LIDAR Extended Object Tracking of a Maritime Vessel Using an Ellipsoidal Contour Model

Kristian Amundsen Ruud

Master of Science in Physics and Mathematics

Submission date:	July 2018
Supervisor:	Jo Eidsvik, IMF
Co-supervisor:	Edmund Brekke, ITK

Norwegian University of Science and Technology
Department of Mathematical Sciences

Abstract

Extended object tracking (EOT) have numerous applications and can be integrated in autonomous systems like self-driving cars or autonomous surface vehicles. These systems can be improved by using robust tracking algorithms that accurately estimate the position, velocity and extent of surrounding targets from sensor data. EOT is relevant for many different sensors, and the methods developed are based on which sensor that is used. In this thesis LIDAR is used to track a single target in a maritime environment that have a shape which can be approximated by an ellipse. The sensor detections of the target will often be noisy, and it is crucial to model the measurement uncertainties in a proper way. Both measurements from target and other objects (clutter) need to be included in the probabilistic modelling, where the data association problem is important to solve.

In this thesis we present and implement two different methods designed for tracking a single target from LIDAR data. To model clutter measurements the generalized probabilistic data association (GPDA) filter is used together with the filter methods. Through simulations of a single elliptical target with and without clutter measurements, it is found that the method using contour measurement modelling is performing considerably better than the random matrix method. This result is obtained by considering both absolute error and covariance consistency of the two methods. The contour measurement method is also superior when testing the methods on real LIDAR data of the Munkholm boat taken from Ravnkloa in Trondheim, by looking at track plots, extent estimation errors and innovation statistics.

Sammendrag

Målfølging av utvidede objekter (EOT) har mange anvendelser og kan bli integrert i autonome systemer som for eksempel selvstyrte biler eller båter. Disse systemene kan forbedres ved å bruke robuste algoritmer som nøyaktig estimerer posisjon, fart og utstrekning på andre mål fra sensordata. EOT er relevant for mange forskjellige sensorer, og metodene som er utviklet er basert på hvilken sensor som er brukt. I denne oppgaven brukes LIDAR til å følge et enkelt mål til sjøs som har en form som kan tilnærmes en ellipse. Sensorens målinger av målet vil ofte ha støy i seg, og det er viktig å modellere måleusikkerhetene på en korrekt måte. Både målinger fra mål og andre objekter (falske) må inkluderes i den probabilistiske modelleringen, der data-assosiasjonsproblemet er viktig å løse.

I denne oppgaven presenteres og implementeres to forskjellige metoder som er designet for å følge et enkelt mål fra LIDAR-data. For å modellere falske målinger brukes det generaliserte probabilistiske data-assosiasjon-filteret (GPDA) sammen med filtermetodene. Gjennom simuleringer av et enkelt elliptisk mål, med og uten falske målinger, blir det vist at metoden basert på konturmodellering av målingene er betydelig bedre enn tilfeldig matrise-metoden. Dette er et resultat basert på absoluttfeil og kovarians konsistenthet hos de to metodene. Konturmetoden er også best når man tester metodene på ekte LIDAR-data fra Munkholm-båten i Ravnkloa, Trondheim, ved å se på målfølgingsfigurer, estimeringsfeil for utstrekning og innovasjonsstatistikker.

Preface

This thesis is the final product of the 5-year Master of Science degree in Physics and Mathematics at NTNU. I have chosen industrial mathematics and statistics as specialization, and through my courses I have gained knowledge in several statistical disciplines like stochastical processes, linear regression, time series and spatial statistics.

The topic of this work was chosen based on my interest in autonomous vehicles and the statistical problems related to making these. I was fortunate to get the opportunity of writing my thesis on this subject and I have learned a lot about target tracking and cybernetics through the work. A special thanks goes to my supervisors Jo Eidsvik and Edmund Førland Brekke from the departments for mathematics and cybernetics at NTNU respectively. They have provided me with useful advice through the writing period, and their expertise in statistics and cybernetics has been important for this thesis. Through this collaboration we have delivered a contribution to the extended object tracking research field, and submitted a paper to the sensor fusion conference in Bonn, Germany.

I hope this thesis can inspire further research in extended object tracking with the LIDAR sensor, and I have certainly enjoyed working with it. I want to thank Vegard Kamsvåg from the cybernetics MSc program for data gathering and processing of the LIDAR data used in this thesis. I also want to thank my partner, friends and family for all support through the semester, and my fellow students at the industrial mathematics program for discussions about both relevant and irrelevant matters.

Trondheim, June 2018 Kristian Amundsen Ruud

Table of Contents

Abbreviations

EOT	Extended object tracking
LIDAR	Light detection and ranging
IR	Infrared
pdf	Probability density function
EKF	Extended Kalman filter
CEKF	Contour EKF
PHD	Probability hypothesis density
RFS	Random finite sets
PDA	Probabilistic data association
GPDA	Generalized PDA
C-GPDA	Contour GPDA
JPDA	Joint PDA
RMSE	Root mean square error
NEES	Normalized estimation error squared
NIS	Normalized innovation squared

Nomenclature

T	Number of time steps
k	Index for time steps, $k = 0, ..., T$. $(\cdot)_k$ means at time step k
t_k	Time at time step k
Δt_k	Time sampling interval, $\Delta t_k = t_k - t_{k-1}$
(x, y)	Cartesian positional coordinates
(v_x, v_y)	Velocity in x- and y-direction
(r, θ)	Polar positional coordinates (range and bearing)
(a, b)	Major and minor axes of the target ellipse
ϕ	Heading of target ellipse, $\phi = \arctan(v_y/v_x)$
γ	Course of target ellipse, $\gamma = \arctan\left(\frac{xv_y - yv_x}{xy + v_x v_y}\right)$
$\mathbf{x}_k$	State vector
$\mathbf{z}_k$	Measurement vector
n_x, n_z	Sizes of state and measurement vector
m_k	Number of total measurement points
n_k^t	Number of measurements from target
n_k^c	Number of measurements from clutter
$\mathbf{z}_k^j$	Measurement point given as (x_k^j, y_k^j) for $j = 1, ..., m_k$
$\mathbf{Z}_k$	Measurement set, $\lvert\mathbf{Z}_k\rvert = m_k$, containing the measurements $\mathbf{z}_k^j$
Θ_k	Set of measurements from target, $\lvert\Theta_k\rvert = n_k^t$
K_k	Set of measurements from clutter, $\lvert K_k\rvert = n_k^c$
$\mathbf{Z}_{1:k}$	Set of measurement sets from time step 1 to k
$\mathbf{X}_k$	Extent matrix of target ellipse
$s(t), r(t)$	Transmitted and reflected sensor signals at time t
$\mathbf{F}_k$	Dynamical model matrix at time step k
$\mathbf{f}(\cdot)$	Dynamic model function
$\mathbf{H}_k$	Measurement model matrix at time step k
$\mathbf{h}(\cdot)$	Measurement model function
$\mathbf{y}_k^j(\cdot)$	Predicted measurement function
$\mathbf{y}_k^j$	Predicted measurements
$\mathbf{z}_k^*, \mathbf{y}_k^*$	Vertical vectorial concatenations of $\mathbf{z}_k^j$ and $\mathbf{y}_k^j$
$\mathbf{R}_k^*$	Block diagonal measurement noise covariance matrix
$\mathbf{H}_\mathbf{x}(\cdot)$	Jacobian matrix with respect to the state vector $\mathbf{x}$
$\mathbf{q}_k$	Process noise vector with length n_x
$\mathbf{Q}_k$	Covariance matrix for the process noise $\mathbf{q}$
$\mathbf{m}_k$	Filtered state vector
$\mathbf{M}_k$	Filtered extent matrix
$\mathbf{P}_k$	Filtered covariance matrix
$\mathbf{S}_{k\|k-1}, \mathbf{W}_{k\|k-1}$	Scalar innovation and gain matrix EKF

η	Degrees of freedom in random matrix approach
η^*, τ	Extension evolution parameters in random matrix approach
$\mathbf{R}^{\text{RM}}$	Measurement noise covariance matrix in random matrix
$\bar{\mathbf{z}}_k$, $\bar{\mathbf{Z}}_k$	Centroid measurement and corresponding scattering matrix
$\bar{\mathbf{z}}^*_k$, $\bar{\mathbf{Z}}^*_k$	LIDAR centroid measurement and corresponding scattering matrix
$\mathbf{N}_{k\|k-1}$	Innovation matrix in random matrix method
α_k, ν_k	Filtered degrees of freedom in random matrix method, $\alpha_k = \nu_k - 3$.
$\mathbf{S}^*_{k\|k-1}$, $\mathbf{K}_{k\|k-1}$	Scalar innovation and gain matrix random matrix
$\mathbf{Y}_{k\|k-1}$	Predicted covariance measurement
z	Scaling factor in random matrix
$\hat{\mathbf{N}}_{k\|k-1}$, $\hat{\mathbf{Z}}_{k\|k-1}$	Generated matrices in random matrix
$\mathbf{V}_k$	Variance for extent estimate $\mathbf{M}_k$
$\mathbf{R}_\phi(\phi_k)$	Rotation matrix in the counterclockwise direction for heading ϕ_k
λ	Poisson parameter for number of clutter measurements
λ_{sim}	Poisson parameter used in simulation of clutter measurements
Λ_k, V_Λ	Square region and its volume for simulation of clutter measurements
Γ_k	Validation region
V_k	Volume of validation region
$\tilde{\mathbf{z}}^{i,j}_k$	Concatenated measurement vector for $\mathbf{z}^{i,j}_k \in E^j_i$
$\tilde{\mathbf{y}}^j_k$	Predicted measurement vector for $\mathbf{z}^{i,j}_k \in E^j_i$
$\tilde{\mathbf{R}}^j_k$	Noise covariance matrix for $\mathbf{z}^{i,j}_k \in E^j_i$
$\mathbf{H}^j_k$	Jacobian matrix for $\mathbf{z}^{i,j}_k \in E^j_i$
$\mathbf{v}^{i,j}_k$	Innovation vector for $\mathbf{z}^{i,j}_k \in E^j_i$
$\mathbf{S}^j_{k\|k-1}$, $\mathbf{W}^j_{k\|k-1}$	Innovation covariance and gain matrix for $\mathbf{z}^{i,j}_k \in E^j_i$
$\mathbf{m}^{i,j}_k$	Filtered mean for $\mathbf{z}^{i,j}_k \in E^j_i$
$\mathbf{P}^{i,j}_k$	Filtered covariance for $\mathbf{z}^{i,j}_k \in E^j_i$
$\bar{\mathbf{z}}^{i,j}_k$, $\bar{\mathbf{Z}}^{i,j}_k$	Centroid measurement and scattering matrix for $\mathbf{z}^{i,j}_k \in E^j_i$
$\mathbf{N}^{i,j}_{k\|k-1}$	Innovation matrix in random matrix for $\mathbf{z}^{i,j}_k \in E^j_i$
$\hat{\mathbf{N}}^{i,j}_{k\|k-1}$, $\hat{\mathbf{N}}^{i,j}_{k\|k-1}$	Generated matrices in random matrix for $\mathbf{z}^{i,j}_k \in E^j_i$
$\alpha^{i,j}_k$, $\mathbf{M}^{i,j}_k$	Degrees of freedom and extent matrix for $\mathbf{z}^{i,j}_k \in E^j_i$
$\mathcal{E}, E^j, E^j_i$	Association hypothesis space, subspace and single hypothesis.
$\beta^{i,j}_k$	Association weight for hypothesis E^j_i
β^{max}_k	Maximum association weight
$\bar{x}_k, \bar{y}_k$	Mean values in x- and y-coordinates of $\mathbf{Z}_k$
N	Number of Monte Carlo runs
n^{min}, n^{max}	Upper and lower bound for n^t_k
g	Subtraction parameter for finding n^{min}
$n^{min}_{sim}, n^{max}_{sim}$	Simulation values in uniform distribution for n^t_k
$\mathbf{C}$	Constraint matrix in ellipse fitting
$\mathbf{D}$	Design matrix in ellipse fitting
$\mathbf{S}$	Scatter matrix in ellipse fitting
$F(\cdot,\cdot)$	Algebraic distance in ellipse fitting
$\mathbf{a}, \mathbf{c}$	Ellipse fitting coefficients and data vector

σ_a^{tun}, $\Sigma_{a,b}^{tun}$	Tuning parameters for kinematics and extent covariances
$\mathbf{Q}^{tun}$, $\mathbf{R}^{tun}$	Tuning parameters for process- and measurement noise covariance
α	Significance level
K	Number of cluster centroids
$\mathbf{C}_k$	Set of cluster centroid positions
$\lvert \cdot \rvert$	Cardinality of set
$p(\cdot)$	Probability density function, continuous variables
$P(\cdot)$	Probability mass function, discrete variables
$\mathcal{N}(\cdot)$	Normal pdf
$U\{\cdot,\cdot\}$	Discrete uniform distribution
χ_d^2	Chi-squared distribution with d degrees of freedom
$\mathcal{W}_d(\cdot)$	Wishart pdf of matrix with dimension d
$\mathcal{IW}_d(\cdot)$	Inverse Wishart of matrix with dimension d
$C_{(\cdot)}$	Constant
$(\cdot)_{k\lvert k-1}$	Predicted quantity
$\text{arctan2}(\cdot,\cdot)$	Arctan-function based on the argument signs
$\text{diag}[\cdot]$	Diagonal matrix with arguments along the main diagonal
$\max_{(\cdot)}$, $\min_{(\cdot)}$	Maximum and minimum with respect to the argument
$\text{tr}(\cdot)$	Trace

Chapter 1

Introduction

This chapter provides background and motivation for the thesis, a presentation of sensors, description of target tracking, a problem description, and an outline of the different chapters in the thesis.

1.1 Background and motivation

Automation of manual labor has entered several industries over the last couple of years, for instance with robots in production processes, trading stocks, filling out forms and so on. These robots are made to do specific tasks in a restricted environment and are mostly rule-based. In recent years new types of autonomous systems have emerged, that require no human interaction and are self-governing. Examples of these systems are the autonomous cars of Google, Tesla and Uber.

The work with developing autonomous ships has also come a long way, and Norwegian companies like DNV GL, Kongsberg Maritime and Maritime Robotics are leading players in this field. An example is the container ship Yara Birkeland (see Kongsberg [2017]) that will have control systems delivered by Kongsberg Maritime and is planned to be fully autonomous in 2020. The purpose of the ship is to move containers from the Yara facility at Herøya to the ports of Brevik and Larvik. Other vessels that have been developed is the DNV GL Revolt model ship and the Telemetron by Maritime Robotics. Both these ships have been used in testing of tracking and collision avoidance systems for autonomous ships, for instance in Wilthil et al. [2017].

This thesis is written as a contribution to the Autosea and Autoferry projects at NTNU. These projects focuses on developing technology for autonomous ships. The Autosea project belongs to the department of engineering cybernetics and is about sensor fusion and collision avoidance for autonomous surface vehicles (ASVs). The Autoferry project is a collaboration between the departments for engineering cybernetics, electronic systems and marine technology to create methods that will enable the development of autonomous passenger ferries. The project has so far come up with the concept ferry depicted in Figure 1.1, which is a 1:2 scaled model of the autonomous passenger ferry that will be built. Experiments with collision avoidance with LIDAR and other sensors will probably happen in the spring 2019.

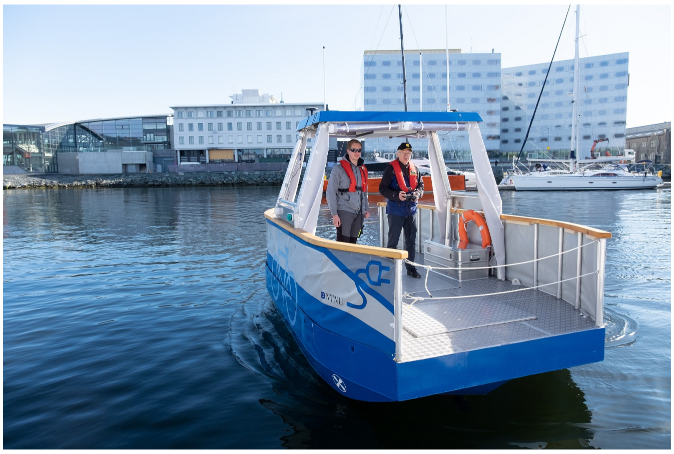


Figure 1.1: Autonomous pilot ferry Milliampere in 1:2 scaled model of the autonomous ferry that will be built in the Autoferry project. Courtesy of Kai Dragland.

The autonomous ferry is supposed to travel in the Trondheim canal between Ravnkloa and Vestre Kanalkai shown in Figure 1.2. The idea behind this ferry is to transport pedestrians and cyclists across the canal, which is about 95 metres wide, instead of building a footbridge over it. The ferry will be operational during most of the daytime and carry up to 12 passengers.

To be able to operate autonomously between the two piers in Figure 1.2 the ferry needs to have a good understanding of what is happening around it in the canal. The information it gathers of the surroundings comes through sensors that are mounted on the ferry to give it a visual perception. It is of importance to have accurate sensors and software that interpret the sensor data correctly such that the steering algorithms can operate with estimates of the surroundings close to the true situation.

1.2 Sensors

On the ferry there will be four different sensors: camera, radar, LIDAR and infrared (IR). All these will give different types of detections and capture the surrounding features of the canal in four different ways. Hence it is essential to both have good algorithms for each sensor, but also have robust sensor fusion algorithms that gather the results from all sensors into a consistent world picture of the reality around the ferry. Even though LIDAR data will be the main focus of the current work, all sensor data types will be presented for

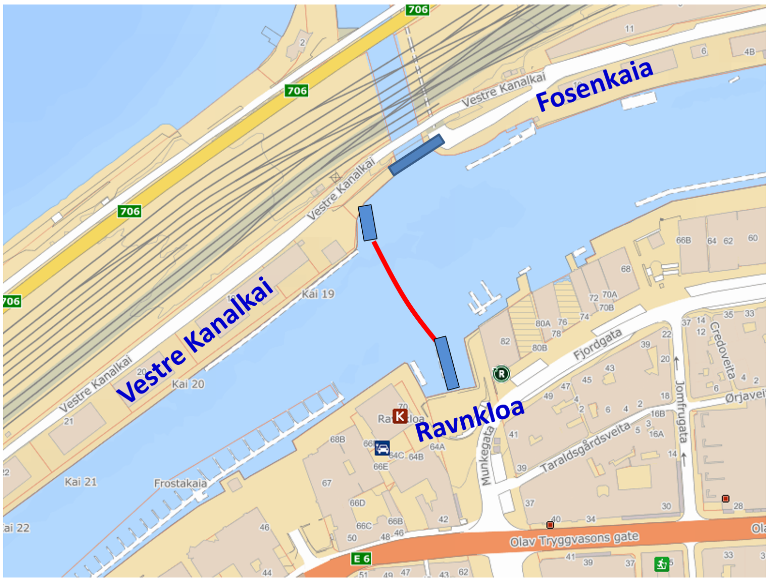


Figure 1.2: Planned route for the autonomous ferry. Courtesy of Egil Eide.

completeness. There are two types of sensors: active and passive. Active sensors works by transmitting a signal $s(t)$ and receiving a reflected signal $r(t)$ from a target, as shown in Figure 1.3. Radar and LIDAR are active sensors that use different electromagnetic signals to get information about the surroundings. Camera is a passive sensor that only receives signals in the form of visible light. IR can be both an active and a passive sensor.

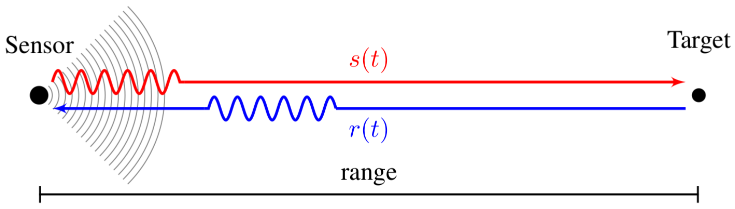


Figure 1.3: Signal transmission between a sensor and a point target. The transmitted and reflected signal is denoted $s(t)$ and $r(t)$ respectively. Courtesy of Rødningsby [2010].

The camera sensor takes several pictures per second and give a lot of image data to process. For a camera to capture objects in an image properly it needs at least some light that comes through the lens. An image is represented by a 2D array consisting of pixels that can take RGB vector values in the spectrum from 0 to 255. For instance, the vectors $[0, 0, 0]$ and $[255, 255, 255]$ represents a black and white pixel respectively. The scientific field of extracting information from image data is called computer vision, and there are numerous methods to find interest points and feature descriptors that can be used with machine learning algorithms to detect objects. Methods like neural networks are typically used for object detection in images because of the ability to recognize patterns in the nonlinear image data.

Radar uses radio-waves to detect objects and it determines the range, angle or velocity of the objects. It works by sending out electromagnetic pulses from a transmitting antenna, and then receive reflected pulses through a receiving antenna (usually the same as the transmitting antenna). The received signals are then processed to extract information about the detected objects. When the transmitted signals hits an object, most of them are reflected or scattered in different directions. However, some of them penetrate into the object. The radar detection image of an object will thus be scattered over the object rather than just detections along the contour of the object. The radar signals have a low frequency which gives longer wavelengths and thus higher uncertainties in the received signals. Materials of considerable electric conductivity reflects radar signals well, which makes it well suited on aircraft and ship detection for instance. If the detected object is moving either towards or away from the transitter, the reflected waves will have a slight equivalent change in frequency caused by the Doppler effect.

LIDAR stands for light detection and ranging, and is a laser-range sensor that transmits laser beams and recieves reflected signals. The beams sweep over the surveillance area with a small angular distance between them, and they are reflected by the first surface they hit. This gives a different structure to the data compared to a radar, because the LIDAR measurements are distributed along the object surface and not penetrating into the object. It is used in mapping of terrain because of its 3D reconstruction of the surroundings.

The infrared (IR) sensor transmits electromagnetic rays in the infrared spectrum and receives the reflected rays, but it can also just detect IR waves without transmitting signals (Chilton [2014]). The result is an image with color codes where objects that emit heat waves will be highlighted. IR sensors also work when it is dark, and computer vision algorithms are used to detect objects in the images that are obtained from the sensor.

1.3 Target tracking

Target tracking refers to the situation where one or several sensors, for instance radars, LIDARs, cameras or infrared sensors are used to determine the kinematical properties like position, velocity and/or acceleration of one or multiple remote targets. Tracking one target is referred to as single-target tracking, and multi-target tracking when there are more than one target. In this thesis we will focus on single-target tracking, and not consider multiple targets in the surveillance area.

Single target tracking is done over some time span with one or multiple measurements for each time step. The standard assumption is that the target generates at most one measurement per time step, and we refer to this as point object tracking. When the target generates multiple measurements per time step it is called extended object tracking (EOT).

The difference is therefore caused by both the sensor resolution and the spatial extent of the target. If the sensor resolution is poor, a target with extent may only generate one measurement per time step, while a sensor with better resolution would detect dozens of points.

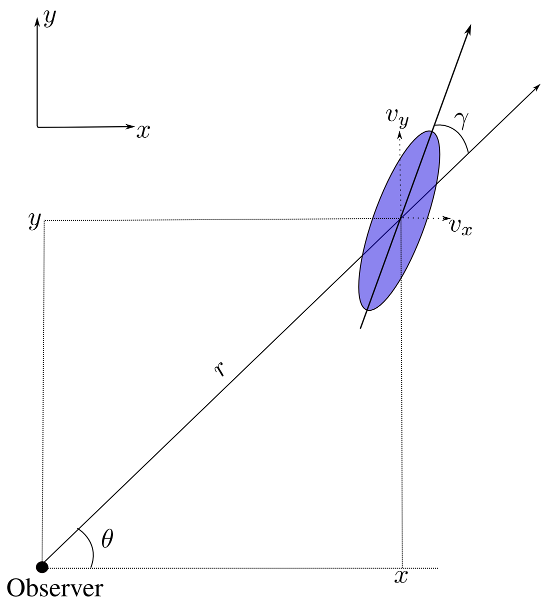
Traditionally, tracking methods have been developed with the point target assumption, but over the years as the sensor resolution has gotten better, this assumption has become more and more invalid. From these advances in sensor technology the field of EOT has emerged, and an extensive amount of work has been done to develop tracking methods for targets with extent. EOT was not treated as a separate discipline until 2008, and most existing tracking systems do not use this technology. The target extent can be modelled as different geometrical shapes like a rectangle, ellipse or a line, dependent on which approximation that fits the true target shape best. There have been numerous different approaches when it comes to modeling the EOT problem mathematically, and some of them will be presented in the next chapter.

When it comes to tracking extended objects at sea the targets are typically ships, boats, pleasure crafts, kayaks or other vessels. For most of these targets, a good approximation of the extent will be an ellipse as shown in Figure 1.4b with major axis a and minor axis b. The extended target tracking scenario is illustrated in Figure 1.4a where an observer typically positioned in the origin of a Cartesian coordinate system observes the elliptical target. The center of the target ellipse (x, y) is located a distance r from the observer, which is called the range. The angular position of the target is given by the bearing θ, and together with the range it gives the target position in polar coordinates (r, θ). In addition to position away from the observer, the target has velocities v_x and v_y, which determine its heading. Figure 1.4a, shows the course γ. If the target has side-slip the course will be different from the heading, otherwise they will be the same. The course thus determines where the target is going, and is a function of the position (x, y) and velocities (v_x, v_y).

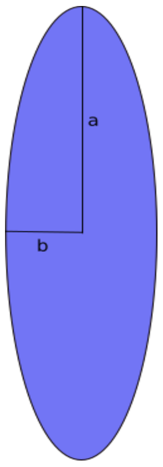
1.4 Problem description

In the setting of the autonomous ferry that is supposed to travel over the canal in Figure 1.2, it is important to have good estimates of the boat traffic around the planned ferry route. The aim is to accurately estimate the vessels' position, velocity and extent by elliptical approximations, such that the ferry can maneuver according to these estimates. In the canal there is one vessel that travels regularly in and out from the pier at Ravnkloa, and that is the MS Nidarholm boat travelling to the small island of Munkholmen. It is also referred to as the Munkholm boat, and it departs every hour during summer season. Hence it is of importance that the ferry is able to track this vessel with great precision every time it arrives and departs at Ravnkloa.

