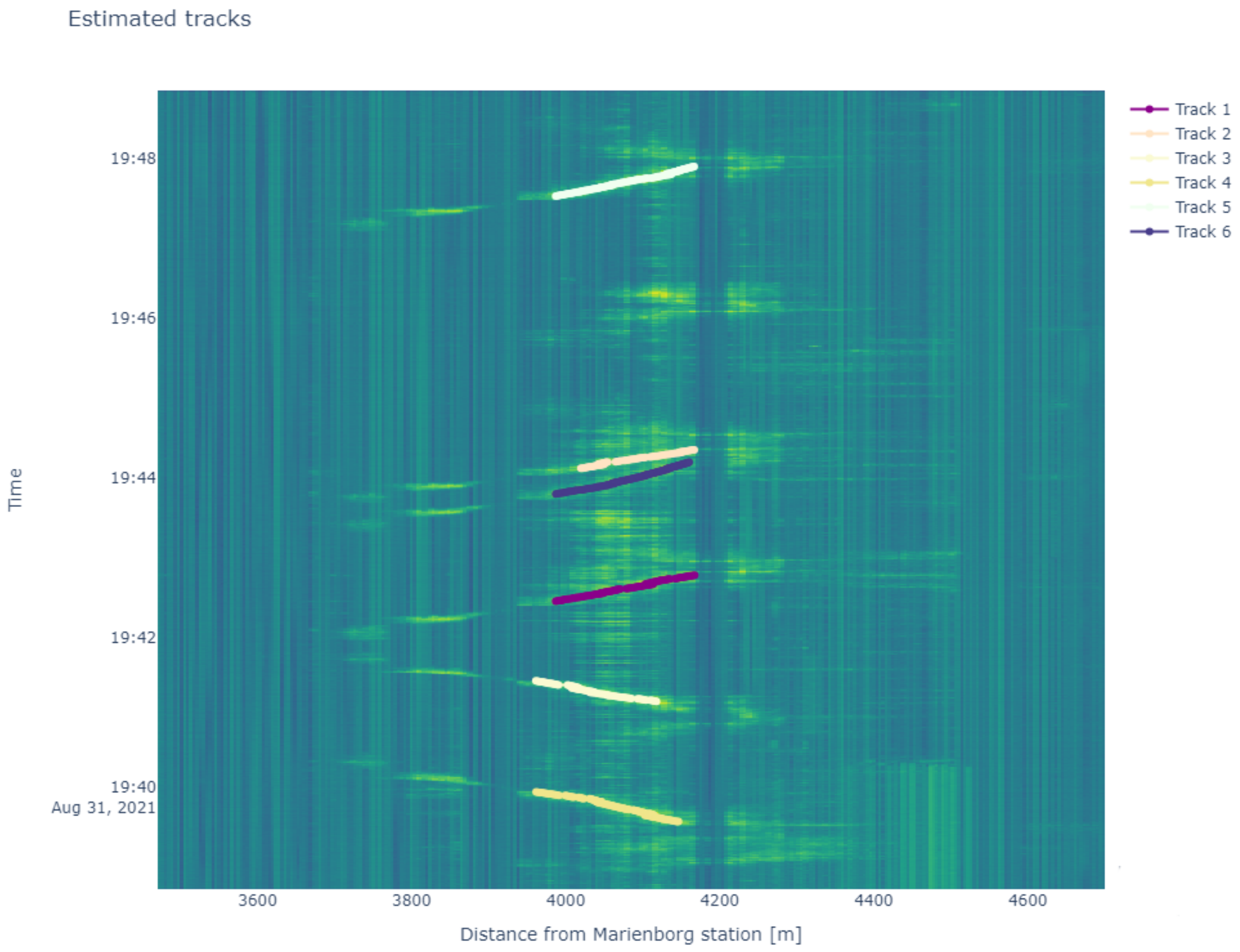


Figure 5.18: Heat map of fiber strain values, logged events and estimated positions of tracked cars and trains during the time window from 19:39 to 19:48 the 31st of August 2021