The problem of tracking the Munkholm boat can be formulated as a single target EOT problem as described in the previous section. To give the ferry a best possible understanding of where this boat is going, it needs a sensor with high resolution. This is provided by the LIDAR, and the one that we use has a range of 100 metres which means that it will cover the whole ferry route shown in Figure 1.2. Hence the goal of this thesis is to find



(a) Observer setting



(b) Target ellipse

Figure 1.4: Illustration of the EOT setting for a single target approximated by an ellipse.

robust methods that can track a single extended target, more specifically the Munkholm boat, by using data from a LIDAR sensor that is mounted to the pier at Ravnkloa. When using a sensor like the LIDAR, it will detect many points that is not from the target we want to track. These measurements are called clutter, and the tracking methods need to filter them out such that the target generated measurements are used in the estimation.

1.5 Outline

The thesis is organized in the following way:

- Chapter 2 is a literature survey of relevant papers and books on target tracking and EOT.
- Chapter 3 presents the theory behind EOT with the extended Kalman filter (EKF), contour EKF and random matrix filter. It is assumed that all measurements come from the target.
- Chapter 4 presents the theory behind the general probabilistic data association (GPDA) filter when assuming that measurements can be both from target and clutter.
- Chapter 5 presents the simulation experiments and gives simulation results for the GPDA filter with contour EKF and random matrix when generating LIDAR measurements both with and without clutter.

- Chapter 6 gives results from real LIDAR data of the Munkholm boat which is tracked by using the GPDA filter with contour EKF and random matrix.
- Chapter 7 discusses the results from simulations and real LIDAR data, and presents alternative implementation choices.
- Chapter 8 gives concluding remarks and discusses further research topics.

Chapter 2

Literature Review

In this chapter the relevant papers and books written on point target tracking and EOT are summarized and discussed. First the traditional point target approaches for both single- and multi-target will be presented, followed by extended single target and multi-target contributions.

Point target tracking have been around for some time and after Kalman [1960] developed the Kalman filter, there were several contributions concerning both single- and multitarget tracking in a cluttered environment, for instance Reid [1979]. A lot of these papers was summarized in the book by Bar-Shalom and Li [1995], where the probabilistic data association (PDA) is a key method when tracking with clutter measurements. Later, in Bar-Shalom et al. [2001] further advances in point target tracking was summarized, and the field was then a well established research area.

The first contribution in EOT can be traced back to Drummond et al. [1988], but there were few contributions that came after it. In the 2000s the sensor technology had developed significantly, and the traditional point target assumption seemed more inappropriate. One of the methods developed were based on the extended Kalman filter (EKF), for instance in Salmond and Ristic [2004], which introduced the minor and major axes of the target ellipse as a part of the state vector. Another approach that modelled the ellipse by a random matrix was first introduced by Koch [2008], and it was developed to track a single target or a target group from radar data. The random matrix approach was developed further in Feldmann et al. [2011], where the sensor noise was included in the model which was not the case in the original paper by Koch [2008]. However, this made the mathematical derivation of the filter less correct, and more based on assumptions than analytical probabilistic results. This problem was handled in the contribution by Lan and Li [2012] and later in Granström and Orguner [2014]. The random matrix filter was now generalized to handle rotational motion of the target ellipse, but most of the work were done using radar data.

In Schuster and Reuter [2015] the random matrix filter were used on LIDAR data from a boat with a GPS tracker, and compared it to a similar experiment with the radar sensor. The general probabilistic data association (GPDA) filter was used to handle clutter measurements in the sensor data. The GPDA was introduced in Schubert et al. [2012] as a method to deal with clutter measurements in an extended target situation, and thus generalizing the original PDA filter from Bar-Shalom and Li [1995]. The GPDA was built on the same idea as the multiple detection PDA first presented by Habtemariam et al. [2011], which allowed multiple detections to originate from target.

In Mahler [2003] a new modelling of the measurements were introduced using the rigorous formulation of finite set statistics (FISST). This set theoretic approach solves the multi-target tracking problem by using random finite sets (RFS) to model the targets and measurements. The first-order moment of an RFS is called probability hypothesis density (PHD), and it is an intensity function defined over the target states. The filter that arise from the PHD has been thoroughly studied, for instance in Granström et al. [2011a] where a Gaussian mixture approach is used to limit the number of set partitions. This approach resulted in further work by Granström et al. [2011b] who presented the contour measurement modelling of LIDAR measurements on both rectangular and elliptical targets in a multi-target tracking scenario with the PHD filter. In Granström et al. [2014] this method was further investigated and tested on LIDAR data from cars under an assumed rectangular shape. One of the advantages with the Gaussian mixture modelling of the LIDAR measurements is that it enables the use of EKF, which is a well-known tracking method. More recent RFS approaches using the Poisson multi-Bernoulli mixture (PMBM) filter are given in Granström et al. [2017a] and Granström et al. [2017b]. Together with the most recent approach in Granström et al. [2018], these contributions represent the state-of-the-art methods in multi-target extended object tracking.

In this thesis we will study the contour EKF for elliptical targets developed by Granström et al. [2011b], and use it in a single target case instead of the multi-target case it originally was developed for. To handle the clutter measurements, we could have used the PHD filter from the original approach, but we choose to apply the GPDA filter akin to Schubert et al. [2012] instead. When doing this, we avoid the rigorous RFS modelling, and are able to create a simple tracking algorithm for LIDAR data. This has not been done before as far as the authors know, and is a new contribution to the EOT field. To compare the method with an existing GPDA method, the random matrix filter from Schuster and Reuter [2015] is reconstructed.

Chapter 3

Extended object tracking

In this chapter we present theory for dealing with EOT, where multiple measurements originates from target. First, the extended Kalman filter (EKF) theory is presented together with the contour EKF (CEKF) for tracking an elliptically shaped object. The random matrix approach is then presented as an alternative way of dealing with the EOT problem. Both the CEKF and random matrix are presented with the underlying assumption that all the measurements comes from the target.

3.1 Extended Kalman filter

In the target tracking problem which includes EOT, the target typically moves in some time span from $k = 1$ to $k = T$. During this movement its position, velocity and extent can be described by the state vector $\mathbf{x}_k$ of size n_x for each time step k. The goal is to estimate the hidden state $\mathbf{x}_k$ in time step k from the measurements $\mathbf{Z}_{1:k} = \{\mathbf{z}_1, ..., \mathbf{z}_k\}$, where each $\mathbf{z}_k$ has size n_z. This is supposed to happen in real time at time step k and thus an efficient algorithm needs to be constructed. The marginal posterior, also referred to as the filtering density, is given by

$$p(\mathbf{x}_k|\mathbf{Z}_{1:k}) = \frac{p(\mathbf{Z}_{1:k}|\mathbf{x}_k)p(\mathbf{x}_k)}{p(\mathbf{Z}_{1:k})}, \tag{3.1}$$

where $p(\mathbf{x}_k)$ is the prior density defined by the dynamic model and $p(\mathbf{Z}_{1:k}|\mathbf{x}_k)$ is the likelihood model for the measurements. To compute this posterior as k increases will be a computationally complex problem, and not an efficient way of solving the filtering problem. Hence we make a first order Markov assumption on the dependence between the states and the measurements. We assume that the current state $\mathbf{x}_k$ only depends on the previous state $\mathbf{x}_{k-1}$, and that each measurement vector $\mathbf{z}_k$ only depends on the state vector $\mathbf{x}_k$ as shown in Figure 3.1.

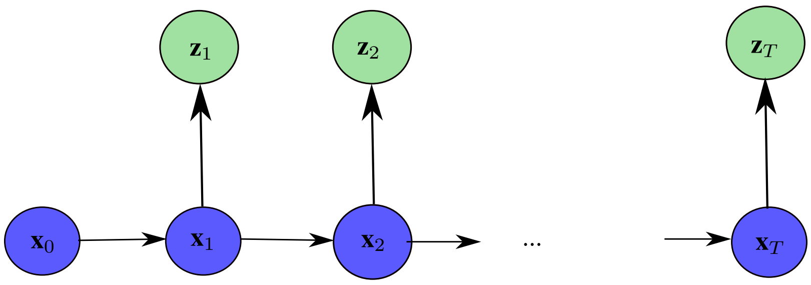


Figure 3.1: Conditional dependence structure for estimating states from observations.

Now the filtering density from (3.1) can be written as

$$p(\mathbf{x}_k|\mathbf{Z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{Z}_{1:k-1})}, \tag{3.2}$$

The prior density $p(\mathbf{x}_k|\mathbf{Z}_{1:k-1})$, also referred to as the predicted density, is given by the Chapman-Kolmogorov equation

$$p(\mathbf{x}_k|\mathbf{Z}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{Z}_{1:k-1})d\mathbf{x}_{k-1}, \tag{3.3}$$

where $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is the transition density or dynamic model, and $p(\mathbf{x}_{k-1}|\mathbf{Z}_{1:k-1})$ is the filtering density for time step $k-1$. The extended Kalman filter (EKF) is based on the assumption that the dynamic model is given by

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}) + \mathbf{q}_{k-1}, \quad \text{where} \quad p(\mathbf{q}_{k-1}) = \mathcal{N}(\mathbf{q}_{k-1}; \mathbf{0}, \mathbf{Q}_{k-1}), \tag{3.4}$$

which means that the transition model $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is Gaussian with mean vector $\mathbf{f}(\mathbf{x}_{k-1})$ and covariance matrix $\mathbf{Q}_{k-1}$. Here the three slot notation $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ means that the vector $\mathbf{x}$ has a multivariate normal density with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. The function $\mathbf{f}(\mathbf{x}_{k-1})$ is the dynamic model function that describes how the state vector evolves, and we assume it is a linear function represented by the matrix $\mathbf{F}_{k-1}$. The vector $\mathbf{q}_{k-1} \in \mathbb{R}^{n_x}$ is called the process noise and $\mathbf{Q}_{k-1} \in \mathbb{R}^{n_x \times n_x}$ is the corresponding noise covariance matrix. The previous filtering density at time step $k-1$ is given by

$$p(\mathbf{x}_{k-1}|\mathbf{Z}_{1:k-1}) = \mathcal{N}(\mathbf{x}_{k-1}; \mathbf{m}_{k-1}, \mathbf{P}_{k-1}), \tag{3.5}$$

and when using the Chapman-Kolmogorov equation (3.3) on these two Gaussian densities we get that the predicted density is

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{Z}_{1:k-1}) &= \int \mathcal{N}(\mathbf{x}_k; \mathbf{F}_{k-1}\mathbf{x}_{k-1}, \mathbf{Q}_{k-1})\mathcal{N}(\mathbf{x}_{k-1}; \mathbf{m}_{k-1}, \mathbf{P}_{k-1})d\mathbf{x}_{k-1} \\ &= \mathcal{N}(\mathbf{x}_k; \mathbf{m}_{k|k-1}, \mathbf{P}_{k|k-1}). \end{aligned} \tag{3.6}$$

Here the predicted mean vector and covariance matrix are given by

$$\mathbf{m}_{k|k-1} = \mathbf{F}_{k-1}\mathbf{m}_{k-1} \tag{3.7}$$
$$\mathbf{P}_{k|k-1} = \mathbf{F}_{k-1}\mathbf{P}_{k-1}\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1}. \tag{3.8}$$

The full derivations of these results can be found in Särkkä [2013].

To calculate the filtering density in (3.2) we need the measurement model $p(\mathbf{z}_k|\mathbf{x}_k)$, also referred to as the likelihood density. In the EKF it is given by

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{r}_k, \quad \text{where} \quad p(\mathbf{r}_k) = \mathcal{N}(\mathbf{r}_k; \mathbf{0}, \mathbf{R}_k), \tag{3.9}$$

which means that $p(\mathbf{z}_k|\mathbf{x}_k) = \mathcal{N}(\mathbf{z}_k; \mathbf{h}(\mathbf{x}_k), \mathbf{R}_k)$, where $\mathbf{h}(\mathbf{x}_k)$ is the measurement model function and $\mathbf{r}_k \in \mathbb{R}^{n_z}$ and $\mathbf{R}_k \in \mathbb{R}^{n_z \times n_z}$ are the measurement noise vector and covariance matrix respectively. Both the functions $\mathbf{f}(\mathbf{x}_k)$ and $\mathbf{h}(\mathbf{x}_k)$ needs to be differentiable, and we have already assumed that $\mathbf{f}(\mathbf{x}_k) = \mathbf{F}_{k-1}$ is linear which makes it differentiable. We do not assume that $\mathbf{h}(\mathbf{x}_k)$ is linear, and we need to linearize it so that we can write the filtering density as the approximation

$$p(\mathbf{x}_k|\mathbf{Z}_{1:k}) \simeq \mathcal{N}(\mathbf{x}_k; \mathbf{m}_k, \mathbf{P}_k), \tag{3.10}$$

where $\mathbf{m}_k$ and $\mathbf{P}_k$ are the filtered mean vector and covariance matrix respectively. These approximations are obtained through Taylor series expansion of the mean and covariance of the non-linear function $\mathbf{h}(\mathbf{x}_k)$. We will drop the time indexing k in this derivation.

A first order approximation of the measurement function $\mathbf{h}(\cdot)$ can be written like

$$\mathbf{h}(\mathbf{x}) = \mathbf{h}(\mathbf{m} + \delta\mathbf{x}) \approx \mathbf{h}(\mathbf{m}) + \mathbf{H}_{\mathbf{x}}(\mathbf{m})\delta\mathbf{x}, \tag{3.11}$$

where $\mathbf{x} = \mathbf{m} + \delta\mathbf{x}$ and $\delta\mathbf{x} \sim N(\mathbf{0}, \mathbf{P})$. The Jacobian matrix $\mathbf{H}_{\mathbf{x}}(\mathbf{m}) = \left[\frac{\partial h_j(\mathbf{x})}{\partial \mathbf{x}_i}\right]_{\mathbf{x}=\mathbf{m}}$ for $i = 1, .., n_x$ and $j = 1, ..., n_z$, is given by the partial derivatives of the function $\mathbf{h}(\cdot)$, and hence it needs to be differentiable. If we do not have a closed form expression of $\mathbf{h}()$, we can calculate the Jacobian numerically. This is the case in the Gaussian mixture EKF presented in the next section. The expectation of the measurement model function with respect to $\mathbf{x}$ can be approximated as

$$\mathrm{E}_{\mathbf{x}}[\mathbf{h}(\mathbf{x})] \simeq \mathbf{h}(\mathbf{m}) + \mathbf{H}_{\mathbf{x}}(\mathbf{m})\,\mathrm{E}_{\mathbf{x}}[\delta\mathbf{x}] = \mathbf{h}(\mathbf{m}) \tag{3.12}$$

The covariance matrix approximation of $\mathbf{h}(\mathbf{x})$ then becomes

$$\begin{aligned}
\mathrm{Cov}_{\mathbf{x}}[\mathbf{h}(\mathbf{x})] &= \mathrm{E}_{\mathbf{x}}[(\mathbf{h}(\mathbf{x}) - \mathrm{E}_{\mathbf{x}}[\mathbf{h}(\mathbf{x})])(\mathbf{h}(\mathbf{x}) - \mathrm{E}_{\mathbf{x}}[\mathbf{h}(\mathbf{x})])^T] \\
&\simeq \mathrm{E}_{\mathbf{x}}[(\mathbf{h}(\mathbf{x}) - \mathbf{h}(\mathbf{m}))(\mathbf{h}(\mathbf{y}) - \mathbf{h}(\mathbf{m}))^T] \\
&\simeq \mathrm{E}_{\mathbf{x}}[(\mathbf{H}_{\mathbf{x}}(\mathbf{m})\delta\mathbf{x})(\mathbf{H}_{\mathbf{x}}(\mathbf{m})\delta\mathbf{x})^T] \\
&= \mathbf{H}_{\mathbf{x}}(\mathbf{m})\,\mathrm{E}_{\mathbf{x}}[\delta\mathbf{x}\delta\mathbf{x}^T]\mathbf{H}_{\mathbf{x}}^T(\mathbf{m}) = \mathbf{H}_{\mathbf{x}}(\mathbf{m})\mathbf{P}\mathbf{H}_{\mathbf{x}}^T(\mathbf{m}).
\end{aligned} \tag{3.13}$$

Here, the equations (3.11) and (3.12) are used to find the approximation. In terms of the measurement model, the approximation of $\mathbf{h}(\cdot)$ can be used to obtain the EKF prediction and filtering step. Since the dynamic model is assumed linear, the prediction step becomes the same as in the linear Kalman filter given by (3.7) and (3.8). To obtain the filtering equations, the joint distribution for $\mathbf{x}_k$ and $\mathbf{z}_k$ is derived by using the filtering density

(3.10), the likelihood density $p(\mathbf{z}_k|\mathbf{x}_k)$ given by (3.9), and the lemma presented in Särkkä [2013] about joint normal densities. From these results the joint density becomes

$$\begin{aligned} p(\mathbf{x}_k, \mathbf{z}_k|\mathbf{Z}_{1:k-1}) &= p(\mathbf{x}_k|\mathbf{Z}_{1:k-1})p(\mathbf{z}_k|\mathbf{x}_k) \\ &= \mathcal{N}(\mathbf{x}_k; \mathbf{m}_{k|k-1}, \mathbf{P}_{k|k-1})\mathcal{N}(\mathbf{z}_k; \mathbf{h}(\mathbf{m}_{k|k-1}), \mathbf{R}_k) \\ &= \mathcal{N}\left(\begin{bmatrix} \mathbf{x}_k \\ \mathbf{z}_k \end{bmatrix}; \begin{bmatrix} \mathbf{m}_{k|k-1} \\ \mathbf{h}(\mathbf{m}_{k|k-1}) \end{bmatrix}, \begin{bmatrix} \mathbf{P}_{k|k-1} & \mathbf{P}_{k|k-1}\mathbf{H}_\mathbf{x}^T \\ \mathbf{H}_\mathbf{x}\mathbf{P}_{k|k-1} & \mathbf{H}_\mathbf{x}\mathbf{P}_{k|k-1}\mathbf{H}_\mathbf{x}^T + \mathbf{R}_k \end{bmatrix}\right), \end{aligned} \tag{3.14}$$

where the Jacobian is denoted $\mathbf{H}_\mathbf{x} = \mathbf{H}_\mathbf{x}(\mathbf{m}_{k|k-1})$. From this result the conditional density $p(\mathbf{x}_k|\mathbf{Z}_{1:k}) \simeq \mathcal{N}(\mathbf{x}_k; \mathbf{m}_k, \mathbf{P}_k)$ is found by using known formulas for conditional Gaussians on the joint density (3.14). The result yields

$$\mathbf{S}_{k|k-1} = \mathbf{H}_\mathbf{x}\mathbf{P}_{k|k-1}\mathbf{H}_\mathbf{x}^T + \mathbf{R}_k \tag{3.15}$$

$$\mathbf{W}_{k|k-1} = \mathbf{P}_{k|k-1}\mathbf{H}_\mathbf{x}^T\mathbf{S}_{k|k-1}^{-1} \tag{3.16}$$

$$\mathbf{m}_k = \mathbf{m}_{k|k-1} + \mathbf{W}_{k|k-1}[\mathbf{z}_k - \mathbf{h}(\mathbf{m}_{k|k-1})] \tag{3.17}$$

$$\mathbf{P}_k = \mathbf{P}_{k|k-1} - \mathbf{W}_{k|k-1}\mathbf{S}_{k|k-1}^{-1}\mathbf{W}_{k|k-1}^T \tag{3.18}$$

as the filtering equations for EKF. The matrices $\mathbf{W}_{k|k-1}$ and $\mathbf{S}_{k|k-1}$ are called the gain and innovation covariance matrices respectively. The filtering equations (3.17) and (3.18) are a local linearization of the non-linear measurement model $\mathbf{h}(\mathbf{x}_k)$, and are used in the implementations for EOT later in this thesis.

Example 3.1:
Let us consider a case where we track a target ellipse and the state vector is

$$\mathbf{x}_k = [x_k, y_k, v_{x,k}, v_{y,k}, a_k, b_k]^T.$$

The dynamic model matrix $\mathbf{F}_{k-1}$ is given by

$$\mathbf{F}_{k-1} = \begin{bmatrix} 1 & 0 & \Delta t_k & 0 & 0 & 0 \\ 0 & 1 & 0 & \Delta t_k & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.19}$$

The positional measurements are given in polar coordinates, and the minor and major axes of the target ellipse is also observed. Then the measurement model function is $\mathbf{h}(\mathbf{y}_k) = [r(\mathbf{x}_k), \theta(\mathbf{x}_k), a_k, b_k]^T$, where the range and bearing functions are given by

$$r(\mathbf{x}_k) = \sqrt{x_k^2 + y_k^2} \tag{3.20}$$

$$\theta(\mathbf{x}_k) = \arctan(y_k/x_k). \tag{3.21}$$

Hence the Jacobian matrix expressed by the predicted mean vector

$$\mathbf{m}_{k|k-1} = [x_{k|k-1}, y_{k|k-1}, v_{x,k|k-1}, v_{y,k|k-1}, a_{k|k-1}, b_{k|k-1}]^T \tag{3.22}$$

is given by

$$\mathbf{H}_\mathbf{x}(\mathbf{m}_{k-1}) = \begin{bmatrix} \frac{x_{k|k-1}}{\sqrt{(x_{k|k-1})^2+(y_{k|k-1})^2}} & \frac{y_{k|k-1}}{\sqrt{(x_{k|k-1})^2+(y_{k|k-1})^2}} & 0 & 0 & 0 & 0 \\ -\frac{y_{k|k-1}}{(x_{k|k-1})^2+(y_{k|k-1})^2} & \frac{y_{k|k-1}}{(x_{k|k-1})^2+(y_{k|k-1})^2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (3.23)$$

This can be used in the filtering equations (3.17) and (3.18) to obtain estimates of the state vector. The parameter values are

$$T = 100, \quad \mathbf{Q}_{k-1} = \text{diag}[0.01^2, 0.01^2, 0.01^2, 0.01^2, 0.001^2, 0.001^2],$$
$$\mathbf{R}_k = \text{diag}[0.1^2, (\frac{3}{360} \cdot 2\pi)^2, 0.01^2, 0.01^2], \quad \mathbf{x}_0 = [10, 10, 1, 1, 5, 1.5]^T, \quad \Delta t_k = 1,$$

where the notation $\text{diag}[\cdot]$ means a diagonal matrix. We simulate a target process from the dynamic equation (3.4) and the measurements from (3.9). We initialize the filter with the ground truth, i.e. $\mathbf{m}_0 = \mathbf{x}_0$, and estimate the predicted and filtered values with the parameters above. The results are shown in Figure 3.2. We have plotted the position and extent for both the true target process and the filter estimate, together with the positional measurements. Observe that the measurements deviates more the longer away from the origin the ellipse is, and the filter becomes less accurate. This is because of the uncertainty in the θ-measurement gives higher deviance for increasing r. $\square$

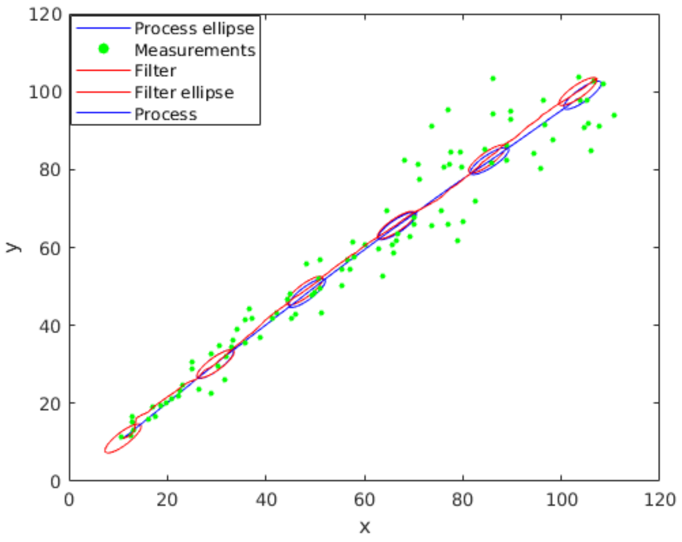


Figure 3.2: Dynamic process, measurements and filtered estimate for the target ellipse in example 3.1.

3.2 Contour EKF

With the EKF equations given in (3.17) and (3.18) we have a filtering method that handles a non-linear measurement function $\mathbf{h}(\cdot)$, and we want to apply this to LIDAR measurements of an ellipse-shaped object. We assume the same dynamics model as in (3.4) for the state vector $\mathbf{x}_k = [x_k, y_k, v_{x,k}, v_{y,k}, a_k, b_k]^T$ with the linear dynamic function $\mathbf{f}(\cdot) = \mathbf{F}_{k-1}$ given in (3.19).

The measurements for each time step k are given by the set $\mathbf{Z}_k = \left\{\mathbf{z}_k^j\right\}_{j=1}^{m_k}$ where the measurement vector consist of two-dimensional positional coordinates given by $\mathbf{z}_k^j = [x_k^j, y_k^j]^T$ for all $j = 1, ..., m_k$. The cumulative measurement set $\mathbf{Z}_{1:k} = \left\{\mathbf{Z}_1, ..., \mathbf{Z}_k\right\}$ is defined to be a set consisting of each measurement set for all time steps up to k. Each of the measurements in $\mathbf{Z}_k$ are assumed to be independent of each other and the likelihood can be expressed as

$$p(\mathbf{Z}_k|\mathbf{x}_k) = \prod_{j=1}^{m_k} p(\mathbf{z}_k^j|\mathbf{x}_k). \tag{3.24}$$

The LIDAR sensor sweeps the surveillance area with time sampling interval Δt_k and measures the bearing $\theta_k^j = \arctan2(y_k^j, x_k^j)$ and range $r_k^j = \sqrt{(x_k^j)^2 + (y_k^j)^2}$ for the closest object reflecting the laser beam. A target generated measurement $\mathbf{z}_k^j$ can be seen as a realization from a random measurement generating point $\mathbf{y}_k^j$ that is measured with some noise $\mathbf{r}_k^j$. The measurement generating points are given as nonlinear functions $\mathbf{y}_k^j(\cdot) : \mathbb{R}^{n_x} \to \mathbb{R}^{2m_k}$ of the target state vector. It is assumed that each measurement is generated by exactly one measurement generating point as shown in Figure 3.3.