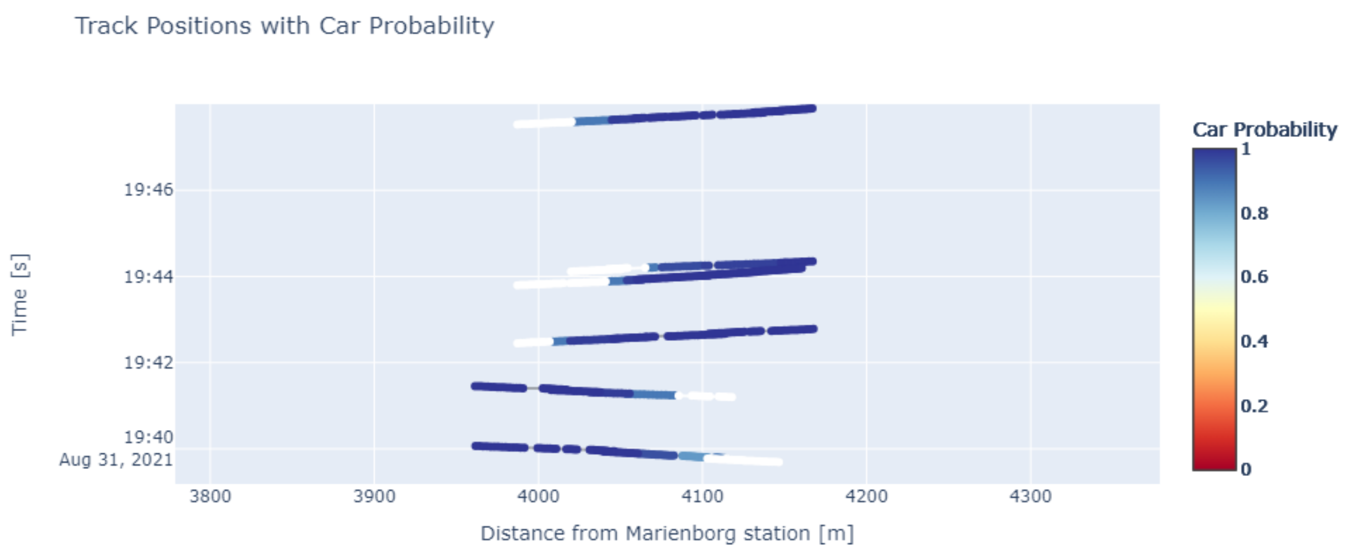


Figure 5.19: Car probability at each estimated position for all tracks during the time window from 19:39 to 19:48 the 31st of August 2021.

reported in Figure 5.22. There are mainly two reasons why the tracking fails here:

- The signal extraction algorithm produces too many picks: Since the train is very long, it will cause strain signals at the same spatial location over a longer time period. Therefore, we obtain a ”band” of signal picks for each spatial location, which in turn will confuse the JPDA algorithm. After some experimentation with different values of κ and ε, we found that this issue could be resolved by increasing the window size κ. However, it is clear that this increased κ would yield lower quality signal picks for smaller objects (cars), and would lead to poor car tracking performance. Therefore, it seems that the hyperparameter values for Algorithm 2 (Especially κ) must be chosen according to the object we wish to track.
- The JPDA algorithm is designed based on the assumption that the physical extent of the objects is limited to points. Therefore, in combination with bands of measurements for each spatial location, it can in theory believe that there are many objects, when in reality there is one physically extensive object causing many measurements.

5.5 Observed limitations of the algorithm

We have seen that the combination of the presented signal extraction algorithm and the JPDA filter have yielded great tracking results for cars and shorter passenger trains moving in the area around Selsbakk station. However, we have been able to identify cases in which this approach fails or performs poorly. We have gathered the observed limitations to the algorithm together with suggested remedies for these in Table 5.5.

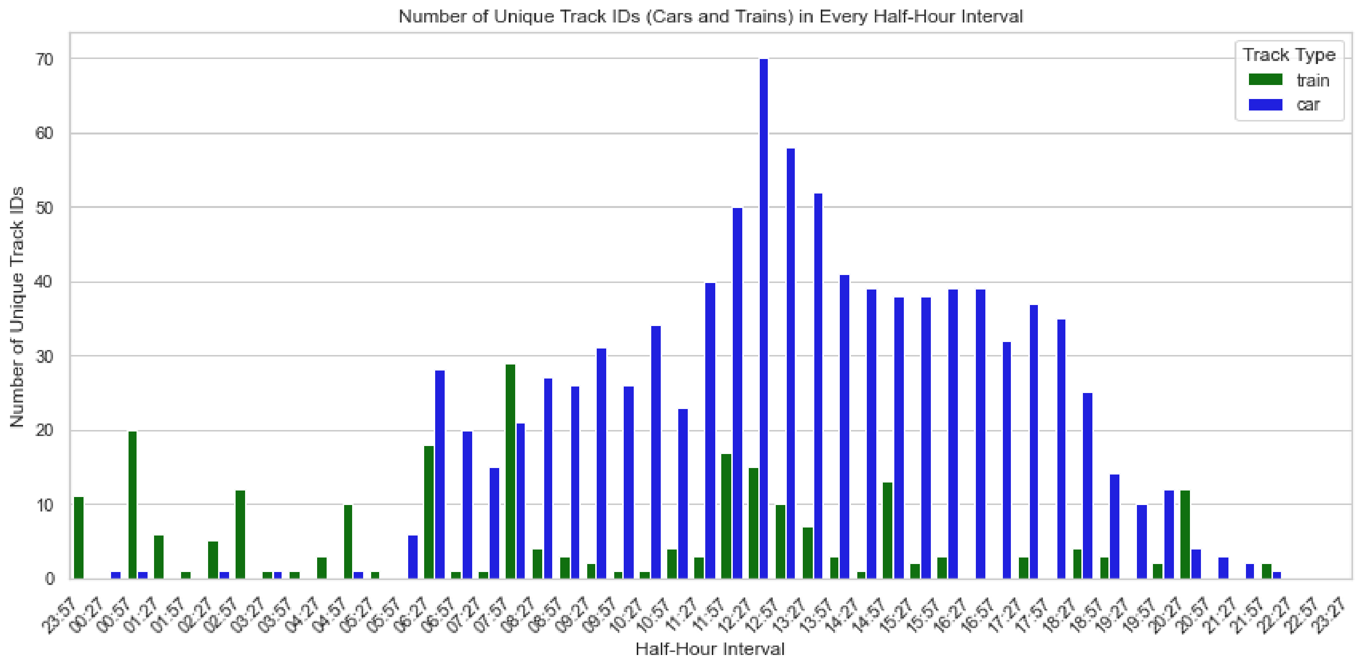


Figure 5.20: Number of tracked cars and trains during the entire day of 31st of August 2021, with $N_{init} = 5$.