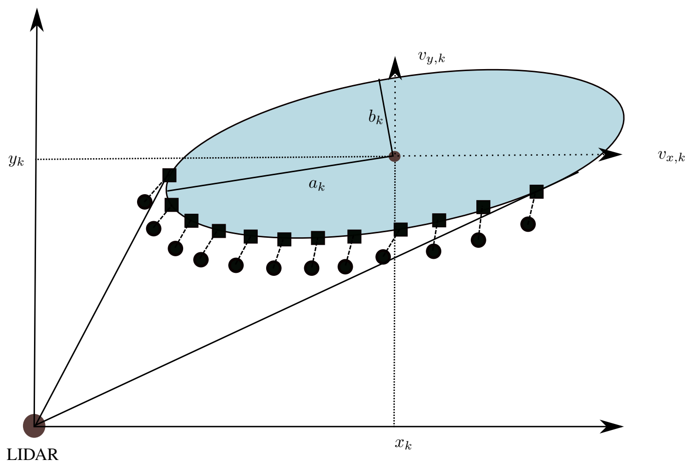


Figure 3.3: Illustration of EOT with LIDAR. State variables, measurement generating points $\mathbf{y}_k^j$ (squares) along the target ellipse connected with associated measurements $\mathbf{z}_k^j$ (circles) are shown.

Now the measurement likelihood $p(\mathbf{z}_k^j|\mathbf{x}_k)$ can be written as the convolution

$$p(\mathbf{z}_k^j|\mathbf{x}_k) = \int p(\mathbf{z}_k^j|\mathbf{y}_k^j(\mathbf{x}_k))p(\mathbf{y}_k^j(\mathbf{x}_k)|\mathbf{x}_k)d\mathbf{y}_k^j(\mathbf{x}_k). \tag{3.25}$$

To find an analytical expression for this likelihood that corresponds with the distribution of real world LIDAR data is challenging, and thus we approximate it by a Gaussian mixture given by

$$p(\mathbf{z}_k^j|\mathbf{x}_k) \approx \sum_{l=1}^{N_L} w^l \mathcal{N}\big(\mathbf{z}_k^j; \mathbf{y}_k^l(\mathbf{x}_k), \mathbf{R}_k^l\big), \quad \sum_{l=1}^{N_L} w^j = 1. \tag{3.26}$$

Here, w^l are the weights for each Gaussian density and $\mathbf{R}_k^l$ is the corresponding measurement noise covariance matrix. Each measurement $\mathbf{z}_k^j$ can be associated with N_L different predicted measurements $\mathbf{y}_k^l$. However, we have already assumed that each measurement comes from exactly one predicted measurement, i.e. $N_L = 1$, and the mixture density in (3.26) is simplified to

$$p(\mathbf{z}_k^j|\mathbf{x}_k) \approx \mathcal{N}\big(\mathbf{z}_k^j; \mathbf{y}_k^j(\mathbf{x}_k), \mathbf{R}_k^j\big). \tag{3.27}$$

This likelihood enables the use of well known filtering methods like the EKF, and is the main motivation behind using Gaussian mixture approximation.

The measurement generating points $\mathbf{y}_k^j(\mathbf{x}_k)$ are computed by using properties of the ellipse geometry, and we switch to a more convenient notation without the timestep subscript k and using (x_0, y_0) and ϕ to respectively denote the center and rotation of the ellipse. First, given that the sensor is located in $(0, 0)$ in the Cartesian plane, we can calculate the bearing interval $[\theta_1, \theta_2]$ where the target measurements can occur. This is done by using the reference coordinates (x^e, y^e) for the ellipse coordinate system, which can be expressed by the regular Cartesian coordinates as

$$\begin{bmatrix} x^e \\ y^e \end{bmatrix} = \begin{bmatrix} \cos(\phi) & \sin(\phi) \\ -\sin(\phi) & \cos(\phi) \end{bmatrix} \begin{bmatrix} x - x_0 \\ y - y_0 \end{bmatrix} \tag{3.28}$$

In the target ellipse coordinate system we have the ellipse equation

$$\frac{(x^e)^2}{a^2} + \frac{(y^e)^2}{b^2} = 1. \tag{3.29}$$

In addition, the tangent through the point (x_t^e, y_t^e) is given by the equation

$$\frac{x_t^e}{a^2}x^e + \frac{y_t^e}{b^2}y^e = 1. \tag{3.30}$$

By inserting the coordinate expressions from (3.28) in (3.30) and rewriting the tangent equation (3.30) such that we get an expression of the form $y = Ax + B$, we get that the slope and constant term are

$$A = \frac{\big(-\frac{b^2 x_t^e}{a^2 y_t^e}\cos(\phi) - \sin(\phi)\big)}{\cos(\phi) - \frac{b^2 x_t^e}{a^2 y_t^e}\sin(\phi)}, \tag{3.31}$$

$$B = \frac{\frac{b^2}{y_t^e} - \frac{b^2 x_t^e}{a^2 y_t^e}\big(-x_0\cos(\phi) + y_0\sin(\phi)\big) + x_0\sin(\phi) + y_0\cos(\phi)}{\cos(\phi) - \frac{b^2 x_t^e}{a^2 y_t^e}\sin(\phi)}. \tag{3.32}$$

To find the solution for the two tangent points laying on a line through the origin we use that the constant term $B = 0$ in (3.32) and that the ellipse equation (3.29) must hold for the tangent points (x_t^e, y_t^e). Rewriting these two equations gives

$$y_t^e = \frac{b^2}{C_2} - \frac{b^2 C_1}{a^2 C_2} x_t^e = \pm b\sqrt{1 - (\frac{x_t^e}{a})^2}, \tag{3.33}$$

where $C_1 = -x_0 \cos(\phi) - y_0 \sin(\phi)$ and $C_2 = x_0 \sin(\phi) - y_0 \cos(\phi)$. This second degree polynomial equation has two solutions for x_t^e and we find the corresponding y_t^e-coordinates from the right hand side of (3.33). This could also have been done by expressing x_t^e in terms of y_t^e in (3.33), and solve for y_t^e. This is done in the implementation when C_2 is close to zero. The corresponding angles θ_1 and θ_2 are calculated from the slope A in (3.31) and we get the different cases

$$\theta_1 = \begin{cases} \min\{\text{arctan2}(A_1, 1), \text{arctan2}(A_2, 1)\} & x_{min} > 0 \\ \min\{\text{arctan2}(-A_1, -1), \text{arctan2}(-A_2, -1)\} & x_{max} < 0 \\ \max\{\text{arctan2}(A_1, 1), \text{arctan2}(A_2, 1)\} - \pi & x_{min} < 0, x_{max} > 0, y_{max} > 0 \\ \max\{\text{arctan2}(A_1, 1), \text{arctan2}(A_2, 1)\} & x_{min} < 0, x_{max} > 0, y_{max} < 0 \end{cases}$$

$$\theta_2 = \begin{cases} \max\{\text{arctan2}(A_1, 1), \text{arctan2}(A_2, 1)\} & x_{min} > 0 \\ \max\{\text{arctan2}(-A_1, -1), \text{arctan2}(-A_2, -1)\} & x_{max} < 0 \\ \min\{\text{arctan2}(A_1, 1), \text{arctan2}(A_2, 1)\} & x_{min} < 0, x_{max} > 0, y_{max} > 0 \\ \min\{\text{arctan2}(A_1, 1), \text{arctan2}(A_2, 1)\} + \pi, & x_{min} < 0, x_{max} > 0, y_{max} < 0 \end{cases}$$

for the two desired angles in all possible configurations of the ellipse. Here we have defined (x_{min}, x_{max}) and (y_{min}, y_{max}) as the minimum and maximum values of all the points on the ellipse in each Cartesian coordinate. For an ellipse centered in $(4, 5)$, $a = 4$, $b = 1.5$, and rotated with $\phi = \pi/4$ the angle span is shown in Figure 3.4.

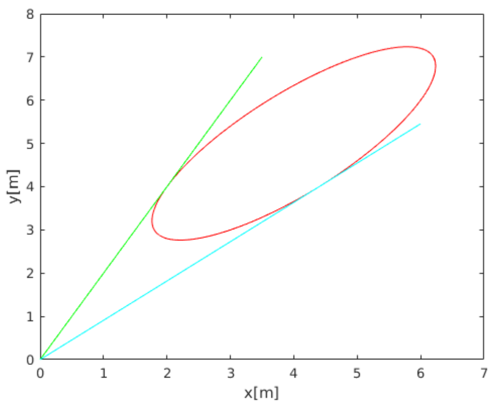


Figure 3.4: Target ellipse and the two tangent lines through the origin that give the two bearings θ_1 and θ_2.

When the angle span $[\theta_1, \theta_2]$ is calculated we can decide the bearing of each predicted measurement $\mathbf{y}_k^j(\mathbf{x}_k)$ by assigning equally spaced angles with distance $(\theta_2 - \theta_1)/m_k$ between them in the angle span. To calculate the corresponding range for each angle, such that the predicted measurements are located on the contour of the ellipse, we use the same formulas given in Granström et al. [2011b]. This is done by expressing the Cartesian coordinates as polar coordinates given by

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} r\cos(\theta) \\ r\sin(\theta) \end{bmatrix}. \tag{3.34}$$

Inserting this in (3.28) and using the resulting coordinates (x^e, y^e) in (3.29), we obtain a second degree polynomial in r. The result yields

$$1 = \left(\frac{C_3^2}{a^2}+\frac{C_4^2}{b^2}\right)r^2+2\left(\frac{C_1C_3}{a^2}+\frac{C_2C_4}{b^2}\right)r+\left(\frac{C_1^2}{a^2}+\frac{C_2^2}{b^2}\right) = A^*r^2+2B^*r+C^*, \tag{3.35}$$

where C_1 and C_2 is the same as before and the other constants are given by $C_3 = \cos(\theta)\cos(\phi) + \sin(\theta)\sin(\phi)$ and $C_4 = -\cos(\theta)\sin(\phi) + \sin(\theta)\cos(\phi)$. The solution to (3.35) is $r = -\frac{B}{A} \pm \sqrt{\frac{B^2}{A^2} - \frac{C-1}{A}}$, and since we want the solution that is closest to the sensor we use the negative solution.

Finally, we obtain the predicted measurements $\mathbf{y}_k^j(\mathbf{x}_k) = [r_k^j, \theta_k^j]^T$ for $j = 1, ..., m_k$ from the equations above. When we obtain measurements from real world targets they are unsorted in terms of range and bearing, and we need to associate each measurement to its correct predicted measurement. This is done by sorting the measurements according to their bearing, and each $\mathbf{z}_k^j$ are associated with the predicted measurement $\mathbf{y}_k^j(\mathbf{x}_k)$ that has the same position in the sorted bearing set.

The covariance matrices $\mathbf{R}_k^j$ represent ellipses that are centered in the predicted measurement points and can be rotated in two different ways. The first alternative is to use the formula

$$\mathbf{R}_k^j = \mathbf{J}_k^j \mathbf{R} \mathbf{J}_k^j, \quad \mathbf{J}_k^j = \begin{bmatrix} \cos(\theta_k^j) & -r_k^j\sin(\theta_k^j) \\ \sin(\theta_k^j) & r_k^j\cos(\theta_k^j) \end{bmatrix}, \tag{3.36}$$

where $\mathbf{R} = \mathrm{diag}[\sigma_x^2, \sigma_y^2]$ is the measurement covariance matrix and (r_k^j, θ_k^j) is the range and bearing of each measurement generating point $\mathbf{y}_k^j(\mathbf{x}_k)$. The Jacobian matrix $\mathbf{J}_k^j$ is computed from the polar coordinate function $\mathbf{h}(r_k^j, \theta_k^j)$. The second alternative is to rotate the ellipse along the tangent line of the target ellipse, and the formula is given by

$$\mathbf{R}_k^j = \mathbf{R}_\phi(\phi_k^j)\mathbf{R}\mathbf{R}_\phi(\phi_k^j)^T, \quad \mathbf{R}_\phi(\phi_k^j) = \begin{bmatrix} \cos(\phi_k^j) & -\sin(\phi_k^j) \\ \sin(\phi_k^j) & \cos(\phi_k^j) \end{bmatrix}, \tag{3.37}$$

where the angle ϕ_k^j is the rotation angle corresponding to the tangent line through $\mathbf{y}_k^j$, and $\mathbf{R}_\phi$ is the counterclockwise rotation matrix. An example for a target ellipse with state vector $\mathbf{x}_k = [5, 5, 3, 1, 5, 1.5]^T$ and measurement covariance matrix $\mathbf{R} = \mathrm{diag}[0.4^2, 0.25^2]$, is shown in Figure 3.5a and 3.5b for the Jacobian and tangent rotation respectively. The difference is clearly significant where the Jacobian rotation gives more uncertainty perpendicularly to the target contour as opposed to the tangent uncertainty which is along the contour. How the tangent angles ϕ_k^j are computed is similar to finding the predicted measurements, where the slope of the tangent line from (3.30) is used with the $\mathrm{arctan2}$-function as shown previously.

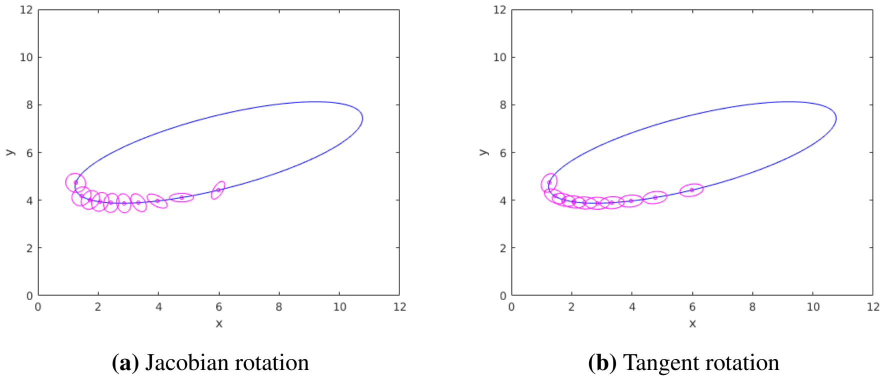


Figure 3.5: Example of a target ellipse (blue) with measurement generating points $\mathbf{y}_k^j$ (magenta) together with the corresponding covariance matrices $\mathbf{R}_k^j$ from (3.36).

The likelihood of the measurements $\mathbf{Z}_k$ from (3.9) combined with the approximation in (3.27) can now be expressed as

$$p(\mathbf{Z}_k|\mathbf{x}_k) \approx \prod_{j=1}^{m_k} \mathcal{N}(\mathbf{z}_k^j; \mathbf{y}_k^j(\mathbf{x}_k), \mathbf{R}_k^j) = \mathcal{N}(\mathbf{z}_k^*; \mathbf{y}_k^*, \mathbf{R}_k^*), \tag{3.38}$$

where $\mathbf{z}_k^*$ and $\mathbf{y}_k^*$ are vertical vectorial concatenations of the measurements and predicted measurements respectively. The noise covariance matrix $\mathbf{R}_k^*$ is a $2m_k \times 2m_k$ block diagonal matrix with all the 2×2-matrices $\mathbf{R}_k^j$ as the blocks. Now the filtering density becomes

$$p(\mathbf{x}_k|\mathbf{Z}_{1:k}) = \frac{1}{C_p} p(\mathbf{Z}_k|\mathbf{x}_k) p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{Z}_{1:k-1}) \tag{3.39}$$

$$= \frac{1}{C_p} \mathcal{N}(\mathbf{z}_k^*; \mathbf{y}_k^*, \mathbf{R}_k^*) \mathcal{N}(\mathbf{x}_k; \mathbf{m}_{k|k-1}, \mathbf{P}_{k|k-1}) = \mathcal{N}(\mathbf{x}_k; \mathbf{m}_k, \mathbf{P}_k), \tag{3.40}$$

where $\mathbf{m}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ are computed in the EKF prediction equations (3.7) and (3.8), and C_p is the normalization constant.

To find the filtered mean and covariance in (3.40), the same procedure as in (3.14) is used. However, the Jacobian $\mathbf{H}_k$ is computed numerically since we do not have a closed form expression of the predicted measurements. This is done by adding a small perturbation constant ϵ to the measurement function $\mathbf{y}_k(\mathbf{x}_k)$ for each variable in the state vector $\mathbf{x}_k$ and calculate the $2m_k \times 1$ vector

$$\mathbf{h}_k^i = \frac{\mathbf{y}_k(\mathbf{x}_k[i+\epsilon]) - \mathbf{y}_k(\mathbf{x}_k)}{\epsilon} \tag{3.41}$$

for each $i = 1, ..., n_x$ and put them together in the $2m_k \times n_x$-matrix $\mathbf{H}_k = [\mathbf{h}_k^i]_{i=1}^{n_x}$. Here the notation $\mathbf{x}_k[1+\epsilon] = [x_k + \epsilon, y_k, v_{x,k}, v_{y,k}, a_k, b_k]^T$ for $i = 1$. This yields the contour EKF (CEKF) equations

$$\mathbf{S}_{k|k-1} = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k^* \tag{3.42}$$

$$\mathbf{W}_{k|k-1} = \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_{k|k-1}^{-1} \tag{3.43}$$

$$\mathbf{m}_k = \mathbf{m}_{k|k-1} + \mathbf{W}_{k|k-1}[\mathbf{z}_k^* - \mathbf{y}_k^*] \tag{3.44}$$

$$\mathbf{P}_k = \mathbf{P}_{k|k-1} - \mathbf{W}_{k|k-1} \mathbf{S}_{k|k-1}^{-1} \mathbf{W}_{k|k-1}^T, \tag{3.45}$$

which is used to obtain estimates of the state vector and its covariances. These equations are almost identical to the regular EKF equations in (3.17) and (3.18), the only difference comes from the concatenated vectors $\mathbf{z}_k^*$, $\mathbf{y}_k^*$ and block diagonal matrix $\mathbf{R}_k^*$. The filtering algorithm is shown in Algorithm 1.

```
input  : Data from target $\mathbf{Z}_{1:T}$, parameters $\mathbf{Q}$, $\mathbf{R}$, $\Delta t_k$
output: Filtered mean $\mathbf{m}_k$ and covariance $\mathbf{P}_k$ for $k = 1, ..., T$
Initialize filter state vector estimate $\mathbf{m}_0$ and covariance matrix $\mathbf{P}_0$ ;
for $k = 1$ to $T$ do
    Compute predictions $\mathbf{m}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ ;
    Find $m_k = |\mathbf{Z}_k|$ predicted measurements $\mathbf{y}_k^j$ on the predicted ellipse ;
    Calculate the Jacobian matrix $\mathbf{H}_k$ numerically from $\mathbf{y}_k^*$ ;
    Compute the block diagonal noise covariance matrix $\mathbf{R}_k^*$ from $\mathbf{R}$ and $\mathbf{y}_k^j$ ;
    Compute the filtered values $\mathbf{m}_k$ and $\mathbf{P}_k$.
end
```

Algorithm 1: Contour EKF algorithm.

3.3 Random matrix

In this section we present an alternative filtering method where the extent ellipse is treated as a random matrix, and the kinematic variables are modelled in the same way as before. The random matrix method was first introduced by Koch [2008], and the total state consists of the kinematic state vector

$$\mathbf{x}_k = [x_k, y_k, v_{x,k}, v_{y,k}]^T, \tag{3.46}$$

and $\mathbf{X}_k$ which is a symmetric positive definite (SPD) random matrix that describes the extent of the target ellipse. This matrix has dimensions 2×2 in the case of a 2D-plane tracking problem, and this case is used from now on. The main idea is to estimate $\mathbf{x}_k$ through linear Kalman filtering, and $\mathbf{X}_k$ from its own filtering equations, hence they are treated separately. The target ellipse is represented by the points (x, y) given by the equation

$$([x,y]^T - \mathbf{H}\mathbf{x}_k)^T \mathbf{X}_k^{-1}([x,y]^T - \mathbf{H}\mathbf{x}_k) = 1, \tag{3.47}$$

where $\mathbf{H} = [\mathbf{I}_2, \mathbf{0}_2]$ is the measurement model matrix not to be confused with the matrix $\mathbf{H}_k$ for CEKF which is changing for each time step k. The position (x_k, y_k) is the center of the ellipse and $\mathbf{X}_k$ can be viewed as a covariance matrix defining the extent. We will present the random matrix approach from Feldmann et al. [2011] because Schuster and Reuter [2015] used it for single target tracking with clutter, and we want to reproduce this filter.

3.3.1 Prediction

The transition density of the kinematic state in (3.46) is given in the same way as in (3.4), but without the extent variables a_k and b_k which results in the dynamic model matrix

$$\mathbf{F}_{k-1} = \begin{bmatrix} 1 & 0 & \Delta t_k & 0 \\ 0 & 1 & 0 & \Delta t_k \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.48}$$

Hence the kinematic prediction becomes the same as in (3.7) and (3.8), but with four instead of six state variables.

In the extent process model, the transition density $p(\mathbf{X}_k|\mathbf{X}_{k-1})$ is a Wishart density, because it ensures that the extent matrix $\mathbf{X}_k$ stays SPD for all time steps k. It is given by

$$\begin{aligned} p(\mathbf{X}_k|\mathbf{X}_{k-1}) &= \mathcal{W}_2(\mathbf{X}_k; \eta_{k|k-1}, \mathbf{X}_{k-1}/\eta_{k|k-1}) \\ &\propto |\mathbf{X}_k|^{(\eta_{k|k-1}-3)/2} \exp\{-\operatorname{tr}(\mathbf{X}_{k-1}^{-1}\mathbf{X}_k)/2\}, \end{aligned} \tag{3.49}$$

for the two dimensional case (hence $\mathcal{W}_2(\cdot)$) where $|\mathbf{X}_k|$ is the determinant of $\mathbf{X}_k$ and $\operatorname{tr}(\mathbf{A}) = \sum_{i=1}^n \mathbf{A}_{i,i}$ is the trace of the $n \times n$ matrix $\mathbf{A} = (\mathbf{A}_{i,j})_{i=1,j=1}^n$. The degrees of freedom is given by the update $\eta_{k|k-1} = \eta^* e^{-\Delta t_k/\tau}$ where η^* and τ are constant extension evolution parameters. Note that with constant time interval $\Delta t_k = \Delta t$ the degrees of freedom also becomes a constant in time, hence $\eta_{k|k-1} = \eta^* e^{-\Delta t/\tau} = \eta$. When simulating the extent matrix this parameter will determine the uncertainty in the process, because the variance is given as $\operatorname{Var}_{\mathbf{X}_k}[\mathbf{X}_k|\mathbf{X}_{k-1}] \propto 1/\eta_{k|k-1}$. This is a result from the variance of the Wishart density (3.49) (see Gupta and Nagar [2000] for more details).

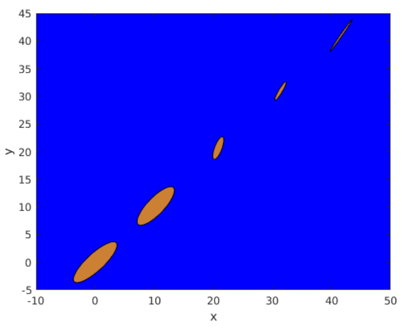
Example 3.2:
By using the transition density for the kinematics and extent (3.49), it is possible to sample

a target ellipse process. The parameter settings are

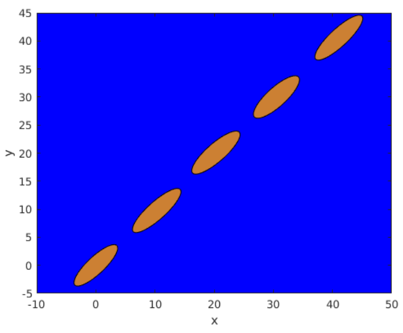
$$T = 20, \quad \Delta t_k = 2.5, \quad \mathbf{Q} = \mathrm{diag}[0.1^2, 0.1^2, 0.01^2, 0.01^2], \tag{3.50}$$

$$\mathbf{x}_0 = [0, 0, 1, 1]^T, \quad \mathbf{X}_0 = \mathbf{R}_\phi(\phi_0) \begin{bmatrix} 5^2 & 0 \\ 0 & 1.5^2 \end{bmatrix} \mathbf{R}_\phi(\phi_0), \tag{3.51}$$

and the degrees of freedom is $\eta_{k|k-1} = \eta$ because the time interval is a constant. The initial heading angle is $\phi_0 = \mathrm{arctan2}(1, 1) = \pi/4$. The simulation results are shown in Figure 3.6a and 3.6b for $\eta = 10$ and $\eta = 1000$ respectively, and it is evident what impact the degrees of freedom has to the extent process. For $\eta = 10$ the target is changing extent in an irregular pattern, while for $\eta = 1000$ it has approximately the same extent for each sample. $\square$



(a) $\eta = 10$



(b) $\eta = 1000$

Figure 3.6: Sampled target process for random matrix approach with two values for η from example 3.2.