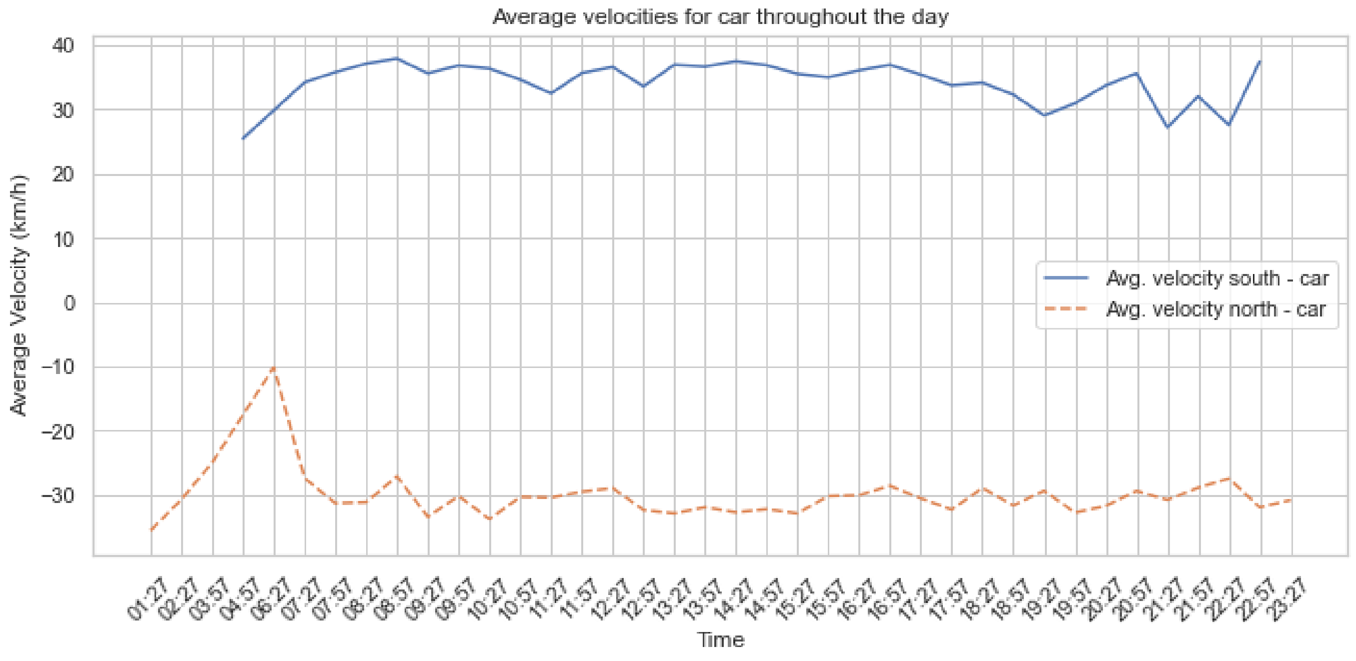


Figure 5.21: Average velocities of tracked cars driving at Selsbakk during the entire day, both north- and south-going vehicles.

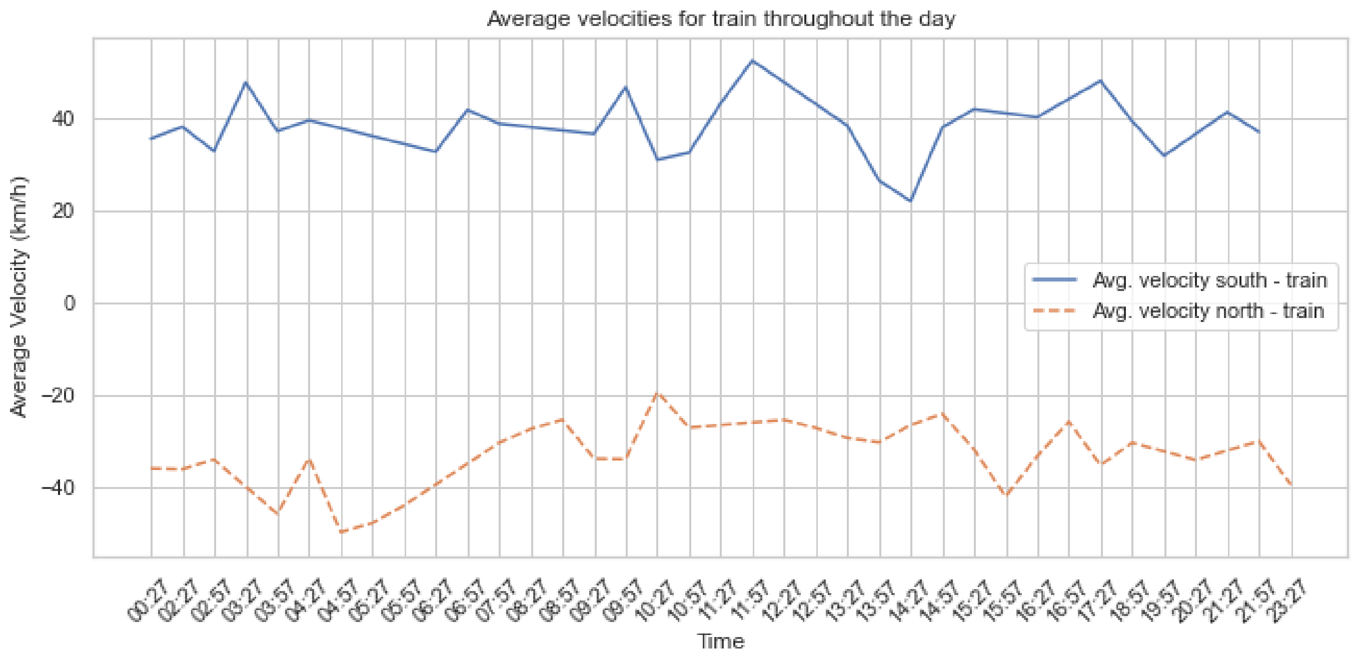


Figure 5.22: Average velocities of tracked trains driving at Selsbakk during the entire day, both north- and south-going trains.

Table 5.5: Observed limitations of the signal extraction algorithm (Algorithm 2) and the JPDA algorithm, and suggested remedies.

Limitation	Suggested solution
• Same prior state for all objects: This is an issue, since often trains pass at higher velocities than cars, and thus some trains might pass undetected.	• Separate priors for cars and trains, with a higher initial velocity for trains (Challenging, since we don't know if the object is a car or train beforehand).
• Unable to capture unexpected behavior of the objects like strong acceleration, or stopping for a while and then start moving again.	• The implementation of an IMM to account for multiple dynamic behavior regimes over time.
• Unable to effectively track objects of larger physical extent (like long freight trains), due to the inherent assumption of objects being points in the JPDA filter.	• An adaptive signal extraction algorithm that automatically adjusts ε, κ and a when bands of signal picks for each spatial location are detected **or:** The application of an EOT algorithm.