For the object extension part it is assumed that the filtering density at the previous time step is given by

$$\begin{aligned} p(\mathbf{X}_{k-1}|\mathbf{Z}_{1:k-1}) &= \mathcal{IW}_2(\mathbf{X}_{k-1}; \nu_{k-1}, \mathbf{M}_{k-1}) \\ &\propto |\mathbf{X}_{k-1}|^{-\nu_{k-1}/2} \exp\{-\operatorname{tr}(\mathbf{M}_{k-1}\mathbf{X}_{k-1}^{-1})/2\}, \end{aligned} \tag{3.52}$$

where $\mathbf{M}_{k-1}$ is the filtering estimate of the random extent matrix at time step $k-1$ and ν_{k-1} is the estimated degrees of freedom. The notation $p(\mathbf{X}) = \mathcal{IW}_p(\mathbf{X}; \nu, \mathbf{A})$ means that the $p \times p$ random matrix $\mathbf{X}$ is inverse Wishart distributed with ν degrees of freedom. The expectation of the extent matrix is thus $\mathrm{E}[\mathbf{X}_{k-1}|\mathbf{Z}_{1:k-1}] = \mathbf{M}_{k-1}/(\nu_{k-1} - 3)$ from Gupta and Nagar [2000] for an inverse Wishart distribution. In Koch [2008] it is postulated that the expectation of the predicted density should be equal to the expectation of the previous filtering step

$$\mathbf{M}_{k|k-1}/(\nu_{k|k-1} - 3) = \mathbf{M}_{k-1}/(\nu_{k-1} - 3). \tag{3.53}$$

In addition it is stated that the degrees of freedom $\nu_{k|k-1}$ should be decreasing with increasing length of each time interval Δt_k. This is because the variance in the estimates should be larger when fewer samples are obtained from the sensor per time unit. The approach in Feldmann et al. [2011] is slightly different and it is assumed that the extent matrix is constant. This gives the random matrix extent prediction equations

$$\mathbf{M}_{k|k-1} = \mathbf{M}_{k-1} \tag{3.54}$$

$$\alpha_{k|k-1} = 2 + (\alpha_{k-1} - 2)e^{-\Delta t_k/\tau}, \tag{3.55}$$

where the degrees of freedom are defined as $\alpha_{k-1} = \nu_{k-1} - 3$ where ν_{k-1} is the degrees of freedom in (3.52). To make sense with the inverse Wishart distribution in (3.52) and expectation matching in (3.53) the degrees of freedom for the prediction step should be equal to the filtered value at the previous time step, i.e.

$$\mathbf{M}_{k|k-1} = \mathbf{M}_{k-1} \quad \Leftrightarrow \quad \alpha_{k|k-1} = \alpha_{k-1}. \tag{3.56}$$

The reason why Feldmann et al. [2011] assume that the predicted degrees of freedom $\alpha_{k|k-1}$ in (3.55) is different from α_{k-1} is because the variance of the extension estimate $\mathbf{M}_{k-1}$ is approximately proportional to $1/(\alpha_k - 2)$. This is inconsistent with the inverse Wishart distribution of $\mathbf{X}_{k-1}$ in (3.52), but is justified by the exponential increase of variance over time. The variance of the inverse Wishart distribution in (3.52) at time step k is given by

$$\mathbf{V}_k = \mathrm{Var}_{\mathbf{X}_k}[\mathbf{X}_k|\mathbf{Z}_{1:k}] = \frac{\alpha_k(\mathrm{tr}\,\mathbf{M}_k)\mathbf{M}_k + (\alpha_k + 2)\mathbf{M}_k^2}{(\alpha_k + 1)(\alpha_k - 2)}, \tag{3.57}$$

and this expresses the uncertainty of the filtering estimate $\mathbf{M}_k$.

3.3.2 Filtering

In the filtering step, Feldmann et al. [2011] assume that the object extension $\mathbf{X}_k$ is not part of the estimation problem for the kinematic vector $\mathbf{x}_k$. Hence the kinematic filtering equations are similar to the ordinary Kalman filter given in (3.17) and (3.18), but the likelihoods are different and thus give different filtering equations. The random matrix measurement likelihood is given by

$$p(\mathbf{Z}_k|m_k, \mathbf{x}_k, \mathbf{X}_k) = \prod_{j=1}^{m_k} \mathcal{N}(\mathbf{z}_k^j; \mathbf{H}\mathbf{x}_k, \mathbf{Y}_{k|k-1}), \tag{3.58}$$

where $\mathbf{Y}_{k|k-1} = z\mathbf{M}_{k|k-1} + \mathbf{R}^{\mathrm{RM}}$ is the predicted covariance of a single measurement. The scaling factor z allows us to account for different spreads of measurements around the target ellipse, and with a value of $z = 1/4$ the measurement spread is close to an uniform distribution. The random matrix measurement covariance matrix $\mathbf{R}^{\mathrm{RM}}$ models the sensor noise in the measurements, and it was not included in the original random matrix approach by Koch [2008].

Two important measurement quantities in the random matrix method are the centroid measurement and corresponding spread matrix defined as

$$\bar{\mathbf{z}}_k = \frac{1}{m_k}\sum_{j=1}^{m_k}\mathbf{z}_k^j = \frac{1}{m_k}\left[\sum_{j=1}^{m_k} x_k^j, \sum_{j=1}^{m_k} y_k^j\right]^T \tag{3.59}$$

$$\bar{\mathbf{Z}}_k = \sum_{j=1}^{m_k}(\mathbf{z}_k^j - \bar{\mathbf{z}}_k)(\mathbf{z}_k^j - \bar{\mathbf{z}}_k)^T. \tag{3.60}$$

The centroid measurement is supposed to capture the target center, and with a measurement spread over the whole target, the expression in (3.59) will give a reasonable center position. In a tracking problem with LIDAR as measurement sensor this centroid measurement will not give a good measure of where the target center is located. This is because the mean in (3.59) over the contour measurements from a LIDAR will be located close to the edge of the target as shown as the green point in Figure 3.7. The corresponding spread matrix in (3.60) is represented by the green ellipse, and it surrounds the detections. Hence we need a method to find a better centroid measurement from $\mathbf{Z}_k$, and from Schuster and Reuter [2015] we have

$$\bar{\mathbf{z}}_k^* = \begin{bmatrix} \min\{\min_\theta \mathbf{Z}_k[1], \max_\theta \mathbf{Z}_k[1]\} + |\max_\theta \mathbf{Z}_k[1] - \min_\theta \mathbf{Z}_k[1]|/2 \\ \min\{\min_\theta \mathbf{Z}_k[2], \max_\theta \mathbf{Z}_k[2]\} + |\max_\theta \mathbf{Z}_k[2] - \min_\theta \mathbf{Z}_k[2]|/2 \end{bmatrix}, \tag{3.61}$$

where the notation $\min_\theta \mathbf{Z}_k[1]$ means the x-coordinate of the measurement point in $\mathbf{Z}_k$ with lowest bearing θ. The centroid measurement is thus given as the center of the line connecting the maximum and minimum values in the θ-coordinate. We take the minimum of these points in both x and y-coordinate so that we get a general rule for computing $\mathbf{z}_k^* = [x_k^*, y_k^*]^T$ in all four quadrants. This is shown as the blue point in Figure 3.7, and is generally a better estimate of the target center. The blue ellipse represents the spread matrix $\bar{\mathbf{Z}}_k^*$ which is given by

$$\bar{\mathbf{Z}}_k^* = \sum_{j=1}^{m_k}(\mathbf{z}_k^j - \bar{\mathbf{z}}_k^*)(\mathbf{z}_k^j - \bar{\mathbf{z}}_k^*)^T, \tag{3.62}$$

i.e. the spread matrix from (3.60) computed with the new centroid measurement from (3.61).

In the paper by Koch [2008] the likelihood is given as

$$\begin{aligned} p(\mathbf{Z}_k|m_k, \mathbf{x}_k, \mathbf{X}_k) &= \prod_{j=1}^{m_k}\mathcal{N}(\mathbf{z}_k^j; \mathbf{H}\mathbf{x}_k, \mathbf{X}_k) \\ &\propto \mathcal{N}(\bar{\mathbf{z}}_k; \mathbf{H}\mathbf{x}_k, \mathbf{X}_k/m_k)\mathcal{W}_2(\bar{\mathbf{Z}}_k; m_k - 1, \mathbf{X}_k), \end{aligned} \tag{3.63}$$

but with the likelihood in (3.58) it is not possible to find an analytical solution to the posterior density $p(\mathbf{x}_k, \mathbf{X}_k|\mathbf{Z}_k)$. Hence the assumption about a constant extent matrix is made, and the uncertainty coming with the estimation of $\mathbf{X}_k$ is ignored. The kinematic

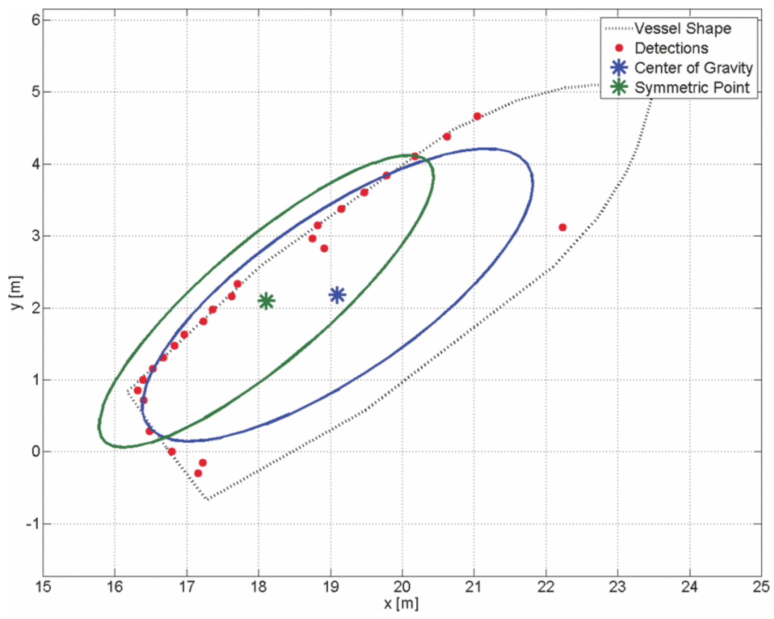


Figure 3.7: Original centroid measurement (symmetric point) from (3.59) and LIDAR centroid measurement (center of gravity) from (3.61) with corresponding target ellipses. The detections comes from a LIDAR in $(0, 0)$, and the true target size and shape is outlined as well. Figure is from Schuster and Reuter [2015].

filtering equations are thus given by

$$\mathbf{S}^*_{k|k-1} = \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^T + \frac{\mathbf{Y}_{k|k-1}}{m_k} \tag{3.64}$$

$$\mathbf{K}_{k|k-1} = \mathbf{P}_{k|k-1}\mathbf{H}(\mathbf{S}^*_{k|k-1})^{-1} \tag{3.65}$$

$$\mathbf{m}_k = \mathbf{m}_{k|k-1} + \mathbf{K}_{k|k-1}(\bar{\mathbf{z}}^*_k - \mathbf{H}\mathbf{m}_{k|k-1}) \tag{3.66}$$

$$\mathbf{P}_k = \mathbf{P}_{k|k-1} - \mathbf{K}_{k|k-1}\mathbf{S}^*_{k|k-1}\mathbf{K}^T_{k|k-1}, \tag{3.67}$$

where $\bar{\mathbf{z}}^*_k$ is used as the centroid measurement instead of $\bar{\mathbf{z}}_k$. These are similar to the EKF filtering equations, but the innovation covariance and gain are denoted by $\mathbf{S}^*_{k|k-1}$ and $\mathbf{K}_{k|k-1}$ in the random matrix filter.

The limiting case of a non-random $\mathbf{X}_k$ is maintained in the extent filtering and to obtain the estimate $\mathbf{M}_k$, Feldmann et al. [2011] uses the innovation matrix $\mathbf{N}_{k|k-1}$ given by Koch [2008]

$$\mathbf{N}_{k|k-1} = (\bar{\mathbf{z}}_k - \mathbf{H}\mathbf{m}_{k|k-1})(\bar{\mathbf{z}}_k - \mathbf{H}\mathbf{m}_{k|k-1})^T. \tag{3.68}$$

Here the original centroid measurement is used because it gives the expected value of $\mathbf{N}_{k|k-1}$ to be

$$\mathrm{E}_{\mathbf{N}_{k|k-1}}[\mathbf{N}_{k|k-1}|\mathbf{Z}_{1:k-1}, \mathbf{X}_k = \mathbf{M}_{k|k-1}] = \mathbf{S}^*_{k|k-1}. \tag{3.69}$$

This is not the case if we use $\bar{\mathbf{z}}_k^*$ in (3.68) instead of $\bar{\mathbf{z}}_k$, and which is not accounted for by Schuster and Reuter [2015]. Similarly for the scattering matrix $\bar{\mathbf{Z}}_k$ given in (3.60) the expectation is

$$\mathrm{E}_{\bar{\mathbf{Z}}_k}[\bar{\mathbf{Z}}_k|\mathbf{Z}_{1:k-1}, \mathbf{X}_k = \mathbf{M}_{k|k-1}] = (m_k - 1)\mathbf{Y}_{k|k-1}, \tag{3.70}$$

and if we use $\bar{\mathbf{Z}}_k^*$ from (3.62) instead, the result will not be the same. Hence we have a problem with these two expectations, when using the derivation from Feldmann et al. [2011] with the new centroid measurement $\bar{\mathbf{z}}_k^*$. This can give tracking results that have high absolute errors and are not consistent. Also the rule for computing the centroid measurement will give more deviant results for other target headings than shown in Figure 3.7. For instance if only the side or rear part is visible to the LIDAR, the centroid will be computed to be on the target contour.

Now two quantities $\hat{\mathbf{N}}_{k|k-1}$ and $\hat{\mathbf{Y}}_{k|k-1}$ are generated such that they are proportional to $\mathbf{X}_k = \mathbf{M}_{k|k-1}$ and are SPD matrices. This is done by using the Cholesky decomposition of matrices, so that we can write $\mathbf{M}_{k|k-1} = (\mathbf{M}_{k|k-1})^{1/2}(\mathbf{M}_{k|k-1})^{T/2}$ where $\mathbf{A}^{T/2}$ means $\mathbf{A}^{1/2}$ transposed for an arbitrary SPD matrix $\mathbf{A}$. The generated values are given by

$$\hat{\mathbf{N}}_{k|k-1} = (\mathbf{M}_{k|k-1})^{1/2}(\mathbf{S}_{k|k-1})^{-1/2}\mathbf{N}_{k|k-1}(\mathbf{S}_{k|k-1})^{-T/2}(\mathbf{M}_{k|k-1})^{T/2} \tag{3.71}$$

$$\hat{\mathbf{Z}}_{k|k-1} = (\mathbf{M}_{k|k-1})^{1/2}(\mathbf{Y}_{k|k-1})^{-1/2}\bar{\mathbf{Z}}_k^*(\mathbf{Y}_{k|k-1})^{-T/2}(\mathbf{M}_{k|k-1})^{T/2} \tag{3.72}$$

and from this the extent filtering equations are given as

$$\alpha_k = \alpha_{k|k-1} + m_k \tag{3.73}$$

$$\mathbf{M}_k = (\alpha_{k|k-1}\mathbf{M}_{k|k-1} + \hat{\mathbf{N}}_{k|k-1} + \hat{\mathbf{Z}}_{k|k-1})/\alpha_k. \tag{3.74}$$

These are in close analogy to the original filtering equations in Koch [2008], but are more a result of constructed quantities than derived from the distributions assigned to the variables. Although the method of Feldmann et al. [2011] is based on several assumptions and generated values, it does not ignore the measurement noise given by the covariance matrix $\mathbf{R}_k$ in (3.58), which is an important parameter because the sensor most likely gives noisy measurements. The final random matrix filter is given in Algorithm 2.

```
input  : Data from target $\mathbf{Z}_{1:T}$, parameters $\tau$, $\eta$, $\mathbf{Q}$, $\Delta t_k$, $z$
output: Filtered mean $\mathbf{m}_k$, covariance $\mathbf{P}_k$, extent matrix $\mathbf{M}_k$ and variance $\mathbf{V}_k$ for
        $k = 1, ..., T$
Initialize filter state vector estimate $\mathbf{m}_0$, covariance matrix $\mathbf{P}_0$, degrees of freedom
  $\alpha_0$ and extent matrix $\mathbf{M}_0$ ;
for $k = 1$ to $T$ do
    Compute predictions $\mathbf{m}_{k|k-1}$, $\mathbf{P}_{k|k-1}$, $\alpha_{k|k-1}$ and $\mathbf{M}_{k|k-1}$ ;
    Calculate the centroid position $\bar{\mathbf{z}}_k^*$ and $\bar{\mathbf{Z}}_k^*$ from the measurements ;
    Compute the generated values $\hat{\mathbf{N}}_{k|k-1}$ and $\hat{\mathbf{Z}}_{k|k-1}$ from the innovation matrices
      and gain matrix ;
    Compute the kinematic filtered mean $\mathbf{m}_k$ and covariance $\mathbf{P}_k$. ;
    Compute the extent filtered matrix $\mathbf{M}_k$ and degrees of freedom $\alpha_k$ ;
    Compute the extent variance $\mathbf{V}_k$.
end
```

Algorithm 2: Random matrix algorithm.

Chapter 4

Extended object tracking with clutter

The filtering methods presented in the previous chapter are based on the assumption that no clutter measurements appear in the set $\mathbf{Z}_k$. In this chapter we describe the more realistic situation where measurements can represent clutter. The measurement set is defined as $\mathbf{Z}_k = \Theta_k \cup K_k = \{\mathbf{z}_k^1, ..., \mathbf{z}_k^{m_k}\}$, where Θ_k and K_k are the sets of measurements from target and clutter, respectively. We define the number of target measurements as $|\Theta_k| = n_k^t$ and the number of clutter measurements as $|K_k| = n_k^c$, such that $m_k = n_k^t + n_k^c$. The chapter is beginning with an introduction of GPDA followed by a presentation of the GPDA filter for CEKF and random matrix. In the last section, algorithms for initialization of the target ellipse and shrinking the hypothesis space are presented.

4.1 Generalized Probabilistic Data Association

Traditionally the probabilistic data association (PDA) filter have been used in point target tracking as presented in Bar-Shalom and Li [1995], where it is assumed that the target gives at most one measurement for each time step. In the following we will extend this assumption and allow that multiple measurements can originate from target. This idea was first presented as the multiple detection PDA (MD-PDA) in Habtemariam et al. [2011]. Later, the generalized probabilistic data association (GPDA) filter was derived in Schubert et al. [2012]. The GPDA was later implemented with the random matrix filter presented in section 3.3 by Schuster and Reuter [2015]. In this section the GPDA will be derived in a slightly different way than done before.

The GPDA filter is based on the same assumptions as the MD-PDA given by

- There is only one target of interest.
- The track has been initialized.

- The past information of the target is approximately summarized as

$$p(\mathbf{x}_k|\mathbf{Z}_{1:k-1}) \approx \mathcal{N}(\mathbf{x}_k; \mathbf{m}_{k|k-1}, \mathbf{P}_{k|k-1}). \tag{4.1}$$

- At each time step a validation region Γ_k is set up to validate each measurement.
- Among the validated measurements, one or more can originate from the target.
- The clutter measurements are modelled with uniform spatial distribution and Poisson cardinal distribution within the validation region.
- Target detections occur independently over time with known probability P_D.

These assumptions are similar to the traditional PDA, except for the number of points generated from target. In addition, the GPDA assumes that we maximally can have n^{max} points from target. This can be set prior to the tracking process, but when the measurements are given, we know that $n_k^t \leq m_k$. However, if $n^{max} < m_k$ we know that some of the measurements are not from target.

The validation region is an elliptical region where every measurement that is inside the region is validated. Originally it was defined in Bar-Shalom and Li [1995] as

$$\Gamma_k = \{(x, y) : ([x, y]^T - \mathbf{H}\mathbf{m}_{k|k-1})^T \mathbf{S}_{k|k-1}^{-1}([x, y]^T - \mathbf{H}\mathbf{m}_{k|k-1}) < \gamma\}, \tag{4.2}$$

where $\mathbf{H}$ is the measurement model matrix used in the random matrix filtering. Here γ is the gate threshold that determines the size of the validation region. In the random matrix approach this region is well defined since we have a 2×2 innovation covariance matrix $\mathbf{S}_{k|k-1}^*$ at each time step. For the GMEKF method it is more problematic since this matrix has dimensions $2m_k \times 2m_k$ with calculated covariances in all entries, so we can not use the 2×2 block diagonal matrices because it will exclude the covariances from the rest of the columns.

Considering the difficulties outlined above we need to choose the validation region Γ_k in another way. We choose to use a scaling of the predicted target ellipse in the extension variables, represented by the state vector

$$[x_{k|k-1}, y_{k|k-1}, v_{x,k|k-1}, v_{y,k|k-1}, \gamma_s a_{k|k-1}, \gamma_s b_{k|k-1}]^T. \tag{4.3}$$

Here γ_s is the validation region scale parameter, and it determines the extent of Γ_k which again decides the validated measurements. The reason why we choose this validation region is that the predicted ellipse $\mathbf{m}_{k|k-1}$ is available before the filtering step and give a good measure of where the measurements should appear. By increasing its size with the scaling $\gamma_s > 1$ we account for the noise in the measurements, and remove distant clutter points. Hence this method will validate the measurements in a good manner as long as the predicted ellipse don't get too small. The volume of this validation region is

$$V_k = \pi \gamma_s^2 a_{k|k-1} b_{k|k-1}, \tag{4.4}$$

which is given by the area of an ellipse. For the random matrix filter we can use the original validation region in (4.2) but to get a good comparison of the methods we choose the scaled predicted ellipse for both.

After the measurement validation we have the set $\mathbf{Z}_k = \{\mathbf{z}_k^j\}_{j=1}^{m_k}$ of validated measurements, and we use m_k to denote the number of measurements that are validated. The challenge now is to find out which of the measurements that come from target and clutter. This is referred to as the data association problem, and the GPDA filter is designed to solve it. We define the set of mutually exclusive association hypotheses when $m_k \leq n^{max}$ to be

$$\mathcal{E} = \begin{cases} E^0 & = \left\{ E_0^0 \quad \text{no detection from target} \right. \\ E^1 & = \begin{cases} E_1^1 & \text{Detection } \mathbf{z}_k^1 \text{ originated from target} \\ \vdots & \\ E_{m_k}^1 & \text{Detection } \mathbf{z}_k^{m_k} \text{ originated from target} \end{cases} \\ E^2 & = \begin{cases} E_1^2 & \text{Detections } \mathbf{z}_k^1, \mathbf{z}_k^2 \text{ originated from target} \\ E_2^2 & \text{Detections } \mathbf{z}_k^1, \mathbf{z}_k^3 \text{ originated from target} \\ \vdots & \\ E_{\binom{m_k}{2}}^2 & \text{Detections } \mathbf{z}_k^{m_k-1}, \mathbf{z}_k^{m_k} \text{ originated from target} \end{cases} \\ & \vdots \\ E^{m_k} & = \left\{ E_1^{m_k} \quad \text{all detections originated from target.} \right. \end{cases}$$

A hypothesis E_i^j can from this definition be treated as a set of target generated points. The posterior pdf is defined according to the total probability theorem to be a weighted sum over all association hypotheses

$$p(\mathbf{x}_k|\mathbf{Z}_{1:k}) = \sum_{E_i^j \in \mathcal{E}} p(\mathbf{x}_k|E_i^j, \mathbf{Z}_{1:k}) P(E_i^j|\mathbf{Z}_{1:k}), \tag{4.5}$$

where the hypothesis conditional filtering density $p(\mathbf{x}_k|E_i^j, \mathbf{Z}_{1:k})$ needs to be computed from the filtering method for each hypothesis E_i^j, and this is presented in the next sections of this chapter. When not considering E_0^0, the index $j = 1, ..., m_k$ is the number of target generated points and $i = 1, ..., \binom{m_k}{j}$ is the hypothesis index within the hypothesis space E^j. We will assume that $P(E_0^0) = 0$ because in the simulated and real LIDAR data the target always exists and give at least one detection. This assumption makes the GPDA derivation different from the approaches in Schubert et al. [2012], where it is possible that the target do not exist or is not detected. When the number of measurements is a large number, the computation of all the association pdf's becomes intractable. Hence it is important to narrow down the hypothesis space so that the algorithms run efficiently.

The association probabilities $P(E_i^j|\mathbf{Z}_{1:k})$ are denoted as the weights $\beta_k^{i,j}$ for each time step k, and can be written as

$$\beta_k^{i,j} = P(E_i^j|\mathbf{Z}_k, m_k, \mathbf{Z}_{1:k-1}). \tag{4.6}$$

We write it in this form because it enables probabilistic inference on the number of measurements m_k and their positions. Rewriting using Bayes' formula yields

$$\beta_k^{i,j} = \frac{1}{C_\beta} p(\mathbf{Z}_k | E_i^j, m_k, \mathbf{Z}_{1:k-1}) P(E_i^j | m_k, \mathbf{Z}_{1:k-1}), \tag{4.7}$$

where C_β is the normalization constant such that $\beta_0^0 + \sum_{j=1}^{m_k} \sum_{i=1}^{\binom{m_k}{j}} \beta_k^{i,j} = 1$. As a result of this normalization, all constant terms that do not depend on the association hypothesis E_i^j can be neglected because they will cancel out. The association likelihood in (4.7) can be expressed by

$$p(\mathbf{Z}_k | E_i^j, m_k, \mathbf{Z}_{1:k-1}) = p_{sp}(\Theta_k | E_i^j, m_k, \mathbf{Z}_{1:k-1}) p_{sp}(K_k | E_i^j, m_k, \mathbf{Z}_{1:k-1}), \tag{4.8}$$

where it is assumed that the target generated measurements are independent of the clutter measurements. Both the densities in (4.8) are spatial densities (p_{sp}), because we have conditioned on m_k and E_i^j which leaves no uncertainty about the cardinalities n_k^t and n_k^c. When the hypothesis E_i^j is given, we know how many target and clutter measurements we have, from the index j. In the original formulation of the GPDA in Schubert et al. [2012], these cardinalities are modelled as a uniform and Poisson, and this represents the difference in our approach. However, it is assumed that both n_k^t and n_k^c have prior densities given by

$$P(n_k^t) = \frac{1}{n^{max}}, \quad \text{for } n_k^t \geq 1 \tag{4.9}$$

$$P(n_k^c) = \frac{(\lambda V_k)^{m_k - j}}{(m_k - j)!} e^{-\lambda V_k}. \tag{4.10}$$

Here we have assumed that the target generated points follow a discrete uniform distribution, while the number of clutter measurements are Poisson distributed with parameter λ. The clutter measurements are modelled as a Poisson point process, which means that the spatial density is given by

$$p_{sp}(K_k | E_i^j, m_k, \mathbf{Z}_{1:k-1}) = (V_k)^{-(m_k - j)}. \tag{4.11}$$

This means that the clutter points are distributed uniformly over the validation region volume V_k. Since this expression is dependent on the hypothesis E_i^j it will be a part of the association weights $\beta_k^{i,j}$ in (4.7). The spatial density $p_{sp}(\Theta_k | E_i^j, m_k, \mathbf{Z}_{1:k-1})$ will be given in the next sections for each of the filtering methods.

The prior density in (4.7) will also be equal in the two filtering methods. A simple way to model it is to assume a discrete uniform distribution, i.e. $P(E_i^j | m_k, \mathbf{Z}_{1:k-1}) = (\sum_{j=0}^{m_k} \binom{m_k}{j})^{-1} = 2^{-m_k}$. This is done in both Schubert et al. [2012] and Schuster and Reuter [2015], and in our case this will make the association weights only dependent on the association likelihood in (4.8), and not the Poisson parameter λ from (4.10). The prior densities given in (4.9) and (4.10) will influence the association hypothesis prior density in (4.7) given the number of measurements m_k, and we can find this expression by making inference on m_k.