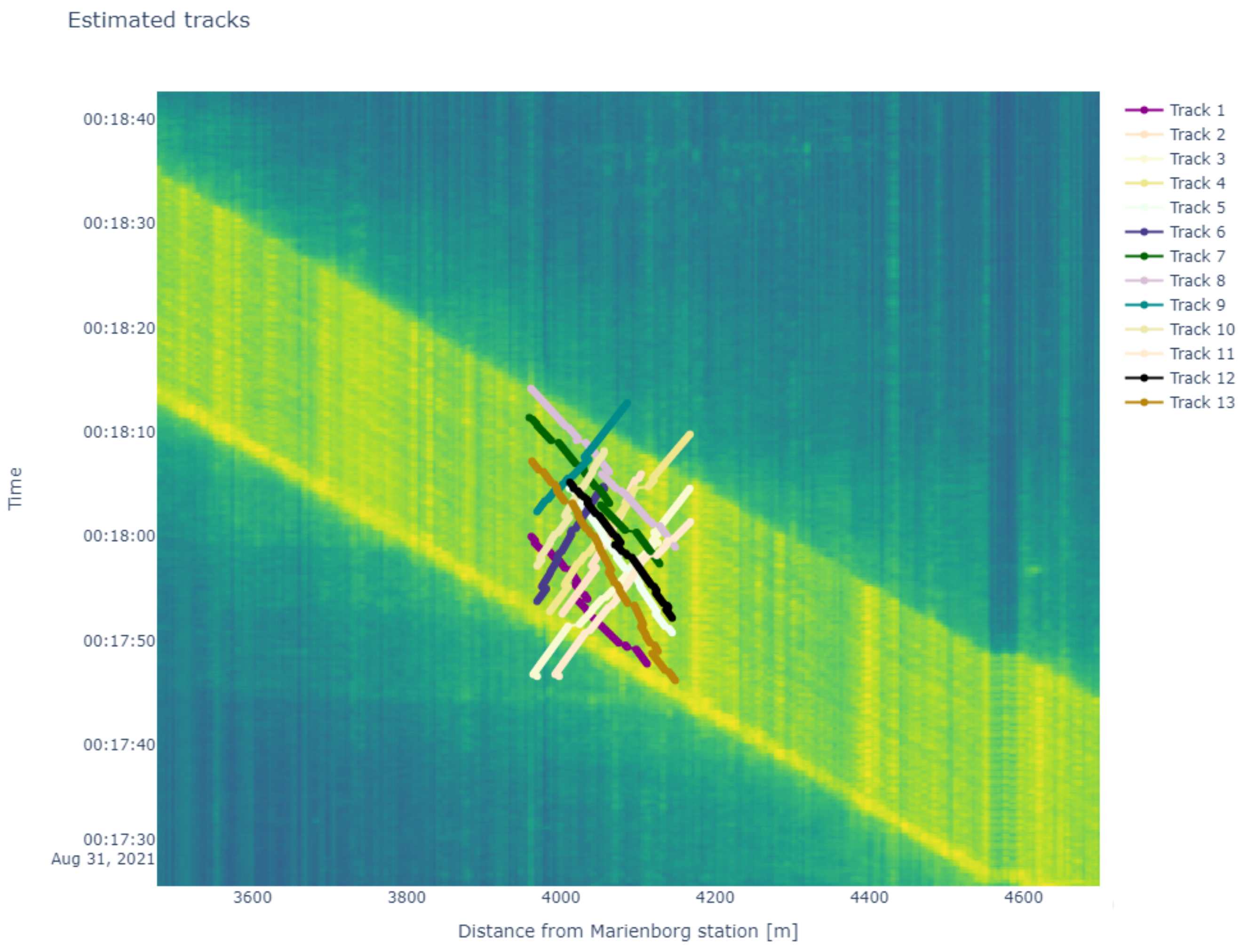


Figure 5.23: JPDA applied to a long train around midnight the 31st of August 2021, with $N_{init} = 5$.

Chapter 6

Conclusion

6.1 Concluding remarks

In this thesis we have presented theory for Bayesian filtering and MOT, and showed how this framework can be combined with DAS data. Due, among other factors, to the point assumption imposed on the tracked objects, it has been necessary to extract signal picks from the log-RMS data CGF provided us. We presented a custom algorithm combining moving average smoothing and DBSCAN clustering to extract these picks, to be used as object measurements in the tracking application. We presented the JPDA algorithm as a feasible solution to the MOT problem, and incorporated functionality for initiation and deletion of objects. Since objects could be both cars and trains, we used the signal pick amplitudes of each tracked object to infer the probability of an object being a car or train at each time step, with the help of likelihood models trained on the logged data we were provided.

The JPDA tracker and Bayes classifier was applied to three different traffic scenarios at Selsbakk: during dense traffic with logged vehicle passings, during dense traffic without logged vehicle passings, and during light traffic without logged vehicle passings. We compared the tracked objects to the logged vehicle passings, and found that the tracker successfully identified and tracked 19 out of 20 vehicles and trains. It successfully classified all objects correctly. In dense traffic without logged events the tracker performed well, but with some mistakes. During light traffic the tracker was able to identify and track all objects present. After multiple experiments we found that the tracker with the particular parameter setup performed well in light and slightly dense traffic for objects of small physical extent (cars and short trains) whose velocity entering the FOV is comparable to the prior velocity defined in the transition model. This was expected as these are conditions that meet the theoretical assumptions on the objects. We saw that objects of large physical extent (long trains) or objects moving at very high or very low speed failed to be tracked with the same parameter setup. We performed vehicle and train counting by counting the number of car and train tracks for each half hour during the day 31st of August 2021. We found that the number of cars increased during daytime, which was expected. We also calculated average velocities of cars and trains moving north and south for the entire day. The car velocity patterns reflected the speed limits at Selsbakk, and the train velocity estimates were more unstable.

One of the most important discoveries we made was that the tracking setup with only one transition model was not sufficient for identifying and tracking all objects present in the DAS data. We saw that we could adjust hyperparameters for signal extraction and parameters in JPDA to track a certain group of objects (long trains, fast moving cars etc.), but this parameter configuration would yield poor results for objects of smaller size or lower velocities. With this in mind, we conclude that DAS data has large potential when it comes to target tracking, and that the method proposed in this thesis works well on DAS data, as long as the objects that are being tracked to a certain extent fulfills the theoretical assumptions that are made.

6.2 Future work and extensions

There are many options as to how the work in this thesis can be further developed. As mentioned earlier, the implementation of an EOT algorithm to track the objects would definitely be useful since we get rid of the point assumption on the objects. Since we have seen that the objects present in the DAS data are of varying size, this would allow for a more realistic modeling of the objects. In addition, we have seen that cars and trains often move at different speeds, and that both objects can accelerate. Therefore the implementation of an IMM model could help facilitate the shifting dynamics of objects, not only to distinguish between car and train dynamics, but also intra-class dynamics like acceleration.

We have only worked with batches of historical DAS data in this thesis. A future development of this application could be the implementation of a real-time decision support system for train or car traffic monitoring. This is feasible since the JPDA filter is an online algorithm that processes measurements in real time, and the signal extraction algorithm presented in this thesis can (in theory) be adapted to pick signals in real-time. Of course, this would require the smoothing window size κ to be 0 or quite small, to avoid introducing a time lag.

To take a step further, we believe the application of DL and neural networks could be an alternative approach for event detection in the DAS data and tracking. In this case one would not be restricted to the assumptions made by the traditional tracking algorithms, and the different object dynamics would be features learned by an AI model. A downside to this approach is that a large amount of training data would be required to achieve satisfactory results.

Bibliography

Bar-Shalom, Y., & Li, X.-R. (1995). *Multitarget-multisensor tracking: Principles and techniques* (Vol. 19). YBs Storrs, CT.

Bertsekas, D. P. (1979). A distributed algorithm for the assignment problem. *Lab. for Information and Decision Systems Working Paper, MIT.*

Dingler, S. (2022). State estimation with the interacting multiple model (imm) method. *arXiv preprint arXiv:2207.04875.*

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *kdd*, *96*(34), 226–231.

Fredriksen, S. (2022). *Object tracking with applications to car tracking using fiber-optic signal data* (Project report in TMA4500). Department of Mathematical Sciences NTNU – Norwegian University of Science and Technology.

Gardner Jr, E. S. (1985). Exponential smoothing: The state of the art. *Journal of forecasting*, *4*(1), 1–28.

Gorshkov, B. G., Yüksel, K., Fotiadi, A. A., Wuilpart, M., Korobko, D. A., Zhirnov, A. A., Stepanov, K. V., Turov, A. T., Konstantinov, Y. A., & Lobach, I. A. (2022). Scientific applications of distributed acoustic sensing: State-of-the-art review and perspective. *Sensors*, *22*(3), 1033.

Granstrom, K., Baum, M., & Reuter, S. (2016). Extended object tracking: Introduction, overview and applications. *arXiv preprint arXiv:1604.00970.*

Härdle, W., & Vieu, P. (1992). Kernel regression smoothing of time series. *Journal of Time Series Analysis*, *13*(3), 209–232.

Hernández, P. D., Ramírez, J. A., & Soto, M. A. (2022). Deep-learning-based earthquake detection for fiber-optic distributed acoustic sensing. *Journal of Lightwave Technology*, *40*(8), 2639–2650.

Horst, J. v., Panhuis, P. i., Al-Bulushi, N., Deitrick, G., Mustafina, D., Hemink, G., Groen, L., Potters, H., Mjeni, R., Awan, K., et al. (2015). Latest developments using fiber optic based well surveillance such as distributed acoustic sensing (das) for downhole production and injection profiling. *SPE Kuwait Oil and Gas Show and Conference*, SPE–175211.

Hough, P. V. (1959). Machine analysis of bubble chamber pictures. *International Conference on High Energy Accelerators and Instrumentation, CERN, 1959*, 554–556.

Hubbard, P. G., Xu, J., Zhang, S., Dejong, M., Luo, L., Soga, K., Papa, C., Zulberti, C., Malara, D., Fugazzotto, F., et al. (2021). Dynamic structural health monitoring of a model wind turbine tower using distributed acoustic sensing (das). *Journal of Civil Structural Health Monitoring*, *11*(3), 833–849.

In't Panhuis, P., den Boer, H., Van Der Horst, J., Paleja, R., Randell, D., Joinson, D., McIvor, P., Green, K., & Bartlett, R. (2014). Flow monitoring and production profiling using das. *SPE Annual Technical Conference and Exhibition?*, SPE–170917.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems.