First we assume that the prior density is independent of past measurements, i.e.

$$P(E_i^j|m_k, \mathbf{Z}_{1:k-1}) = P(E_i^j|m_k). \tag{4.12}$$

Then we observe that the joint density $P(E_i^j, n_k^t = j|m_k) = P(E_i^j|m_k)$ because the event E_i^j will be inside the event $n_k^t = j$. This can be used to obtain an expression for the prior density by using Bayes' rule, and we get that

$$\begin{aligned} P(E_i^j|m_k) = P(E_i^j, n_k^t = j|m_k) &= \frac{P(E_i^j|n_k^t = j, m_k)P(m_k|n_k^t = j)P(n_k^t = j)}{P(m_k)} \\ = \frac{1}{C_m}\binom{m_k}{j}^{-1}\frac{(\lambda V_k)^{m_k-j}}{(m_k-j)!}e^{-\lambda V_k}\frac{1}{n^{max}} &\propto \frac{j!}{(\lambda V_k)^j} \end{aligned} \tag{4.13}$$

Here we have assumed that the density $P(E_i^j|n_k^t = j, m_k)$ is uniform over the hypothesis space E^j, which contains $\binom{m_k}{j}$ hypotheses. The density $P(m_k|n_k^t = j)$ is recognized as the clutter cardinal density $P(n_k^c = m_k - j)$ because of the relation $m_k = n_k^t + n_k^c$. If we take the product of all the densities discussed so far, and excluding the constant terms, we get that the association probabilities in (4.7) are

$$\beta_k^{i,j} \propto p_{sp}(\Theta_k|E_i^j, m_k, \mathbf{Z}_{1:k-1})\frac{j!}{\lambda^j}. \tag{4.14}$$

These weights will prefer hypotheses with a high j because the factor $j!$ increases more rapidly than λ^j for $j = 1, ..., m_k$ and $\lambda < m_k$. This can happen if the spatial density term has the same magnitude as the prior term, but if $p_{sp}(\Theta_k|E_i^j, m_k, \mathbf{Z}_{1:k-1})$ is much higher it will be less significant. In the next sections we will present this target density and the corresponding association weights $\beta_k^{i,j}$ for both filtering methods presented in Chapter 3.

4.2 Contour GPDA filter

First we present the filtering equations for the GPDA algorithm using the CEKF from section 3.2. This will be referred to as the C-GPDA from now on. The target generated density is the likelihood of the concatenated measurements $\tilde{\mathbf{z}}_k^{i,j}$ that are assumed generated by the target in E_i^j. In the same way as $p(\mathbf{z}_k^j|\mathbf{x}_k)$ from (3.25) it is given by the convolution

$$\begin{aligned} p_{sp}(\Theta_k|E_i^j, m_k, \mathbf{Z}_{1:k-1}) &= \int p(\tilde{\mathbf{z}}_k^{i,j}|E_i^j, m_k, \mathbf{Z}_{1:k-1}, \mathbf{x}_k)p(\mathbf{x}_k|E_i^j, m_k, \mathbf{Z}_{1:k-1})d\mathbf{x}_k \\ &= \int \mathcal{N}(\tilde{\mathbf{z}}_k^{i,j}; \tilde{\mathbf{y}}_k^j(\mathbf{x}_k), \mathbf{R}_k^j)\mathcal{N}(\mathbf{x}_k; \mathbf{m}_{k|k-1}, \mathbf{P}_{k|k-1})d\mathbf{x}_k, \end{aligned} \tag{4.15}$$

where $\tilde{\mathbf{z}}_k^{i,j} \in \mathbb{R}^{2j}$, $\tilde{\mathbf{y}}_k^j(\mathbf{x}_k) : \mathbb{R}^{n_x} \to \mathbb{R}^{2j}$ and $\tilde{\mathbf{R}}_k^j \in \mathbb{R}^{2j \times 2j}$ are the concatenated measurement vector for the target measurements $\mathbf{z}_k^{i,j} \in E_i^j$, and corresponding measurement model function and noise covariance matrix. To deal with the nonlinear measurement function $\tilde{\mathbf{y}}_k^j(\mathbf{x}_k)$ we do a linearization akin to (3.11), and the expectation and covariance is found from (3.12) and (3.13). Hence we end up with a joint density similar to (3.14) and when we do the integral over $\mathbf{x}_k$ from (4.15) the result becomes

$$p_{sp}(\Theta_k|E_i^j, m_k, \mathbf{Z}_k) = \mathcal{N}(\tilde{\mathbf{z}}_k^{i,j}; \tilde{\mathbf{y}}_k^j(\mathbf{m}_{k|k-1}), \mathbf{H}_k^j \mathbf{P}_{k|k-1}(\mathbf{H}_k^j)^T + \tilde{\mathbf{R}}_k^j), \tag{4.16}$$

where the covariance term is the $2j \times 2j$ innovation covariance matrix

$$\mathbf{S}_{k|k-1}^j = \mathbf{H}_k^j \mathbf{P}_{k|k-1}(\mathbf{H}_k^j)^T + \tilde{\mathbf{R}}_k^j. \tag{4.17}$$

This normal density will give higher probabilities to points that are close to the predicted measurements $\tilde{\mathbf{y}}_k^j$, and distant clutter points will be assigned to lower values using this pdf. Now that we have obtained all the densities in (4.7), we get the final expression for the association weights

$$\beta_k^{i,j} = \frac{1}{C_\beta} \mathcal{N}(\tilde{\mathbf{z}}_k^{i,j}; \tilde{\mathbf{y}}_k^j(\mathbf{m}_{k|k-1}), \mathbf{S}_{k|k-1}^j) \frac{j!}{\lambda^j} \tag{4.18}$$

where the $2j \times 2j$ innovation covariance matrix is given by

$$\mathbf{S}_{k|k-1}^j = \mathbf{H}_k^j \mathbf{P}_{k|k-1}(\mathbf{H}_k^j)^T + \tilde{\mathbf{R}}_k^j. \tag{4.19}$$

Observe that the innovation covariance is dependent on the hypothesis index j, which gives m_k different sized matrices dependent on how many target generated points we consider.

The next step is to find the hypothesis conditional posterior density $p(\mathbf{x}_k|E_i^j, \mathbf{Z}_{1:k})$ from (4.5). It is given by the mean vector and covariance matrix from the EKF equations (3.17) and (3.18), but for the measurements $\tilde{\mathbf{z}}_k^{i,j}$. From this we can write

$$\begin{aligned}
p(\mathbf{x}_k|E_i^j, \mathbf{Z}_{1:k}) &= \mathcal{N}(\mathbf{x}_k; \mathbf{m}_k^{i,j}, \mathbf{P}_k^{i,j}) && (4.20)\\
\mathbf{W}_{k|k-1}^j &= \mathbf{P}_{k|k-1}(\mathbf{H}_k^j)^T(\mathbf{S}_{k|k-1}^j)^{-1} && (4.21)\\
\mathbf{m}_k^{i,j} &= \mathbf{m}_{k|k-1} + \mathbf{W}_{k|k-1}^j(\tilde{\mathbf{z}}_k^{i,j} - \tilde{\mathbf{y}}_k^j) && (4.22)\\
\mathbf{P}_k^{i,j} &= \mathbf{P}_{k|k-1} - \mathbf{W}_{k|k-1}^j \mathbf{S}_{k|k-1}^j (\mathbf{W}_{k|k-1}^j)^T, && (4.23)
\end{aligned}$$

where $\mathbf{S}_{k|k-1}^j$ is given in (4.19). These equations are analogous to the CEKF equations in (3.44) and (3.45), but for the measurement vector $\tilde{\mathbf{z}}_k^{i,j}$ which gives different innovation and gain quantities.

To ensure that the filtering distribution from (4.5) is one single Gaussian, it is necessary to do a mixture reduction of the Gaussian densities $p(\mathbf{x}_k|E_i^j, \mathbf{Z}_{1:k})$. The expectation of the resulting filtering density $\mathcal{N}(\mathbf{x}_k; \mathbf{m}_k, \mathbf{P}_k)$ is given by

$$\begin{aligned}
\mathbf{m}_k = \mathrm{E}\left[\mathbf{x}_k|\mathbf{Z}_{1:k}\right] &= \sum_{j=1}^{m_k} \sum_{i=1}^{\binom{m_k}{j}} \mathrm{E}\left[\mathbf{x}_k|E_i^j, \mathbf{Z}_{1:k}\right] P(E_i^j|\mathbf{Z}_{1:k})\\
&= \sum_{j=1}^{m_k} \sum_{i=1}^{\binom{m_k}{j}} \mathbf{m}_k^{i,j} \beta_k^{i,j}.
\end{aligned} \tag{4.24}$$

This can be used to find the filtered covariance matrix as

$$
\begin{aligned}
\mathbf{P}_k &= \mathrm{E}\left[(\mathbf{x}_k - \mathbf{m}_k)(\mathbf{x}_k - \mathbf{m}_k)^T | \mathbf{Z}_{1:k}\right] \\
&= \sum_{j=1}^{m_k} \sum_{i=1}^{\binom{m_k}{j}} \mathrm{E}\left[(\mathbf{x}_k - \mathbf{m}_k)(\mathbf{x}_k - \mathbf{m}_k)^T | E_i^j, \mathbf{Z}_{1:k}\right] P(E_i^j | \mathbf{Z}_{1:k}) \\
&= \sum_{j=1}^{m_k} \sum_{i=1}^{\binom{m_k}{j}} \beta_k^{i,j} \int (\mathbf{x}_k - \mathbf{m}_k)(\mathbf{x}_k - \mathbf{m}_k)^T \mathcal{N}(\mathbf{x}_k; \mathbf{m}_k, \mathbf{P}_k) d\mathbf{x}_k \\
&= \sum_{j=1}^{m_k} \sum_{i=1}^{\binom{m_k}{j}} \beta_k^{i,j} \left(\mathbf{P}_k^{i,j} + \mathbf{m}_k^{i,j} (\mathbf{m}_k^{i,j})^T\right) - \mathbf{m}_k \mathbf{m}_k^T.
\end{aligned}
\tag{4.25}
$$

These values are used in the C-GPDA filter for the estimated mean and covariance respectively. The full derivation of the covariance in (4.25) can be found in Bar-Shalom and Li [1995]. The final algorithm is shown in Algorithm 3.

Example 4.1
Let us start with an easy example at an arbitrary time step k where $n_k^t = |\Theta_k| = 3$ measurements are from target and the number of clutter measurements $n_k^c = 1$ is fixed. This gives $m_k = 4$ measurements, and we will see how the GPDA filter finds the most likely hypothesis from the spatial coordinates of the measurements. Since we know $n_k^t = 3$ it is only necessary to investigate E^3, where there are $\binom{4}{3} = 4$ different association hypotheses given by

$$
E^3 = \begin{cases}
E_1^3 & = \quad \text{Detection } \mathbf{z}_k^1, \mathbf{z}_k^2, \mathbf{z}_k^3 \text{ from target} \\
E_2^3 & = \quad \text{Detection } \mathbf{z}_k^1, \mathbf{z}_k^3, \mathbf{z}_k^4 \text{ from target} \\
E_3^3 & = \quad \text{Detection } \mathbf{z}_k^1, \mathbf{z}_k^2, \mathbf{z}_k^4 \text{ from target} \\
E_4^3 & = \quad \text{Detection } \mathbf{z}_k^2, \mathbf{z}_k^3, \mathbf{z}_k^4 \text{ from target.}
\end{cases}
$$

From the prediction step we have $\mathbf{m}_{k|k-1} = [20, 20, 2, 2, 5, 1.5]^T$ and $\mathbf{P}_{k|k-1} = 0.01\mathbf{I}_6$. The spatial density of the clutter measurement is assumed to be uniform over the square $\Lambda = [15, 25] \times [15, 25]$, while the validation region scale is set to $\gamma_s = 4$. This gives the association weights from (4.18) to be

$$
\beta_k^{i,3} \propto \mathcal{N}(\tilde{\mathbf{z}}_k^{i,3}; \tilde{\mathbf{y}}_k^3(\mathbf{m}_{k|k-1}), \mathbf{S}_{k|k-1}^3), \tag{4.26}
$$

where the cardinal clutter density is removed because we know that $n_k^c = 1$. We sample the target measurements $\mathbf{z}_k^j = [(x_k^j, y_k^j)]^T$ for $j = 1, 2, 3$ by adding noise to the predicted measurements, i.e.

$$
\mathbf{z}_k^j = \mathbf{y}_k^j + \mathbf{r}_k^j \quad \text{where } p(\mathbf{r}_k^j) = \mathcal{N}(\mathbf{r}_k^j; [0, 0]^T, \mathbf{R}_k^j). \tag{4.27}
$$

The covariance matrix $\mathbf{R}_k$ is rotated according to the Jacobian rotation from (3.36), and $\mathbf{R} = \mathrm{diag}[0.1^2, 0.1^2]$. The clutter measurement is the last measurement in $\mathbf{Z}_k$, so that the hypothesis E_1^3 is the correct one. In Figure 4.1 the results from the C-GPDA filter are shown. The calculated association weights are $\beta_k^{1,3} = 1$ and $\beta_k^{i,3} = 0$ for $i = 2, 3, 4$, which means that the filter chose the correct hypothesis. This can be seen from the dotted cyan ellipse and the underlying red filtered ellipse in Figure 4.1, and they are close to the prediction and target measurements. Observe that the three other ellipses are moved towards the clutter measurement, and the shapes are shorter and wider than the E_1^3-ellipse which is close to the predicted ellipse. □

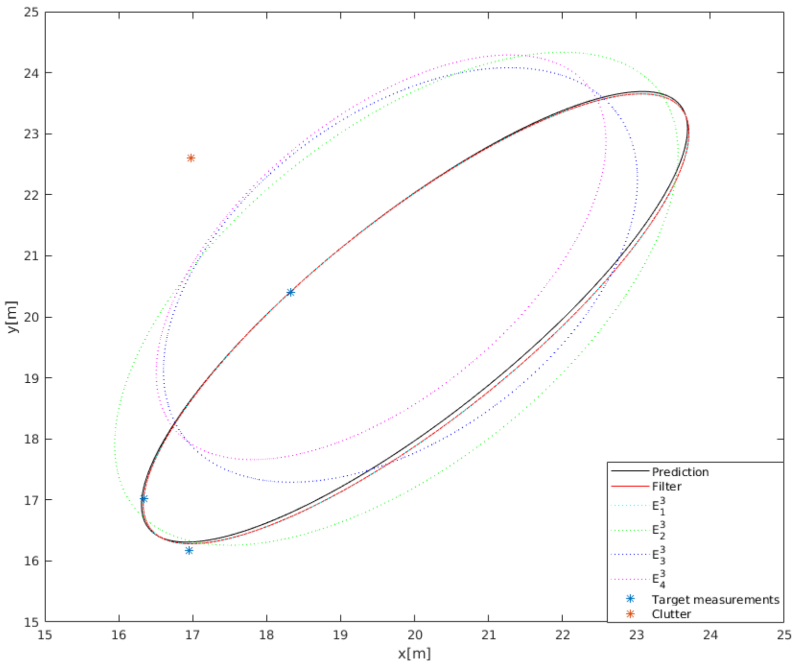


Figure 4.1: Filtered target ellipse (red) from Example 4.1 plotted with the four association hypotheses ellipses (dotted) and the measurements from target (blue *) and clutter (red *).

4.3 Random matrix GPDA filter

The GPDA filter is similar for the random matrix approach but the association probabilities $\beta_k^{i,j}$ becomes slightly different because of the extent matrix $\mathbf{X}_k$. The filter will be presented

```
input  : Data from target and clutter $\mathbf{Z}_{1:T}$, parameters $\mathbf{Q}$, $\mathbf{R}$, $\Delta t_k$, $\gamma_s$
output: Filtered mean $\mathbf{m}_k$ and covariance $\mathbf{P}_k$ for $k = 1, ..., T$
Initialize filter state vector estimate $\mathbf{m}_0$ and covariance matrix $\mathbf{P}_0$ ;
for $k = 1$ to $T$ do
    Compute predictions $\mathbf{m}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ ;
    Validate measurements by scaling the predicted ellipse with $\gamma_s$, and check if $\mathbf{z}_k^j$
      is inside it for all $j = 1, ..., m_k$ ;
    Calculate the number of association hypotheses $|\mathcal{E}|$ ;
    if $|\mathcal{E}|$ = 1 then
        Perform regular CEKF as in Algorithm 1 ;
    else
        for $j = 1, ..., m_k$ do
            Find $j$ predicted measurements $\tilde{\mathbf{y}}_k^j$ on the predicted target ellipse ;
            Calculate the Jacobian matrix $\mathbf{H}_k^j$ numerically from $\tilde{\mathbf{y}}_k^j$ ;
            Compute the block diagonal noise covariance matrix $\tilde{\mathbf{R}}_k^j$ from $\mathbf{R}$ and $\tilde{\mathbf{y}}_k^j$
              ;
            for $i = 1$ to $\binom{m_k}{j}$ do
                Compute the association weights $\beta_k^{i,j}$ ;
                Compute the filter association values $\mathbf{m}_k^{i,j}$ and $\mathbf{P}_k^{i,j}$ ;
            end
        end
        Find the filtered values $\mathbf{m}_k$ and $\mathbf{P}_k$ from the association weights and filter
          association values ;
    end
end
```

Algorithm 3: C-GPDA filter algorithm.

in the same way as in Schuster and Reuter [2015], and the posterior density is given by

$$p(\mathbf{x}_k, \mathbf{X}_k|\mathbf{Z}_k) = \sum_{E_i^j \in \mathcal{E}} p(\mathbf{x}_k, \mathbf{X}_k|\mathbf{Z}_k, E_i^j)\beta_k^{i,j}. \tag{4.28}$$

We will present the kinematic and extent filtering equations for $p(\mathbf{x}_k, \mathbf{X}_k|\mathbf{Z}_k, E_i^j)$ in the same way as in section 3.3, but first we find the association weights.

The random matrix association weights $\beta_k^{i,j}$ can be written in the same way as for the C-GPDA in (4.18). However, the target generated density is different and can be written as

$$p_{sp}(\Theta_k|E_i^j, m_k, \mathbf{Z}_{1:k-1}) = \prod_{\mathbf{z}_k^{i,j} \in E_i^j} \mathcal{N}(\mathbf{z}_k^{i,j}; \mathbf{H}\mathbf{m}_{k|k-1}, z\mathbf{M}_{k|k-1} + \mathbf{R}^{\mathrm{RM}}), \tag{4.29}$$

where the vector $\mathbf{z}_k^{i,j} = [x_k^j, y_k^j]^T$ denotes each single measurement in E_i^j that is generated by the target. Putting the target density together with the spatial clutter density gives the association probabilities

$$\beta_k^{i,j} = \frac{1}{C_\beta} \prod_{\mathbf{z}_k^{i,j} \in E_i^j} \mathcal{N}(\mathbf{z}_k^{i,j}; \mathbf{H}\mathbf{m}_{k|k-1}, z\mathbf{M}_{k|k-1} + \mathbf{R}^{\text{RM}}) \frac{j!}{\lambda^j}. \tag{4.30}$$

The random matrix filtering method is similar to the C-GPDA when it comes to the computation of the filtered mean vector $\mathbf{m}_k$, which is given by the mixture reduction in (4.24). However, the association filter update $\mathbf{m}_k^{i,j}$ for the hypothesis E_i^j is given by (3.66) as

$$\mathbf{m}_k^{i,j} = \mathbf{m}_{k|k-1} + \mathbf{K}_{k|k-1}(\bar{\mathbf{z}}_k^{i,j} - \mathbf{H}\mathbf{m}_{k|k-1}). \tag{4.31}$$

The centroid measurement $\bar{\mathbf{z}}_k^{i,j}$ is computed from (3.61), but with $\mathbf{z}_k^{i,j} \in E_i^j$ as the measurements. This gives different centroids for each hypothesis E_i^j, but the other quantities in (4.31) are equal in each hypothesis. The covariance matrix $\mathbf{P}_k^{i,j}$ is given by (3.67) and will stay the same for each association hypothesis because the gain and innovation matrices in (3.64) and (3.65) are not dependent on the measurements used. This is because $\mathbf{H}$ is the same regardless of the measurements that are used in the filtering. The mixture reduction quantity $\mathbf{P}_k$ from (4.25) then becomes

$$\mathbf{P}_k = \mathbf{P}_{k|k-1} - \mathbf{K}_{k|k-1}\mathbf{S}^*_{k|k-1}\mathbf{K}^T_{k|k-1} + \sum_{j=1}^{m_k} \sum_{i=1}^{\binom{m_k}{j}} \beta_k^{i,j} \left(\mathbf{m}_k^{i,j}(\mathbf{m}_k^{i,j})^T\right) - \mathbf{m}_k\mathbf{m}_k^T, \tag{4.32}$$

which is used in the kinematical update of the random matrix GPDA.

In the extent filtering for random matrix GPDA the association innovation matrix for hypothesis E_i^j is given by

$$\mathbf{N}_{k|k-1}^{i,j} = (\bar{\mathbf{z}}_k^{i,j} - \mathbf{H}\mathbf{m}_{k|k-1})(\bar{\mathbf{z}}_k^{i,j} - \mathbf{H}\mathbf{m}_{k|k-1})^T, \tag{4.33}$$

which analogous to (3.68) but with the association centroid measurement $\bar{\mathbf{z}}_k^{i,j}$. The corresponding generated values akin to (3.71) and (3.72) are then given by

$$\hat{\mathbf{N}}_{k|k-1}^{i,j} = (\mathbf{M}_{k|k-1})^{1/2}(\mathbf{S}^*_{k|k-1})^{-1/2}\mathbf{N}_{k|k-1}^{i,j}(\mathbf{S}^*_{k|k-1})^{-T/2}(\mathbf{M}_{k|k-1})^{T/2} \tag{4.34}$$

$$\hat{\mathbf{Z}}_{k|k-1}^{i,j} = (\mathbf{M}_{k|k-1})^{1/2}(\mathbf{Y}_{k|k-1})^{-1/2}\bar{\mathbf{Z}}_k^{i,j}(\mathbf{Y}_{k|k-1})^{-T/2}(\mathbf{M}_{k|k-1})^{T/2}. \tag{4.35}$$

Here the scattering matrix $\bar{\mathbf{Z}}_k^{i,j}$ needs to be calculated according to (3.60) for the measurements $\mathbf{z}_k^{i,j} \in E_i^j$, which is time consuming and clearly a weakness of the random matrix GPDA. The association extent update is given by (3.73) and (3.74) but with the new generated values in (4.34) and (4.35). The degrees of freedom update from (3.73) will also be dependent on E_i^j, and we get

$$\alpha_k^{i,j} = \alpha_{k|k-1} + j \tag{4.36}$$

$$\mathbf{M}_k^{i,j} = (\alpha_{k|k-1}\mathbf{M}_{k-1} + \hat{\mathbf{N}}_{k|k-1}^{i,j} + \hat{\mathbf{Z}}_{k|k-1}^{i,j})/\alpha_k^j, \tag{4.37}$$

as the association updates for the extent state. To obtain a final estimates α_k and $\mathbf{M}_k$ we use the mixture reduction technique from the mean $\mathbf{m}_k$ in (4.24). The degrees of freedom and filtering estimate then becomes

$$\alpha_k = \sum_{j=1}^{m_k} \sum_{i=1}^{\binom{m_k}{j}} \alpha_k^{i,j} \beta_k^{i,j} \tag{4.38}$$

$$\mathbf{M}_k = \sum_{j=1}^{m_k} \sum_{i=1}^{\binom{m_k}{j}} \mathbf{M}_k^{i,j} \beta_k^{i,j}. \tag{4.39}$$

The GPDA filter with random matrix is summarized in Algorithm 4.

```
input  : Data from target and clutter $\mathbf{Z}_{1:T}$, parameters $\mathbf{Q}$, $\mathbf{R}^{\mathrm{RM}}$, $\Delta t_k$, $\gamma_s$, $\tau$, $z$
output: Filtered mean $\mathbf{m}_k$, covariance $\mathbf{P}_k$, extent matrix $\mathbf{M}_k$ and variance $\mathbf{V}_k$ for
        $k = 1, ..., T$
Initialize estimates $\mathbf{m}_0$, $\mathbf{P}_0$, $\mathbf{M}_0$ and $\alpha_0$ ;
for $k = 1$ to $T$ do
    Compute predictions $\mathbf{m}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ ;
    Validate measurements by scaling the predicted ellipse with $\gamma_s$, and check if $\mathbf{z}_k^j$
      is inside it for all $j = 1, ..., m_k$ ;
    Compute $\mathbf{Y}_{k|k-1}$, $\mathbf{S}_{k|k-1}$ and $\mathbf{K}_{k|k-1}$ ;
    Calculate the number of association hypotheses $|\mathcal{E}|$ ;
    if $|\mathcal{E}| = 1$ then
        Perform regular random matrix filter as in Algorithm 2 ;
    else
        for $j = 1, ..., m_k$ do
            for $i = 1$ to $\binom{m_k}{j}$ do
                Calculate $\bar{\mathbf{z}}_k^{i,j}$ and $\bar{\mathbf{Z}}_k^{i,j}$ from $\mathbf{z}_k^{i,j} \in E_i^j$;
                Compute the association weights $\beta_k^{i,j}$ ;
                Compute the kinematic association values $\mathbf{m}_k^{i,j}$ and $\mathbf{P}_k^{i,j}$ ;
                Compute the association generated values $\hat{\mathbf{N}}_{k|k-1}^{i,j}$ and $\hat{\mathbf{Z}}_{k|k-1}^{i,j}$ ;
                Compute the extent association values $\alpha_k^{i,j}$ and $\mathbf{M}_k^{i,j}$ ;
            end
        end
        Compute $\mathbf{m}_k$ and $\mathbf{P}_k$ from $\beta_k^{i,j}$, $\mathbf{m}_k^{i,j}$ and $\mathbf{P}_k^{i,j}$ ;
        Compute $\mathbf{M}_k$ and $\alpha_k$ from $\beta_k^{i,j}$, $\alpha_k^{i,j}$ and $\mathbf{M}_k^{i,j}$ ;
    end
    Compute the variance $\mathbf{V}_k$ from $\mathbf{M}_k$ and $\alpha_k$ ;
end
```

Algorithm 4: Random matrix GPDA algorithm.

4.4 Implementation methods

The filtering methods presented above will be able to handle LIDAR data with clutter, but not without an initialization method. Also in the GPDA filters the computation of all association hypotheses in $\mathcal{E}$ will make the algorithms slow and unable to be applied to a real life tracking scenario where the filtering needs to happen fast. Hence we need to provide the filters with a robust initialization algorithm and a procedure to reduce the association hypothesis space.

When initializing the target ellipse we need an estimate of $\mathbf{m}_0$ in the C-GPDA method. In the random matrix method we will use the same kinematic vector and compute the extent estimate $\mathbf{M}_0 = \mathbf{R}_\phi(\phi_0)\,\mathrm{diag}[a_0^2, b_0^2]\mathbf{R}_\phi(\phi_0)^T$ where the heading ϕ_0 are given by the velocities in $\mathbf{m}_0$. To obtain this state vector estimate the measurements in $\mathbf{Z}_1$ and $\mathbf{Z}_2$ will be used, which means that in a real tracking scenario we need data from at least two consecutive time steps to initialize the target.

LIDAR data gives points along the surface of the objects that reflects the laser beams, and thus have a structure around the visible parts of the target. The clutter points breaks this structure and can be thought of as outliers in the data. If we are able to remove the clutter measurements from $\mathbf{Z}_1$ and $\mathbf{Z}_2$, it is possible to fit an ellipse to the target generated points that are left. This will give the positional coordinates (x_1, y_1) and the major and minor axes lengths a and b when using $\mathbf{Z}_1$ as the measurement set. When removing outliers from $\mathbf{Z}_2$ we can use the mean values $\bar{x}_2$ and $\bar{y}_2$ to calculate the velocities

$$v_{x,1} = \frac{\bar{x}_2 - \bar{x}_1}{\Delta t_k}, \quad v_{y,1} = \frac{\bar{y}_2 - \bar{y}_1}{\Delta t_k}, \tag{4.40}$$

where $\bar{x}_1$ and $\bar{y}_1$ are the mean values of the measurements in $\mathbf{Z}_1$ after removal of outliers. This is why we need at least two time steps of observations to initialize the ellipse, and with just $\mathbf{Z}_1$ it would be difficult to give an estimate of the velocities.

To remove clutter points from the measurement set $\mathbf{Z}_1$ we use the median absolute deviation (MAD) method, implemented in the function `isoutlier()` in MATLAB. This method computes the values

$$\mathrm{MAD}_x = \mathrm{median}(|\tilde{x}_1 - x_1^{med}|), \tag{4.41}$$

$$\mathrm{MAD}_y = \mathrm{median}(|\tilde{y}_1 - y_1^{med}|), \tag{4.42}$$

where x_1^{med} is the median of the x-coordinates in $\mathbf{Z}_1$, and correspondingly for y_1^{med}. The notation $\tilde{x}_1$ means a vector of all the x-values in $\mathbf{Z}_1$, and correspondingly for $\tilde{y}_1$. The method is removing points that are outside three scaled MADs from the median in both x- and y-coordinates. A scaled MAD is defined as $c \times$ MAD where the scaling constant is given by $c = -1/\sqrt{2}\mathrm{erfc}^{-1}(3/2)$. The function in the denominator is the inverse complementary error function which is defined as $\mathrm{erfc}^{-1}(1-\xi) = \mathrm{erf}^{-1}(\xi)$ for an arbitrary input ξ. After the outliers in x- and y-direction are detected, we keep the points that are not classified as outliers in both coordinates. This gives a higher chance of removing all clutter points, and it is important in the ellipse-fitting to not have extreme values that affects the initialization.

After the removal of outliers in $\mathbf{Z}_1$, hopefully we are left with only the target generated points, but there is no guarantee that there still are clutter measurements in the set. This will affect the ellipse-fitting method, but it is difficult to avoid this problem because there will always be some cases when outlier detection algorithms will struggle. To fit an ellipse to the measurements $\mathbf{Z}_1^v = \{(x_1^j, y_1^j)\}_{j=1}^{m_1^v}$ that are not removed, we will use the least-squares method from Fitzgibbon et al. [1999], which results in an eigenvalue problem which is easy to solve. This method will be presented in the following paragraphs.

The ellipse is defined as a general conic section given by

$$F(\mathbf{a}, \mathbf{c}) = \mathbf{a} \cdot \mathbf{c} = Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0, \tag{4.43}$$

where $\mathbf{a} = [A, B, C, D, E, F]^T$ and $\mathbf{c} = [x^2, xy, y^2, x, y, 1]^T$. We have a set of measurements $\mathbf{Z}_1^v$ and want to minimize the algebraic distance $F(\mathbf{a}, \mathbf{c}_1^j)$ for $j = 1, ..., m_1^v$ where $\mathbf{c}_1^j = [(x_1^j)^2, x_1^j y_1^j, (y_1^j)^2, x_1^j, y_1^j, 1]^T$. From this we get the optimization problem

$$\min_{\mathbf{a}} \sum_{j=1}^{m_k} F(\mathbf{a}, \mathbf{c}_k^j)^2, \tag{4.44}$$

which is a least-squares problem where the trivial solution is $\mathbf{a} = \mathbf{0}$. To get an ellipse solution it is necessary to introduce the constraint

$$4AC - B^2 = 1 \tag{4.45}$$

which can be expressed in matrix form as $\mathbf{a}^T \mathbf{C} \mathbf{a} = 1$ where the matrix $\mathbf{C}$ is given by

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{4.46}$$

Now the ellipse-fitting problem can be written as the minimization problem

$$\min_{\mathbf{a}} ||\mathbf{D}\mathbf{a}||^2, \quad \text{subject to} \quad \mathbf{a}^T \mathbf{C} \mathbf{a} = 1, \tag{4.47}$$

where the design matrix $\mathbf{D}$ is given by the coordinate vectors $\tilde{\mathbf{x}} = [x_1^1, ..., x_1^{m_1^v}]^T$ and $\tilde{\mathbf{y}} = [y_1^1, ..., y_1^{m_1^v}]^T$ as

$$\mathbf{D} = [\tilde{\mathbf{x}} \circ \tilde{\mathbf{x}}, \tilde{\mathbf{x}} \circ \tilde{\mathbf{y}}, \tilde{\mathbf{y}} \circ \tilde{\mathbf{y}}, \tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \mathbf{1}]. \tag{4.48}$$

Here we have used $\circ$ to denote the elementwise product of two vectors, also referred to as the Hadamard product. To solve the problem in (4.47) we introduce the Lagrange multiplier λ_L, and we get the objective function

$$\mathcal{L}(\mathbf{a}, \lambda_L) = \mathbf{a}^T \mathbf{D}^T \mathbf{D} \mathbf{a} - \lambda_L \mathbf{a}^T \mathbf{C} \mathbf{a}. \tag{4.49}$$

After differentiating this function we get the system of simultaneous equations

$$2\mathbf{D}^T\mathbf{D}\mathbf{a} - 2\lambda_L\mathbf{C}\mathbf{a} = 0, \tag{4.50}$$

$$\mathbf{a}^T\mathbf{C}\mathbf{a} = 1. \tag{4.51}$$

The equation (4.50) can be rewritten to

$$\mathbf{S}\mathbf{a} = \lambda_L\mathbf{C}\mathbf{a}, \tag{4.52}$$

where the scatter matrix is given by $\mathbf{S} = \mathbf{D}^T\mathbf{D}$. The system is solved by the generalized eigenvectors of the rewritten equation (4.52), but we need to find which eigenvector that solves (4.51). If the eigenvalue-eigenvector pair $(\lambda_i, \mathbf{u}_i)$ solves (4.52), then so does $(\lambda_i, \mu\mathbf{u}_i)$ for any μ. This allows us to construct a μ_i for each pair $(\lambda_i, \mu_i\mathbf{u}_i)$ such that $\mu_i^2\mathbf{u}_i^T\mathbf{C}\mathbf{u}_i = 1$, and we have that

$$\mu_i = \sqrt{\frac{1}{\mathbf{u}_i^T\mathbf{C}\mathbf{u}_i}} = \sqrt{\frac{1}{\mathbf{u}_i^T\mathbf{S}\mathbf{u}_i}}. \tag{4.53}$$

This gives the final solutions $\hat{\mathbf{a}}_i = \mu_i\mathbf{u}_i$ of (4.51) for $i = 1, ..., 6$. Each of the pairs $(\lambda_i, \mathbf{u}_i)$ represents local minima if the square root in (4.53) is real. The scatter matrix **S** is positive definite so that $\mathbf{u}_i^T\mathbf{S}\mathbf{u}_i$ is positive for all $\mathbf{u}_i$. Hence the generalized eigenvalues λ_i needs to be positive for the square root in (4.53) to be real. In Fitzgibbon et al. [1999] it is proved that the minimization problem in (4.47) has exactly one solution $\hat{\mathbf{a}}_i$ which is given by the eigenvector $\mathbf{u}_i$ that has eigenvalue $\lambda_i > 0$. From the solution $\hat{\mathbf{a}}_i$ we can find the ellipse values x, y, ϕ, a and b by using known formulas for the generalized ellipse parameters from (4.43).

Now we have found methods to remove clutter points from the initial measurement set $\mathbf{Z}_1$ and fit an ellipse to the remaining points. The full initialization is given in Algorithm 5.

```
input : Data from first two time steps $\mathbf{Z}_1$ and $\mathbf{Z}_2$, parameters $\Delta t_k$
output: Initial estimate $\mathbf{m}_0 = [x_0, y_0, v_{x,0}, v_{y,0}, a_0, b_0]^T$
Calculate the scaled median absolute deviation in both $x$- and $y$-direction ;
Remove outliers from $\mathbf{Z}_1$ that are outside three scaled MADs ;
Construct C-matrix, design matrix D and scatter matrix S from the validated
  measurements $\mathbf{Z}_1^v$ ;
Solve the eigensystem for S and C and choose the positive eigenvalue $\lambda_i$ and the
  corresponding eigenvector $\mathbf{u}_i$ ;
Use the solution $\hat{\mathbf{a}}_i = \mu_i\mathbf{u}_i$ to find $x_1$, $y_1$, $a_1$ and $b_1$ ;
Calculate $v_{x,1}$ and $v_{y,1}$ from $\mathbf{Z}_1$ and $\mathbf{Z}_2$ ;
```

Algorithm 5: Initialization algorithm.

Another important problem is to reduce the association hypothesis space $\mathcal{E}$ which has $2^{10} = 1024$ different hypotheses initially with $m_k = 10$. First we assume that we get at least one measurement from the target at each time step and that the target always exists. Hence we can remove E_0^0 from the hypothesis space $\mathcal{E}$. The measurement validation region Γ_k will reduce the number of measurements m_k in some time steps, but there will still be too many hypotheses to investigate in most of the time steps.

To further reduce $\mathcal{E}$ we need to remove some of the hypothesis spaces E^j for $j = 1, ..., m_k$ from $\mathcal{E}$. This means that we need to find which numbers of measurements that can be target generated, without excluding any hypotheses that can be the true association. We have assumed earlier in (4.9) that n_k^t follow a discrete uniform distribution between 1 and n^{max}. Now we introduce the parameter n^{min} which bounds n_k^t from below in this uniform distribution. Then we can write $n_k^t \sim U\{n^{min}, n^{max}\}$, and by finding a reasonable value for n^{min} in each time step, the hypothesis space size will shrink to a more tractable value.

There are many possible ways to find n^{min} from the measurement set $\mathbf{Z}_k$, but it is crucial that the method gives results lower than or equal to the true n_k^t so that the true association hypothesis is maintained in $\mathcal{E}$. A simple way to do this is to perform the outlier detection routine with MAD, and see how many points are left in $\mathbf{Z}_k$. Then we can set $n^{min} = |\mathbf{Z}_k^v| - g$, where $\mathbf{Z}_k^v$ is the measurement set after outlier removal and g is a subtraction parameter to be sure that the true n_k^t is not removed. This method is summarized in Algorithm 6, and will be used in the simulations and real data experiments. In the GPDA filter algorithms we will run the for-loop $j = n^{min}, ..., m_k$ instead.

```
input : Data from time step $\mathbf{Z}_k$, subtraction parameter $g$
output: Lower bound $n^{min}$
Calculate the scaled median absolute deviation in both $x$- and $y$-direction ;
Remove outliers from $\mathbf{Z}_k$ that are outside three scaled MADs ;
Return $n^{min} = |\mathbf{Z}_k| - g$ ;
```

Algorithm 6: Hypothesis reduction algorithm.

Chapter 5

Simulation results

In this chapter we give a description of the simulation studies that are done, and we present simulation results from experiments with CEKF and random matrix. The methods are first tested with only target measurements and filtering with Algorithm 1 and 2 from Chapter 3. Then the GPDA filter algorithms from Chapter 4 is applied on simulated LIDAR data both with and without clutter. To get a good comparison of the two methods we use the same parameters when simulating the single target process.

5.1 Simulation setup

The target ellipse in the CEKF simulation experiments is represented by the state vector $\mathbf{x}_k = [x_k, y_k, v_{x,k}, v_{y,k}, a_k, b_k]^T$ and the dynamic model function is the matrix $\mathbf{F}_{k-1}$ given in (3.19). The process noise covariance matrix $\mathbf{Q}$ is computed through discretization of a white noise acceleration model presented in Li and Vilkov [2003] using the method of Loan [1978] to evaluate the integral. From the experiments in Wilthil et al. [2017] we have that the acceleration noise σ_a should be between 0.05 and 0.5, and in the following experiments we use $\sigma_a = 0.1$. The process noise covariance matrix then becomes

$$\mathbf{Q} = \begin{bmatrix} 10^{-5}/3 & 0 & 5\cdot10^{-5} & 0 & 0 & 0 \\ 0 & 10^{-5}/3 & 0 & 5\cdot10^{-5} & 0 & 0 \\ 5\cdot10^{-5} & 0 & 10^{-3} & 0 & 0 & 0 \\ 0 & 5\cdot10^{-5} & 0 & 10^{-3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 10^{-4} & 0 \\ 0 & 0 & 0 & 0 & 0 & 10^{-4} \end{bmatrix}, \tag{5.1}$$

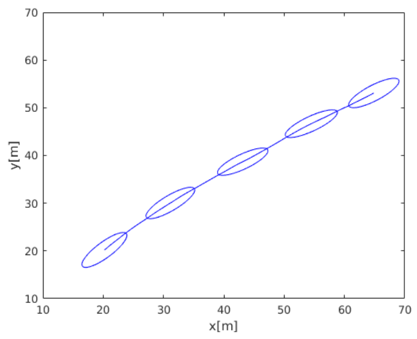
where the variances for a_k and b_k are set manually and added after the discretization. The other parameters in the CEKF simulation experiments are set as

$$T = 200, \quad \mathbf{R} = \text{diag}[0.1^2, 0.1^2], \quad \Delta t_k = 0.1, \quad \mathbf{x}_0 = [20, 20, 2, 2, 5, 1.5], \tag{5.2}$$

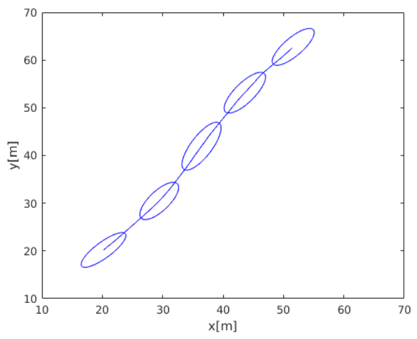
$$\mathbf{P}_0 = \mathbf{Q}. \tag{5.3}$$

These parameters are chosen when considering a medium-sized boat travelling in a straight line away from the LIDAR sensor in $(0, 0)$ which has a measurement standard deviation of 10 cm. The sampling time interval Δt_k is the same as for the LIDAR used in the real data experiments presented in the next chapter. This gives 20 seconds of total tracking time with 200 discrete time steps. When setting the initial estimate $\mathbf{m}_0$ we have the choice of providing it with the true target state $\mathbf{x}_0$, or use the initialization procedure from Algorithm 5. To explore the differences between these two cases, we will try both of them.

When simulating the target ellipse in the random matrix framework, the kinematic process for $\mathbf{x}_k = [x_k, y_k, v_{x,k}, v_{y,k}]^T$ becomes the same as above, but with a 4×4 dynamic model matrix $\mathbf{F}_{k-1}$ and $\mathbf{Q}$ without the entries for a_k and b_k. The extent process for $\mathbf{X}_k$ is simulated like in Example 3.2 by using the Wishart density in (3.49), with $\eta = 2000$ degrees of freedom and the initial extent matrix $\mathbf{X}_0 = \mathbf{R}_\phi(\phi_0)\,\mathrm{diag}[5^2, 1.5^2]\mathbf{R}_\phi(\phi_0)$. This will initially give the same process ellipse as in the CEKF, but not evolve in the same manner. However, when the degrees of freedom are so high, the extent matrix will not change much and be very similar to the CEKF simulated ellipse. This can be seen in Figure 5.1a and 5.1b where examples of processes for both methods are simulated with the parameter values above. Observe that the CEKF process moves faster in the x-direction than in the y-direction, and ends up with a higher x-coordinate than y-coordinate. It is opposite in the random matrix process, and this illustrates the randomness in the simulations. The processes are similar because of the low acceleration noise, but will not be identical. To test the methods on different target processes we run $N = 100$ Monte Carlo simulations of the tracking case with $T = 200$ time steps.



(a) CEKF process



(b) Random matrix process

Figure 5.1: Simulated processes for the CEKF and random matrix method plotted with a blue line for the position (x, y) and blue ellipse at time steps $k = 1, 50, 100, 150, 200$.

The number of target measurements n_k^t will either be fixed, or be drawn from a discrete uniform distribution $U\{n_{sim}^{min}, n_{sim}^{max}\}$ in the simulation experiments. The lower and upper limit of this distribution will not be provided to the GPDA filters, and they find the n^{min} parameter by using Algorithm 6 from section 4.4. The n^{max}-parameter from (4.9) will be set to a high value so that $m_k < n^{max}$ in all experiments. Then the hypothesis space

$\mathcal{E}$ is well defined, and the GPDA filters will use m_k as the maximum number of target measurements.

The target measurements $\Theta_k = \{\mathbf{z}_k^j\}_{j=1}^{n_k^t}$ are simulated in the same way as in (4.27) in Example 4.1, but the predicted measurements are along the contour of the process ellipse and not the predicted ellipse. After calculating each predicted point along the ellipse that are visible to the LIDAR in $(0, 0)$, the measurement noise $\mathbf{r}_k^j$ is added. In the simulation cases we use the tangent rotation of $\mathbf{R}_k^j$ given in (3.37). We could also have chosen the Jacobian rotation because we simulate and filter with the same rotation method, but since the real data problem in the next chapter gives tangential uncertainty we choose the tangent rotation. The measurements in the random matrix experiments are also sampled in this way because we want to test it on simulated LIDAR data.

The clutter measurements $K_k = \{\mathbf{z}_k^j\}_{j=n_k^t+1}^{m_k}$ are simulated from (4.10) and (4.11) with simulation Poisson parameter $\lambda_{sim} = 2/V_\Lambda = 1/800$ in the square region $\Lambda_k = [x_k - 20, x_k + 20] \times [y_k - 20, y_k + 20]$. This means that we expect 2 clutter points in the 40×40 square region around the target, which has the volume $V_\Lambda = 1600$. This region is set quite large so that the clutter measurements can occur far away from the target, and the Poisson intensity is low such that the number of total measurements stays low. It is more realistic to generate the clutter measurements independent of the validation region Γ_k because in a real life tracking scenario there will most likely be clutter outside the validation region. In the following experiments the scaling factor in (4.3) is set to $\gamma_s = 4$. The GPDA algorithms will expect 2 clutter points in the validation region Γ_k, because we set $\lambda = 2/V_k$ in the filter. When comparing the volume of Λ_k and Γ_k we see that the validation region volume V_k from (4.4) will be smaller than V_Λ, when the predicted values $a_{k|k-1}$ and $b_{k|k-1}$ are not much higher than 5 and 1.5 respectively. Hence the GPDA filters expect 2 clutter points in the validation region, which is smaller than the true clutter region. We set the parameter λ in this way, because the GPDA filters should expect that all simulated clutter points are within the validation region.

To measure the performance of the filtering methods it is necessary to specify an error measure, and the root mean squared error (RMSE) is a good measure for the actual error. This is given at time step k for each state variable by

$$\text{RMSE}_k^i = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(\mathbf{m}_k^n[i] - \mathbf{x}_k^n[i])^2}, \tag{5.4}$$

where $i = 1, ..., n_x$. The notation $\mathbf{m}_k^n$ means the filtered value at time step k in the n-th Monte Carlo run, and $\mathbf{x}_k^n$ is the corresponding true state vector with size $n_x = 6$. By looking at the RMSE for all the time steps we can determine if the filter estimates are close to the actual process values, but it does not include the estimated covariance matrix $\mathbf{P}_k$.

A common statistical test for filter consistency is given by the normalized estimation error square (NEES), defined in Bar-Shalom et al. [2001] as the scalar quantity

$$\text{NEES}_k = \sum_{n=1}^{N} (\mathbf{m}_k^n - \mathbf{x}_k^n)^T (\mathbf{P}_k^n)^{-1} (\mathbf{m}_k^n - \mathbf{x}_k^n), \tag{5.5}$$

where the variables are the same as in (5.4), and $\mathbf{P}_k^n$ is the filtered covariance matrix in Monte Carlo run n. We are interested in the NEES because it shows if the filter estimates $\mathbf{m}_k^n$ and $\mathbf{P}_k^n$ corresponds in the way they should. For instance if the absolute error given by the RMSE is low, and the covariance entries are high, the filter is not consistent.

The average NEES given in Salmond and Ristic [2004] can also be used, and the only difference is the scaling by $(n_x N)^{-1}$. The NEES in (5.5) is a $\chi^2_{n_x N}$-distributed variable, and thus a $(1-\alpha)$-confidence region is given by $[\chi^2_{n_x N, 1-\alpha/2}, \chi^2_{n_x N, \alpha/2}]$. With the values $n_x = 6$, $N = 100$ and significance level $\alpha = 0.05$ the interval is $[534.0186, 669.7692]$. This gives a measure of consistency and the NEES should not be outside this confidence region for more than 5% of the time steps. If the NEES values are above the upper confidence limit we say that the filter is too optimistic because the covariance matrix $\mathbf{P}_k^n$ contains low values, and the filter estimates should have higher estimated covariance. It can also happen when the absolute error is too high, or when both is the case. When the NEES is below the lower confidence limit we say that the filter is pessimistic, and it can happen when $\mathbf{P}_k^n$ is too high and/or when the absolute error is low.

A similar consistency measure, referred to as the separate NEES, is used for each component in the state vector, and is defined for the x_k-position as

$$\text{NEES}_{x_k} = \sum_{n=1}^{N} (\mathbf{m}_k^n[1] - \mathbf{x}_k^n[1])^T (\mathbf{P}_k^n[1,1])^{-1} (\mathbf{m}_k^n[1] - \mathbf{x}_k^n[1]), \tag{5.6}$$

where $\mathbf{m}_k^n[1]$ is the first element in the filtered mean vector $\mathbf{m}_k^n$ and $\mathbf{P}_k^j[1,1]$ is the filtered variance for the x-estimate. This is done for all the elements in the filtered vector $\mathbf{m}_k^n$. This error measure is distributed as a χ^2_N variable, and the confidence region is given by $[74.2219, 129.5612]$ for $N = 100$ and significance level $\alpha = 0.05$.

In the random matrix filtering the RMSE and NEES for the extent ellipse is given in Feldmann et al. [2011] as

$$\text{RMSE}_{\mathbf{X}_k} = \sqrt{\frac{1}{N} \sum_{j=1}^{N} \text{tr}((\mathbf{M}_k^j - \mathbf{X}_k^j)^2)} \tag{5.7}$$

$$\text{NEES}_{\mathbf{X}_k} = \sum_{j=1}^{N} \frac{\text{tr}[(\mathbf{M}_k^j - \mathbf{X}_k^j)^2]}{\text{tr}\,\mathbf{V}_k^j}, \tag{5.8}$$

where $\mathbf{V}_k$ is the filter estimate variance given in (3.57). The $\text{NEES}_{\mathbf{X}_k}$ measure will be a χ^2_N-distributed variable with the 95 % confidence interval $[74.2219, 129.5612]$. Also the kinematic NEES from (5.5) will now have 400 degrees of freedom rather than 600, which gives the confidence interval $[346.4818, 457.3055]$.

We construct simulation experiments by starting simple, and increasing the difficulty eventually. This means that the first experiments will be with only a fixed number of target generated measurements, and no clutter. The filtering methods presented in Chapter 3 will be tested on these data, and GPDA will not be used. Then the C-GPDA filter is used on the same data, to see how it performs without clutter. The next step is to introduce clutter measurements following the Poisson distribution in (4.10), and the C-GPDA and random matrix GPDA filters will be applied in these experiments. To further increase the difficulty we simulate the number of target generated points as a uniform between n^{min} and n^{max}. Finally, we introduce a non-homogeneous clutter case that reconstructs wakes behind the target. This clutter pattern is not assumed by the C-GPDA filter, but it will be useful to see how robust the method is.

In addition to the different simulation cases, we have two methods of initializing the filter. To make it easy for the filter to estimate consistent, we can provide the initial ground truth initially by setting $\mathbf{m}_0 = \mathbf{x}_0$. The other alternative is to use the initialization in Algorithm 5, which will be less accurate but it requires no other information than the two first measurement sets. We will use the ground truth initialization in all experiments, but also investigate how the automated method performs in two of the experiments.

5.2 Results with no clutter

In this section the filtering methods are tested on simulated data with $m_k = n_k^t = 10$ measurements from target only. Hence the filtering equations given in section 3.2 and 3.3 are used for GMEKF and random matrix respectively. In addition, the C-GPDA will be tested on the same measurements with the initialization in Algorithm 5 to see how the filter performs with no information about the true process. The method for finding n^{min} in Algorithm 6 is also applied with the subtraction parameter set as $g = 0$.