King, R. G., & Rebelo, S. T. (1993). Low frequency filtering and real business cycles. *Journal of Economic dynamics and Control*, *17*(1-2), 207–231.

Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval research logistics quarterly*, *2*(1-2), 83–97.

L. Svensson, K. (n.d.). Multi-object tracking for automotive systems [URL: `https://www.edx.org/course/multi-object-track`

Liu, X., Wang, C., Shang, Y., Wang, C., Zhao, W., Peng, G., & Wang, H. (2017). Distributed acoustic sensing with michelson interferometer demodulation. *Photonic Sensors*, *7*, 193–198.

Macaulay, F. R. (1931). Introduction to" the smoothing of time series". In *The smoothing of time series* (pp. 17–30). NBER.

Murty, K. G. (1968). An algorithm for ranking all the assignments in order of increasing cost. *Operations research*, *16*(3), 682–687.

Pal, S. K., Pramanik, A., Maiti, J., & Mitra, P. (2021). Deep learning in multi-object detection and tracking: State of the art. *Applied Intelligence*, *51*, 6400–6429.

Pierskalla, W. P. (1968). The multidimensional assignment problem. *Operations Research*, *16*(2), 422–431.

Rahman, M. A., Taheri, H., Dababneh, F., Karganroudi, S. S., & Arhamnamazi, S. (2024). A review of distributed acoustic sensing applications for railroad condition monitoring. *Mechanical Systems and Signal Processing*, *208*, 110983.

Reid, D. (1979). An algorithm for tracking multiple targets. *IEEE transactions on Automatic Control*, *24*(6), 843–854.

Rockafellar, R. T., & Wets, R. J.-B. (2009). *Variational analysis* (Vol. 317). Springer Science & Business Media.

Rørstadbotnen, R. A., Eidsvik, J., Bouffaut, L., Landrø, M., Potter, J., Taweesintananon, K., Johansen, S., Storevik, F., Jacobsen, J., Schjelderup, O., et al. (2023). Simultaneous tracking of multiple whales using two fiber-optic cables in the arctic. *Frontiers in Marine Science*, *10*, 1130898.

Särkkä, S. (2013). *Bayesian filtering and smoothing*. Cambridge university press.

Taweesintananon, K., Landrø, M., Brenne, J. K., & Haukanes, A. (2021). Distributed acoustic sensing for near-surface imaging using submarine telecommunication cable: A case study in the trondheimsfjord, norway. *Geophysics*, *86*(5), B303–B320.

Thomas, P. J., Heggelund, Y., Klepsvik, I., Cook, J., Kolltveit, E., & Vaa, T. (2023). The performance of distributed acoustic sensing for tracking the movement of road vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 1–14.

Tribaldos, V. R., Ajo-Franklin, J. B., Dou, S., Lindsey, N. J., Ulrich, C., Robertson, M., Freifeld, B. M., Daley, T., Monga, I., & Tracy, C. (2021). Surface wave imaging using distributed acoustic sensing deployed on dark fiber: Moving beyond high-frequency noise. *Distributed Acoustic Sensing in Geophysics: Methods and Applications*, 197–212.

Vo, B.-n., Mallick, M., Bar-Shalom, Y., Coraluppi, S., Osborne, R., Mahler, R., & Vo, B.-t. (2015). Multitarget tracking. *Wiley encyclopedia of electrical and electronics engineering*, (2015).

Wang, M., Wan, Q., & You, Z. (2008). A gate size estimation algorithm for data association filters. *Science in China Series F: Information Sciences*, *51*(4), 425–432.

Wiesmeyr, C., Coronel, C., Litzenberger, M., Döller, H. J., Schweiger, H.-B., & Calbris, G. (2021). Distributed acoustic sensing for vehicle speed and traffic flow estimation. *2021 IEEE international intelligent transportation systems conference (ITSC)*, 2596–2601.

Wiesmeyr, C., Litzenberger, M., Waser, M., Papp, A., Garn, H., Neunteufel, G., & Döller, H. (2020). Real-time train tracking from distributed acoustic sensing data. *Applied Sciences*, *10*(2), 448.