5.2.1 Contour EKF

When initializing with $\mathbf{m}_0 = \mathbf{x}_0$ it took 35.2 seconds to do all the simulations for the CEKF method. Plots of the RMSE for each individual state variable are shown in Figure 5.2 for each time step $k = 1, ..., 200$. We observe that the absolute errors are low and the estimates $\mathbf{m}_k$ are close to the ground truth $\mathbf{x}_k$ in all time steps, which is expected since the filter is provided with prior target information. Observe that the errors are a bit higher for v_y than v_x, which may be caused by the high velocity covariance entries in the **Q**-matrix given in (5.1). When the covariance is higher for a state variable, the simulated process values for this variable will be more noisy, and the filter will struggle more to estimate it correctly.

The a_k-estimates in Figure 5.2 have slightly higher errors than b_k, which is probably the lack of information from the measurements along the major axis of the ellipse. The target is moving away from the LIDAR, and it only captures the rear end which gives more information about the width than the length of the ellipse.

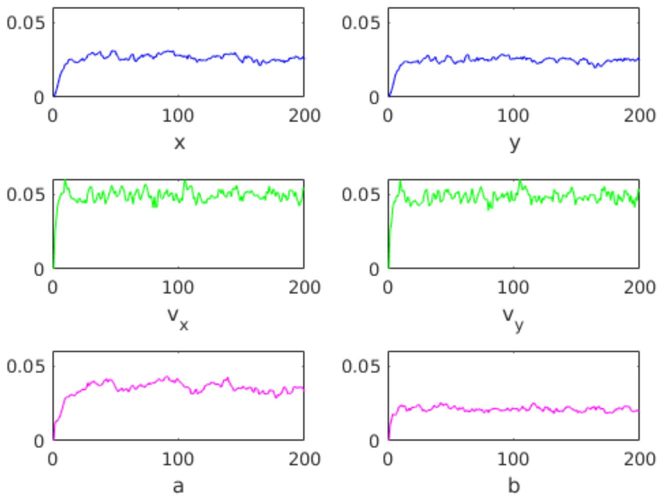
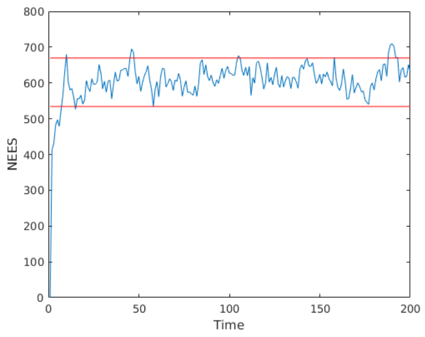


Figure 5.2: RMSE plot for each state variable for CEKF simulation with $m_k = 10$ and no clutter measurements.

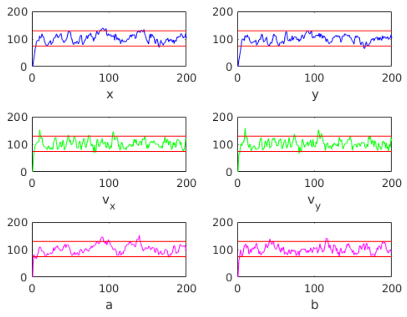
In Figure 5.3a the NEES from (5.5) is plotted for the CEKF together with the 95 % confidence limits, and the errors are within the confidence region 92.35% of the time. This is calculated without the initial 4 time steps it takes for the errors to stabilize, which we refer to as the burn-in period. This indicates that the errors are close to being consistent with the theoretical significance level $\alpha = 0.05$, which is acceptable. In Figure 5.3b the separate NEES given in (5.6) is plotted for all the state vector variables together with the 95 %-confidence limits for the χ^2_N-distribution. Here we can observe that all the estimates are consistent as expected from the total NEES.

When providing the filter with no information about the true target process, and running the same experiment with the C-GPDA filter in Algorithm 3, it takes 73.16 seconds to do the simulations. This increase in time is caused by the hypothesis investigation in the C-GPDA filter. The Poisson parameter was set to $\lambda = 1$ in the association weights $\beta_k^{i,j}$ in (4.18). We apply the initialization method in Algorithm 5, and calculate n^{min} in each time step. Thus we expect to get considerably higher errors because the filter don't know if the measurements are from target or clutter. We do this experiment to see how the C-GPDA filter will work when there are no clutter measurements. This case is not assumed in the derivation of the C-GPDA filter, and it is interesting to see how this model mismatch affects the filter.

The results are shown in the RMSE plots in Figure 5.4, and the errors are way higher than in the previous experiment shown in Figure 5.2. This is mainly caused by the ini-



(a) Average NEES



(b) Separate NEES

Figure 5.3: NEES plots for CEKF experiment with $m_k = 10$ and no clutter measurements.

tialization process which estimates the initial estimates inaccurately. All state variables deviates from the true state $\mathbf{x}_1$ which will be close to the initial state in (5.2). The positional errors are high in the beginning and then follow a decreasing trend throughout the tracking process. In the velocity estimation the errors are lower in the beginning, and also decreases afterwards. The a_k-estimates have high initial error, but they also get smaller eventually. This is because only the rear end is visible in the beginning of each track, and after a few time steps more of the full contour will be detected in some of the runs. The minor axis b_k will be visible for most of the time, and the errors are low throughout the tracking process.

The C-GPDA estimates are not consistent when the filter gets no information about the true process, if we use the process parameter $\mathbf{Q}$ and measurement covariance $\mathbf{R}$ in the filter. This is because the absolute errors are high, and the entries of these covariance matrices are low. As a result of this, the filtered covariance matrix $\mathbf{P}_k$ will contain values that are too small and the NEES in (5.5) will get high. However, if we create some new tuning parameters σ_a^{tun}, $\Sigma_{a,b}^{tun}$ for the kinematic and extent noise respectively, we can create a new process noise covariance matrix $\mathbf{Q}^{tun}$, and set the initial filter covariance as $\mathbf{P}_0 = \mathbf{Q}^{tun}$. In addition we can use this in the prediction and filtering equations with the tuned measurement noise $\mathbf{R}^{tun}$ as well. This will be similar to a filter tuning when tracking a target from real data without knowing the ground truth.

To find $\mathbf{Q}^{tun}$ we construct a new 4×4 kinematic noise matrix $\mathbf{Q}_{kin}^{tun}$ by using the same method as for $\mathbf{Q}$ described above, but with the tuning acceleration noise σ_a^{tun} instead. Adding the diagonal matrix $\Sigma_{a,b}^{tun}$ gives the 6×6 tuning covariance matrix $\mathbf{Q}^{tun}$ that are used in the filter. Then it is possible to get more consistent results, after some trial and error we found that with the acceleration noise $\sigma_a^{tun} = 2.9$, extent noise $\Sigma_{a,b}^{tun} = \text{diag}[0.5^2, 0.1^2]$ and $\mathbf{R}^{tun} = \text{diag}[1.4^2, 1.3^2]$ improved the NEES values to some degree.

The results are shown in Figure 5.5a to not be consistent in the first 100 time steps, but

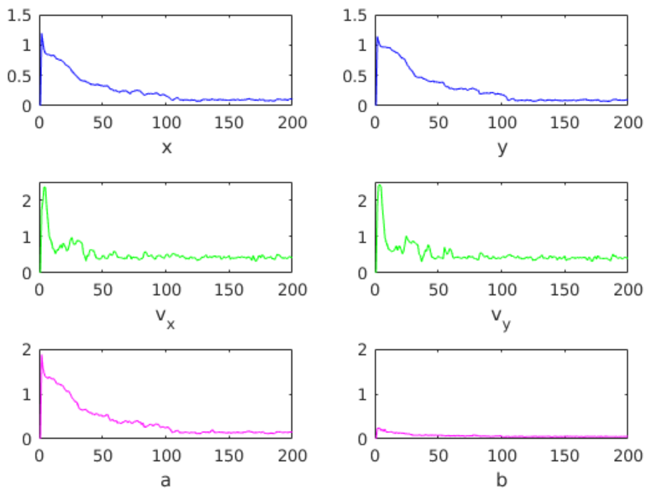


Figure 5.4: RMSE plots from estimation with the C-GPDA filter when no information about the true process $\mathbf{x}_k$ is given. The filter is only provided with $m_k = 10$ target generated measurements at each time step.

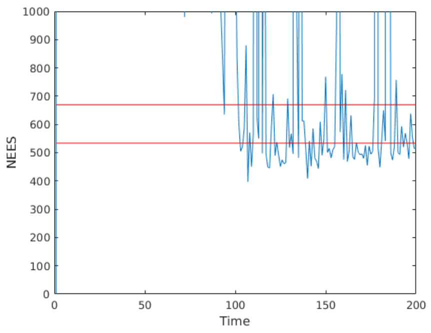
then it gets inside the confidence interval more often the last 100 steps. In total the NEES is outside the confidence interval 86.22 % of the time, which is a long time, but it shows that the C-GPDA filter can give consistent results when no information about the target is given, with some tuning of the filter. The separate plots in Figure 5.5b show that the positional estimates are inside the confidence interval the last 100 time steps, while velocity estimates actually are pessimistic and lies beneath the lower confidence limit. This show that the velocity variance entries in $\mathbf{P}_k$ are a bit too high.

5.2.2 Random matrix

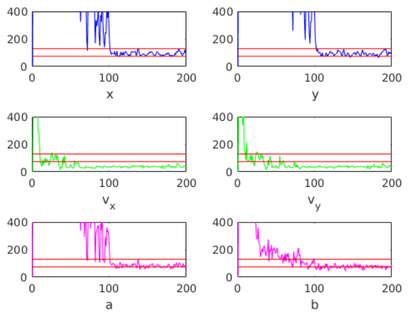
In the random matrix simulation, all the parameters are given above, and we start by initializing the filter with $\mathbf{m}_0 = \mathbf{x}_0$ and $\mathbf{M}_0 = \mathbf{X}_0$. We have three tuning parameters that can be adjusted to give better results, and after looking at the RMSE and NEES plots we get that the optimal parameters are

$$\mathbf{R}^{\mathrm{RM}} = \mathrm{diag}[0.5^2, 0.5^2], \quad z = 1/2, \quad \tau = 0.01. \tag{5.9}$$

It took 23.45 seconds to run all the simulations, which is considerably faster than the CEKF simulations. This is probably caused by the calculations of the predicted measurements $\mathbf{y}_k^j$, the numerical Jacobian matrix $\mathbf{H}_k$ and the covariance matrices $\mathbf{R}_k^j$ in the CEKF method.



(a) Average NEES



(b) Separate NEES

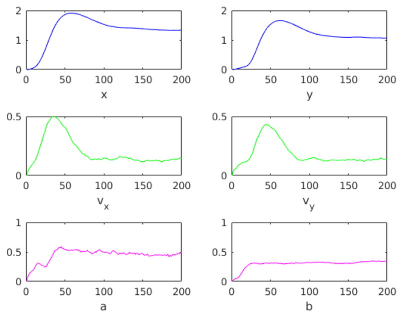
Figure 5.5: NEES plots when C-GPDA have no information about the true process, just the $m_k = 10$ target generated measurements at each time step.

The kinematic RMSE results for the random matrix filter are plotted in Figure 5.6a, and show that the position and velocity estimates have relatively high errors considering the low process and measurement noise in the experiment. The errors are much higher than for the CEKF in Figure 5.2, and show that the random matrix method has some weaknesses in the kinematic estimation. The centroid position calculated in (3.61) is probably the main reason why the positional estimates deviates from the ground truth.

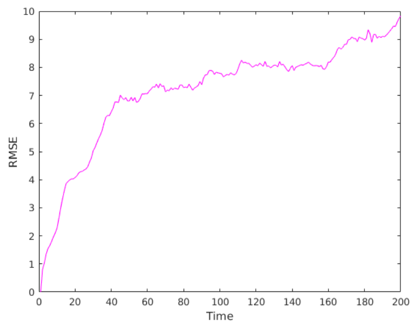
Also in Figure 5.6a we have added the a_k and b_k error plots for comparison with the CEKF results. These quantities are found from the eigenvalues of the filtered extent matrix $\mathbf{M}_k$. The RMSE is computed from (5.4) with the ground truth values as the eigenvalues of $\mathbf{X}_k$. We observe that the absolute errors are considerably higher than in the CEKF experiment. In Figure 5.6b the extent RMSE from (5.7) is plotted, and it confirms that the random matrix method struggles to estimate the extent correctly. In the first time steps it is rising rapidly, but eventually it stabilizes more. These plots show that the random matrix is not preferable when considering absolute error compared to the CEKF.

The kinematic NEES results from the random matrix method are shown in Figure 5.7a for the average quantity and the separate measures are plotted in Figure 5.7b. From the plots we observe that the filter is not consistent for most of the time, but for the last 100 time steps the velocity estimates are inside the confidence interval. This indicates that the kinematic estimation can be consistent in some cases, and this makes sense because the filtering process is similar to the regular Kalman filter. However, the positional estimation is not consistent at all, and shows that the centroid calculation in (3.61) gives inaccurate results.

In the extent estimation the random matrix estimate $\mathbf{M}_k$ is actually giving consistent results in most of the time steps as shown in the NEES plot in Figure 5.8. It is outside the confidence interval in 8.96 % of the time steps, which is almost theoretically consistent with the 95 % confidence interval. Comparing it to the extent RMSE in Figure 5.6b we

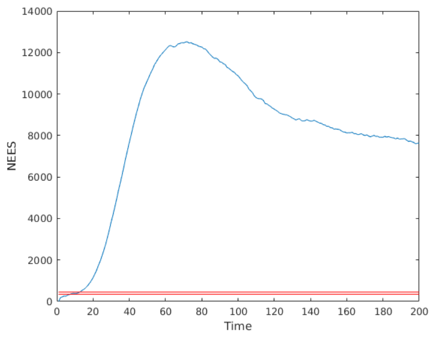


(a) RMSE kinematics

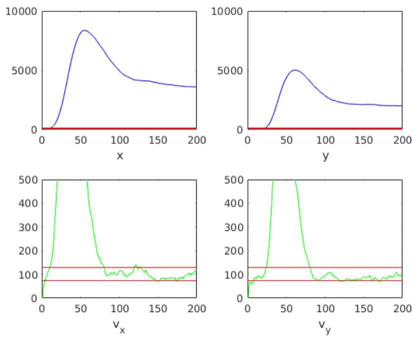


(b) RMSE extent

Figure 5.6: RMSE plots of kinematics and extent estimates from the random matrix filter with $m_k = 10$ and no clutter measurements.



(a) Average NEES



(b) Separate NEES

Figure 5.7: NEES plots for the kinematic estimates $\mathbf{m}_k$ in the random matrix filter with $m_k = 10$ and no clutter measurements.

see that the filter is not consistent when the RMSE is increasing in the beginning, but when the RMSE stabilizes the NEES is inside the interval. Hence the variance $\mathbf{V}_k$ also stabilizes because the NEES in (5.8) and the RMSE (5.7) consists of the same quantity $\sum_{n=1}^{N} \text{tr}[(\mathbf{M}_k^n - \mathbf{X}_k^n)^2]$. If the RMSE increases and the variance $\mathbf{V}_k$ increases at the same rate, the NEES will remain constant.

By tuning the filter parameters and looking at the RMSE and NEES plots, we have found that a higher value of the parameter τ gives less consistent extent estimates. This parameter has an impact on the evolution of the predicted degrees of freedom $\alpha_{k|k-1}$ in (3.55) and a higher τ gives a higher $\alpha_{k|k-1}$ each time step. In the filtering step this will make α_k in (3.73) higher, and thus the filtering variance in (3.57) lower. A lower variance will increase the NEES value if the RMSE stays the same or increases. Hence the τ-parameter can be tuned such that the NEES lies more inside the 95 %-confidence interval, and this is why we have set $\tau = 0.21$.

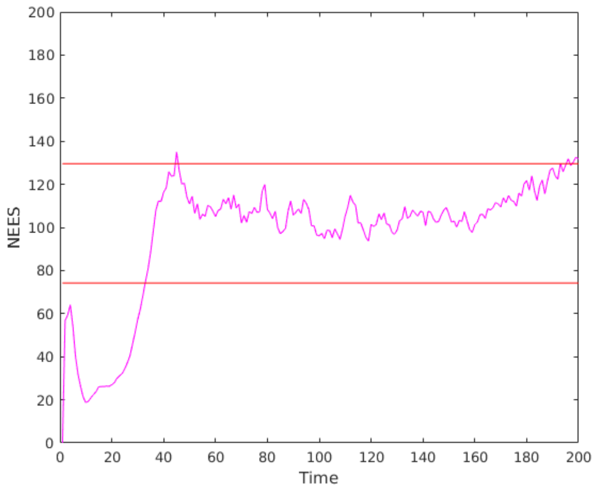


Figure 5.8: NEES plot for the extent estimate $\mathbf{M}_k$ in the random matrix filter with $m_k = 10$ and no clutter measurements.

When considering the results above for the random matrix method, especially the high absolute errors, we will not get any better results when using the initialization method in Algorithm 5, and there will be no point in testing the random matrix GPDA filter like we did with the C-GPDA.

5.3 Results with homogeneous clutter

In this section we simulate a single target where the measurement vector contains clutter measurements from a homogeneous Poisson point process, and use the elliptical validation region Γ_k from section 4.1. The GPDA filters will be used in all experiments, and they will use the same λ-parameter as the simulated clutter measurements in the association weights. We study two different cases with increasing difficulty when it comes to find the correct association hypothesis. First we let the clutter cardinality be Poisson distributed, while keeping $n_k^t = 10$ fixed. Secondly we let both target and clutter measurements have random cardinalities and the target measurements follow a discrete uniform distribution. We initialize the target estimate by using the ground truth in all experiments, except for the last case. The method for finding n^{min} will be used in all experiments with subtraction parameter $g = 3$.

5.3.1 10 target measurements, random clutter measurements

The first step is to let the clutter measurements be random according to a Poisson distribution given in (4.10) and uniform spatial distribution from (4.11). We set the target measurements to $n_k^t = 10$, and sample the clutter points with $\lambda = 2/V_\Lambda = 1/800$ and spatial window Λ_k given in section 5.1. The filters do not know that there are 10 target measurements each time step, and it will search through all the possible hypotheses E_i^j from $j = n^{min}$ to $j = m_k$. The other parameters have the same values as before.

The results from the 100 Monte Carlo runs are summarized in Table 5.1 for both C-GPDA and random matrix GPDA. The high running time difference is probably caused by the calculation of the scatter matrix $\bar{\mathbf{Z}}_k^{i,j}$ in the random matrix GPDA. We observe that the C-GPDA method has low absolute errors around the same values as in Figure 5.2. This shows that even though clutter is added, the C-GPDA filter is able to find the correct association hypothesis or a good mixture at each time step. It is also consistent in over 95 % of the time, which is slightly better than the no clutter case. This is probably caused by the randomness in the simulations, and if we tried to change the parameter settings a bit, the results would get different. Observe that in both cases the n^{min} is found below or equal to the true n_k^t in all time steps, and does not exclude the correct hypothesis.

Measure	C-GPDA	Random matrix GPDA
Time [s]	691.8	14246
Mean x RMSE	0.0271	9.1145
Mean y RMSE	0.0256	3.3116
Mean v_x RMSE	0.0508	0.9654
Mean v_y RMSE	0.05	0.5965
Mean a RMSE	0.0358	0.8301
Mean b RMSE	0.0221	0.9717
Mean NEES	607.4672	36458, 1321
% confidence	0.9541	0.0051, 0
% where $n^{min} \leq n_k^t$	100	100

Table 5.1: Results from GPDA estimation in experiments with $n_k^t = 10$ and n_k^c from a Poisson distribution. Random matrix has two entries in the NEES and confidence cell for kinematic and extent NEES. The last row tells if the n^{min} parameter is found such that it keeps the correct hypothesis in each time step.

From the previous no clutter simulation for random matrix we have the values $\bar{\text{RMSE}} = [1.3455, 1.0775, 0.1995, 0.1805, 0.4499, 0.2986]$, NEES kinematics mean 8255.7, and extent mean 97.4003. By looking at the values in Table 5.1 we see that all these values have increased, which is what we would expect when adding clutter. The random matrix method did struggle in section 5.2 with only target generated measurements, so there was no reason to believe that it would do better with clutter added. Hence we will focus on the C-GPDA method in the following experiments, and revisit the random matrix method in the real LIDAR data chapter.

5.3.2 Random target measurements, random clutter measurements

Now we extend the experiment to give a random number of target measurements, and we set that $n_k^t \sim U\{7, 11\}$ when simulating. This discrete uniform distribution ensures that there is a limit to how many target generated points we can have, such that the association hypothesis space $\mathcal{E}$ stays small. At the same time we need a sufficiently number of target points to be able to get an accurate estimate of the ellipse, and this is why we set 7 points minimum. We will only test the C-GPDA algorithm in this subsection. The clutter measurements are simulated in the same way as before.

We run experiments with both initialization methods of the target ellipse to test the difference in a cluttered environment. We refer to the experiment with initialization from Algorithm 5 as the automated C-GPDA filter, and we used the tuning parameters $\sigma_a^{tun} = 3.5$, $\Sigma_{a,b}^{tun} = \text{diag}[1.5^2, 0.5^2]$ and $\mathbf{R}^{tun} = \text{diag}[1.5^2, 1.4^2]$ in this experiment. The results are shown in Table 5.2, which shows that the automated filter has considerably higher absolute errors. The different running times is probably caused by the validation regions in each experiments, which depends on the predicted state vector. When fewer measurements are validated because of a small predicted ellipse, the running time will go down. This is what is happening in the automated C-GPDA, and by looking at track plots we see that the extent estimates generally are lower than the ground truth. The x, y and a estimates have approximately 100 times higher errors in the automated filter than in the ordinary C-GPDA. This is probably caused by the initialization process and the fact that the full length is not visible initially.

Measure	C-GPDA	C-GPDA automated
Time [s]	558.3	472.3
Mean x RMSE	0.0283	2.4927
Mean y RMSE	0.0265	2.5254
Mean v_x RMSE	0.0511	0.3858
Mean v_y RMSE	0.0501	0.4572
Mean a RMSE	0.037	3.5664
Mean b RMSE	0.0229	0.8515
Mean NEES	602.3924	$2.4298 \cdot 10^6$
$\%$ confidence	88.78	0
$n^{min} \leq n_k^t$	100	100

Table 5.2: Results from estimation with C-GPDA filter in experiments with $n_k^t \sim U\{7, 11\}$ and n_k^c from a Poisson distribution.

The C-GPDA continues to have low RMSE when initializing with $\mathbf{x}_0$, regardless of the random n_k^t. The consistency is a bit lower than in the previous experiment with 88.78 $\%$ inside the confidence interval. However, the filter proves to be consistent most of the time, which shows that the C-GPDA filter still works fine when increasing the difficulty in the experiments.

5.4 Results with non-homogenous clutter

In this section we want to test the C-GPDA filter on a more difficult case with structured clutter, without changing the algorithm. By doing this we will see how robust the C-GPDA method is to a clutter process that is not modelled in the filter. We want to simulate measurements behind the target ellipse that are reconstructions of wake detections, which is a case that is most probably going to happen in a real tracking scenario. In addition we keep the existing target and clutter measurement case, but with $n_k^t \sim U\{7, 9\}$ to limit the size of the hypothesis space.

The number of wake measurements n_{wake} are simulated by drawing values from a Poisson distribution with parameter $\lambda_{wake} = 1$. Then we draw n_{wake} points from an exponential distribution with parameter $\mu_{wake} = 1$ behind the rear end of the target as shown in Figure 5.9. This means that the rear point of the target ellipse is the origin of the coordinate system for the exponential distribution, and the positive direction is opposite of the target heading. To distribute the points to each side we use a normal distribution with mean in the position given by the exponential distribution, and variance $\sigma_{wake}^2 = 1$ which is illustrated in Figure 5.9. This distributes the points perpendicular to the opposite heading direction.

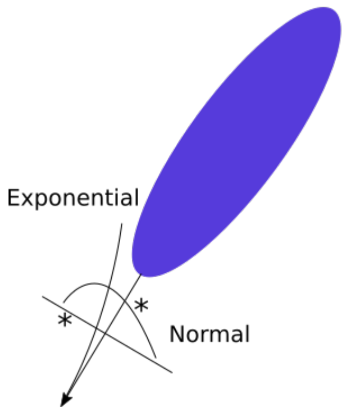


Figure 5.9: Illustration of how the wake measurements are simulated with the target ellipse moving upwards to the right. The exponential distribution is starting in the rear end of the target with direction opposite of the target heading, and the normal distribution is perpendicular to this direction. The stars * are simulated wake clutter points.

The results from $N = 100$ Monte Carlo simulations are shown in Figure 5.10 for the RMSE. The simulations took 1130 seconds to run, which is probably caused by the high number of points generated in some time steps. We observe that all absolute errors are small like in the previous experiments, which shows a robustness in the C-GPDA filter. Even though we have added more complex clutter, the method is still giving RMSE plots like in the first experiments with no clutter.

The NEES plot in Figure 5.11a show that the C-GPDA filter is not consistent in almost

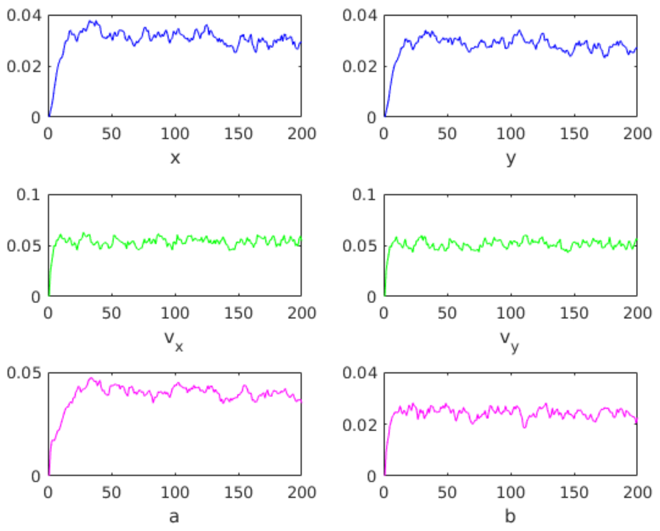
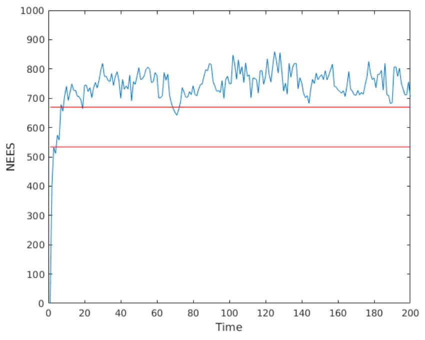
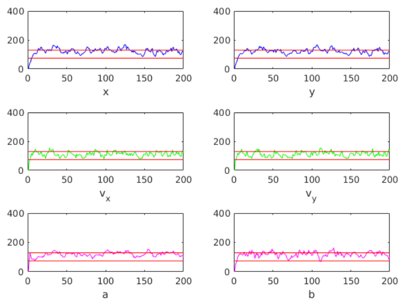


Figure 5.10: RMSE plots for C-GPDA filter with non-homogeneous clutter measurements simulating a wake.

all time steps. However, in Figure 5.11b we observe that the separate measures are more consistent than the average measure. This is probably caused by the off-diagonal covariances in $\mathbf{P}_k$, that are too low. We also see that the positional and major axis estimates have NEES above the confidence interval in several time steps. This means that the filter is too optimistic in the estimation, and should have given higher covariances in $\mathbf{P}_k$.



(a) Average NEES



(b) Separate NEES

Figure 5.11: NEES plots for C-GPDA with non-homogeneous clutter measurements simulating a wake.

In this simulation case we potentially can have many clutter points compared to target generated points, and it is interesting to see if the C-GPDA filter chooses hypotheses that have a high j. The prior density in (4.13) suggests that the C-GPDA filter will prefer a high j, and we want to study if the filter actually does this. In Figure 5.12 we have plotted the calculated mean n^{min} parameter over the Monte Carlo runs for each time step $k = 1, ..., 200$, which we used $g = 6$ to find. This is why this value is lying around 2, and can explain the high running time of the algorithm because the hypothesis space is not narrowed down that much. The true target generated points n_k^t are plotted in blue, with the filtered hypothesis j with highest association weight $\beta_k^{i,j}$ in red. This shows that the filter chooses the correct hypothesis space E^j in most of the time steps, and is not choosing hypotheses that have a higher j than the ground truth.

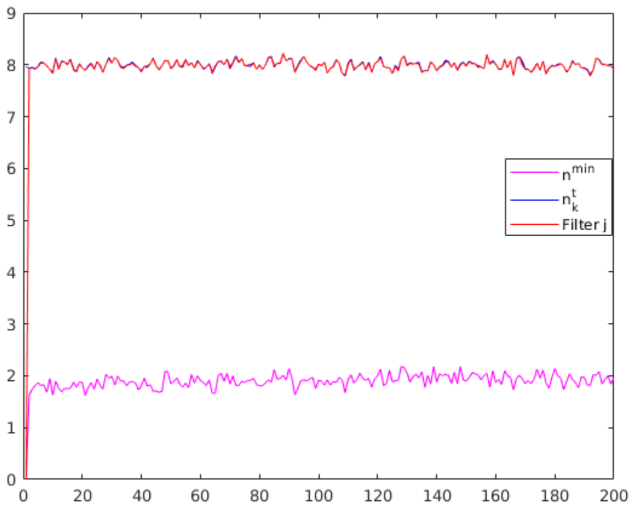


Figure 5.12: Mean n_k^t (blue) for true process plotted with mean n^{min} and mean j (red) for the hypothesis E_i^j with maximum $\beta_k^{i,j}$ in the C-GPDA filter for each time step k. The means are taken over $N = 100$ Monte Carlo runs.

Chapter 6

Real LIDAR data results

In this chapter we will first present the LIDAR dataset of the Munkholm boat together with techniques that are used to prepare the data for filtering. The GPDA filters are tested on the resulting dataset and evaluated by looking at tracking plots and the normalized innovation squared (NIS). We only have the ground truth of the target extent in this chapter, and can look at RMSE or NEES for a_k and b_k, but not the other state variables.

6.1 Data processing

The LIDAR data is taken from the departing Munkholm boat, which is docking next to the pier at Ravnkloa shown in Figure 6.1. The LIDAR is mounted in the outmost part of the pier, giving it clear sight over the whole canal where the Munkholm boat is departing.

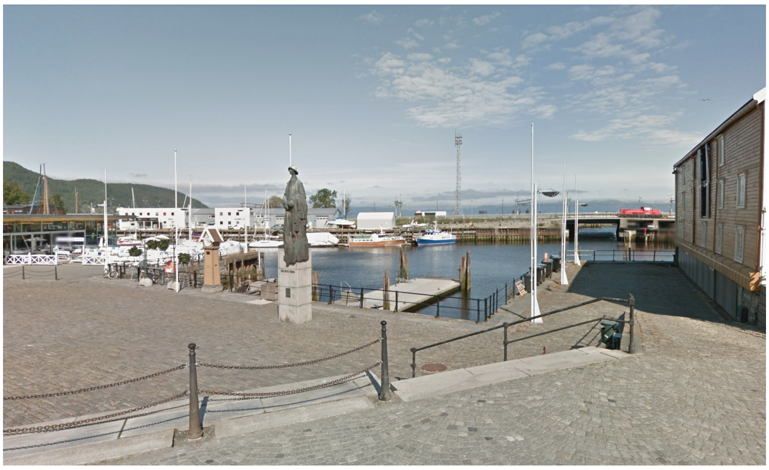


Figure 6.1: Picture from Ravnkloa, Trondheim taken by Google Maps.

In Figure 6.2 the arriving Munkholm boat is depicted and from the picture we see that it has an elliptical shape of in the front, but more squared in the rear end. The true length and width given by Eide [2016] is 19.9 and 4.2 metres respectively, which gives the true major and minor axes as $a_k = 9.95$ and $b_k = 2.1$.

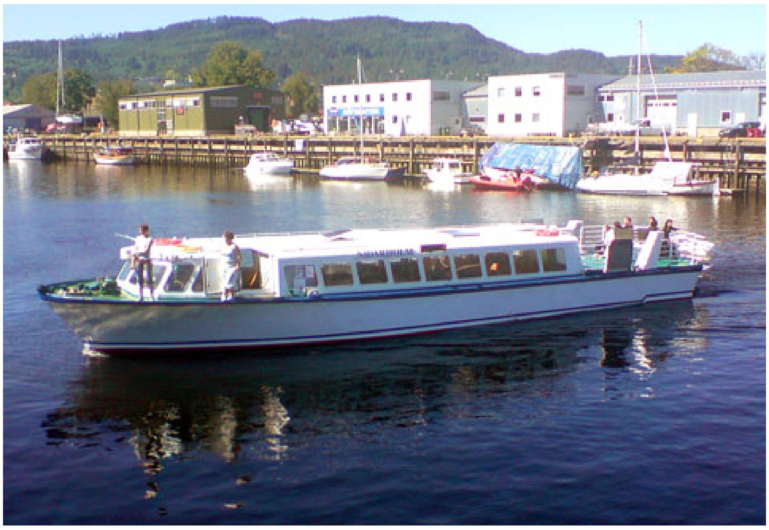


Figure 6.2: Picture of the Munkholm boat MS Nidarholm in Ravnkloa, Trondheim.

The LIDAR that were used to get the data was a Velodyne VLP-16, which is a real-time 3D LIDAR with a range of 100 metres. It can detect 300.000 points per second, and gives a 360 degree horizontal and ± 15 degree vertical view of the surroundings. The frequency used to fetch data was 10 Hz, so the time sampling interval parameter is set to $\Delta t_k = 0.1$ in all experiments. In the dataset we have in total $T = 1089$ time steps, i.e. 108.9 seconds in real time, of the Munkholm boat departing from the pier behind the statue in Figure 6.1. The resulting 3D pointcloud at timestep $k = 257$ is shown in Figure 6.3, where the LIDAR is positioned at the pier in $(0, 0)$, and the Munkholm boat in $(10, -10)$ has just departed. Here we observe how the LIDAR reconstructs the surroundings in the harbour by detecting points for instance along the yellow house wall from $(-20, 0)$ to $(-20, -40)$ in West-South coordinates.

The presented tracking methods are assuming that the data is two-dimensional, and not in 3D as shown in Figure 6.3. Hence we only use the West (x) and South (y) coordinates from now on, which gives a lot of points on top of each other in the resulting 2D dataset. This is because a lot of the points have the same West and South coordinates, but different Height coordinates. The resulting 2D data are plotted in Figure 6.4 in four different time steps. The first step $k = 90$ shown in blue colors is capturing the Munkholm boat starting to back up out of the pier while doing a small turning maneuver such that it can turn around and leave as shown in red, magenta and cyan. It is this track we want to capture by using the C-GPDA and random matrix GPDA algorithms.

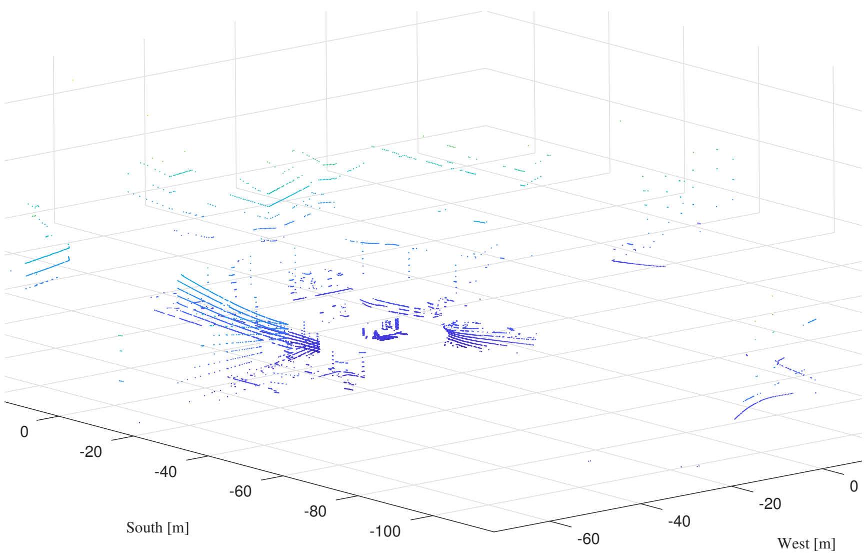


Figure 6.3: 3D pointcloud at time step $k = 257$ of LIDAR detections from Ravnkloa, with the LIDAR in $(0, 0)$ and the Munkholm boat positioned around $(10, -10)$ West-South coordinates.

There are a lot of data points that are stationary in the sense that they do not change position, for instance the pier or the yellow house wall are examples of objects that generate these measurements. We want to reduce the dataset such that it contains mostly detections from the Munkholm boat. This is done by making a square ”cut” in the data and only using the measurements that lie inside the region where $x \in [-16, 30]$ and $y \in [-2, -55]$. By doing this cut we can see from Figure 6.4 that most of the stationary points are removed, but the detections in the early time steps like in $k = 90$ are also removed. So in the final dataset the starting time is $k = 250$ and there are $T = 500$ time steps such that it ends at $k = 750$. From now on we only refer to this dataset and thus $k = 1$ is at time step 250 in the original dataset.

The mean number of measurements at each time step in the dataset is 364, and to investigate the resulting $2^{364} - 1$ association hypotheses in the GPDA filter is an intractable problem. Hence we need to narrow down the number of hypotheses and this is done by clustering the measurement points together. By using the cluster centroids instead of the original measurements, much of the information is maintained and the number of association hypotheses is reduced. The K-means clustering algorithm is used because it is efficient and let us control how many cluster centroids we want to make. The algorithm works by iterating between two steps: assigning each data point to its nearest cluster by calculating the Euclidean distance to each centroid position, and computing the cen-

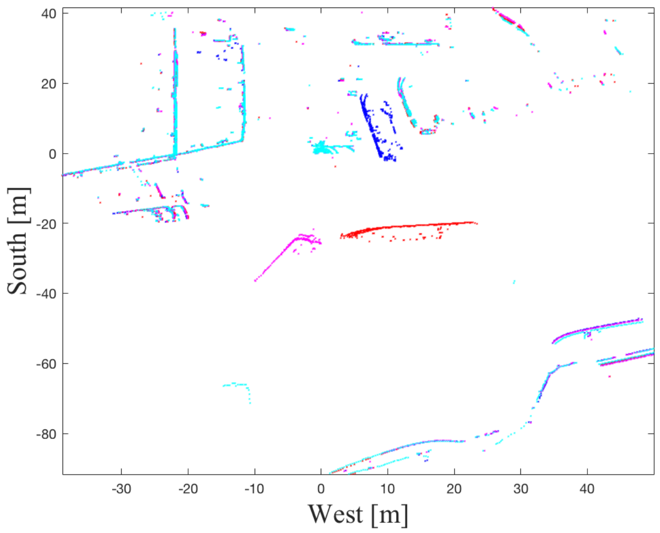


Figure 6.4: 2D plot of LIDAR detections when the Munkholm boat departs at four time steps: $k = 90$ (blue), $k = 490$ (red), $k = 610$ (magenta) and $k = 800$ (cyan). The LIDAR sensor is located at the pier in $(0, 0)$.

troids by taking the mean position of all the data points assigned to that centroid. By running this algorithm on the measurements in $\mathbf{Z}_k$, we get the set of centroid positions $\mathbf{C}_k = \{(x_j, y_j)\}_{j=1}^{K}$, where K is the number of clusters in each time step. This set will be used as the measurement set in the implementations on the LIDAR dataset.

The parameter K is convenient in the sense that it determines the number of total measurements m_k. Hence it gives an upper limit to the number of association hypotheses to investigate. In the following experiments it is set to $K = m_k = 10$, which is high enough to capture most of the information in the original data, but not so high that we get an infeasible number of association hypotheses. In Figure 6.5 the data are plotted for three different time steps, together with the 10 cluster centroids from K-means clustering. Most of the centroids are located along the boats edge, but observe that some are positioned around detections from the center of the boat. In time step $k = 203$ there is clearly one centroid that is not target generated, and should be removed by the validation procedure. Also at $k = 404$ there are two centroids located behind the rest of the measurements, and are most likely generated from the wake behind the boat. To remove cluster centroids located far away from the target, the validation region Γ_k represented by the state vector in (4.3) will be applied to the filters. The scale parameter is set to $\gamma_s = 4$ in all experiments, and the clutter Poisson parameter is $\lambda = 1/V_k$. This is based on the plot in Figure 6.5, where most of the measurements come from target. When finding the lower bound n^{min} we use the method from section 4.4 with parameter $g = 1$.

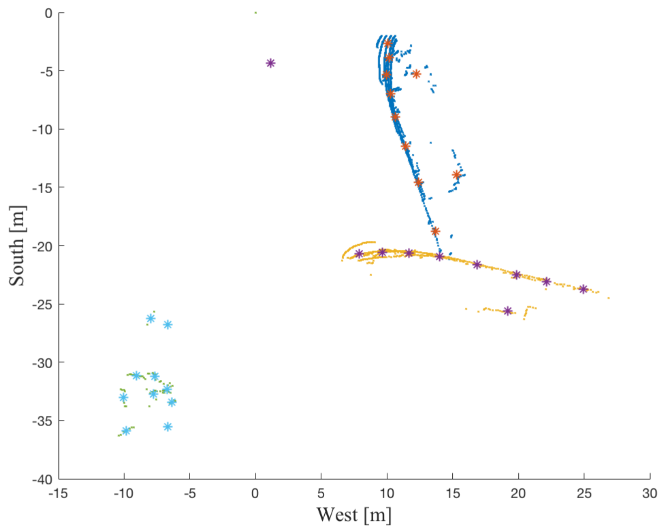


Figure 6.5: Plot of LIDAR measurements from the Munkholm boat and $K = 10$ cluster centroids calculated with K-means clustering. Measurements are from $k = 2$ (blue, red), 203 (beige, purple) and 404 (green, cyan).

We need to provide the filters with the initial filtering estimate $\mathbf{m}_0$ and we use the procedure in Algorithm 5 to find it. However, by using the original measurements $\mathbf{Z}_k$ instead of the clusters $\mathbf{C}_k$ when computing the velocities $v_{x,0}$ and $v_{y,0}$, we obtain estimates that are more realistic than when using the cluster points. The result is shown in Figure 6.6, and we observe that the ellipse is fitted accurately through the points on the visible side of the target. The two clusters that are generated from measurements in the center of the boat are also close to the ellipse contour, and makes the estimated minor axis b_0 larger than it would have been without these points. This is because the least squares ellipse fitting method is biased towards smaller ellipses. We observe that the extent variables are relatively close to the true extent $a = 9.95$ and $b = 2.1$.

6.2 C-GPDA results

Applying the C-GPDA method on the centroid data points $\mathbf{C}_k$ we get the track shown in Figure 6.7, and the whole experiment with 500 time steps took 6.1 seconds to run in MATLAB. This fast running time may come from the n^{min} found at each time step, which has mean value around 8 over all time steps. This gives 56 association hypotheses on average to investigate, which is much lower than the 1023 hypotheses we would have if we used $n^{min} = 1$. This saves a lot of running time for the C-GPDA algorithm, and makes it more efficient.

Initially at $k = 2$ the shape and heading of the ellipse is close to the estimate $\mathbf{m}_0$ from

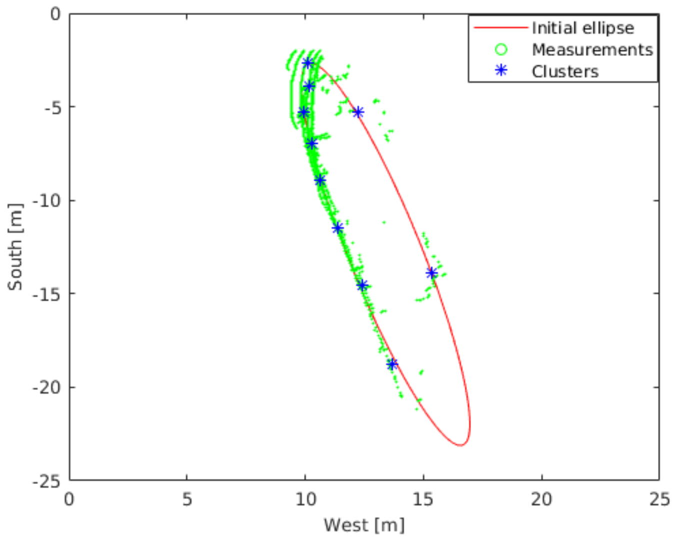


Figure 6.6: Initialized target ellipse $\mathbf{m}_0 = [13.4, -12.89, 0.33, -1.04, 10.7, 1.57]^T$ from the measurements in $\mathbf{Z}_1$ and $\mathbf{Z}_2$ in the Munkholm boat LIDAR dataset.

Figure 6.6. This seems like a good estimate of position, heading and length, but the width is a bit too small. For the next time steps it captures the true width of the boat better, and in $k = 490$ it is close to the true width, but the length estimate is too small. The measurement generating points $\mathbf{y}_k^j$ are from the hypothesis E_i^j with highest association weight $\beta_k^{i,j}$. They seem to lie close to the centroids in all the time steps, which explains why they are positioned on the filtered ellipse because the prediction is close to the update.

Observe in time step $k = 172$ that there are eight cluster centroids located on the contour of the boat, while two of them are laying more in the middle. The predicted measurements in the front part of the filtered ellipse are close to the front centroid from the middle of the boat. This centroid makes the ellipse position slightly shifted away from the contour centroids in the front, and shows that the C-GPDA filter can be sensitive to detections close to the contour. In $k = 342$ we also see that the rearmost centroid generated by the wakes gives the target ellipse a shift backwards.

Since we have the true extent parameters we can calculate the RMSE and separate NEES measures for a_k and b_k. In Figure 6.8a and 6.8b we have plotted the RMSE together with the mean as a red line. We clearly see higher errors in the major axis estimates, which is probably because the LIDAR data gives less information about the length compared to the width during the track. For instance in the last time steps the LIDAR only detects the rear end, and therefore we get an increase in a_k-errors while the b_k-errors are relatively stable. However, there are some peaks in the b_k-errors that are over 1 meter, which occurs during the backing ($k = 20, ..., 90$) and turning maneuver ($k = 190, ..., 270$). During the

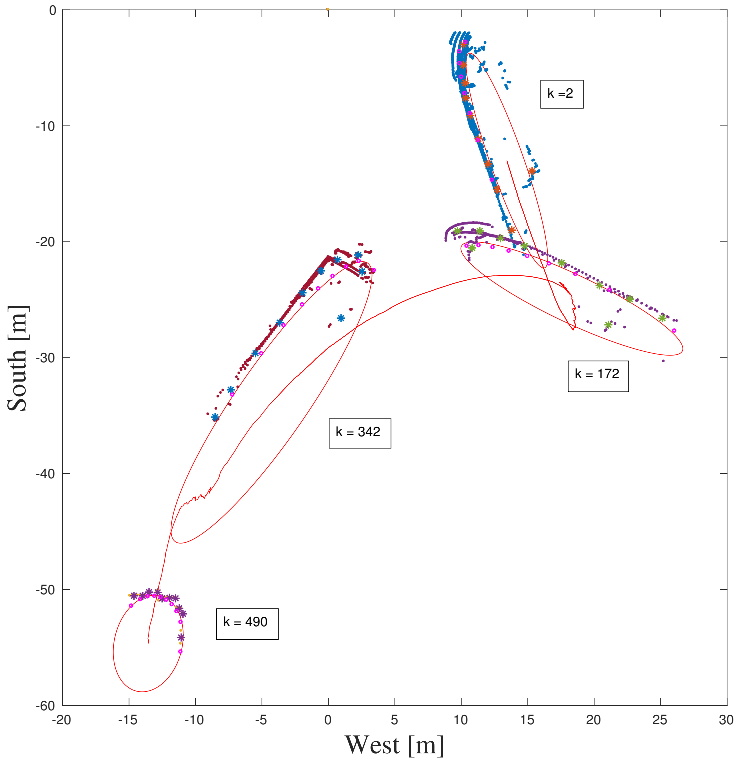
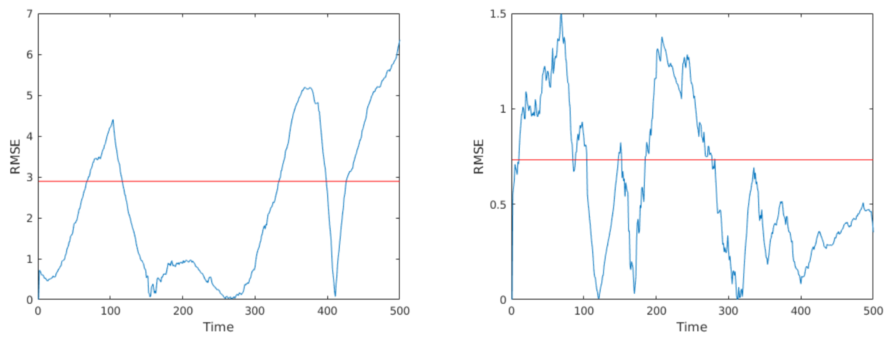


Figure 6.7: Tracking plot of C-GPDA estimates $\mathbf{m}_k$ with target position and ellipse (red) together with the measurements and cluster centroids. Measurements and centroids are from $k = 2$ (blue, red), 172 (purple, beige), 342 (burgundy, blue), and 490 (purple). The predicted measurements $\tilde{\mathbf{y}}_k^j$, for the hypothesis E_i^j with highest association weight, are plotted in magenta for all time steps.

turning maneuver we observe a decrease in the a_k errors because the full length of the boat gets visible. This shows that the C-GPDA algorithm is able to fit an accurate ellipse to the measurements when they provide a sufficient amount of information about either the major or minor axis of the ellipse.

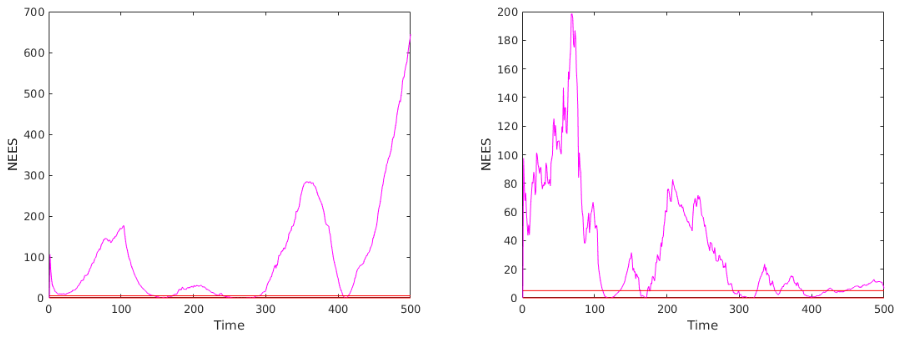
In Figure 6.9a and 6.9b the NEES plots show that the a_k- and b_k-estimates are not consistent for most of the time steps. The curves follow the RMSE pattern very closely, which means that the estimated covariance $\mathbf{P}_k$ is not changing much during the track. The initial estimate $\mathbf{P}_0$ is set to the process noise covariance matrix $\mathbf{Q}$, which is created with acceleration noise $\sigma_a = 0.15$ and extent noise $\Sigma_{a,b} = \text{diag}[0.05^2, 0.04^2]$. These values are tuned by looking at the track plot and higher values of these variances would lead to a more irregular track. The measurement noise covariance matrix were set to $\mathbf{R} = \text{diag}[1.1^2, 0.7^2]$, which is also tuned from the track plot. The covariance matrices $\mathbf{R}_k^j$ are rotated tangential



(a) a_k RMSE **(b)** b_k RMSE

Figure 6.8: RMSE plots for the extent variables a_k and b_k in the C-GPDA method on LIDAR data from the Munkholm boat. The red line is the mean RMSE.

to the predicted ellipse, because the cluster centroids have varying positions along the visible contour of the boat during the track. There are a lot of LIDAR measurements located at the contour, which can be seen in Figure 6.6 for instance. Hence the uncertainty in cluster centroid positions are tangential on the predicted ellipse.



(a) a_k NEES **(b)** b_k NEES

Figure 6.9: NEES plots for the extent estimates from the C-GPDA filter on LIDAR data from the Munkholm boat.

In the kinematic process, we do not have the full true state to compare the estimates with, and we need a way to measure the performance of the filter without using the RMSE and NEES. To do this we use the Normalized Innovation Squared (NIS) from Bar-Shalom and Li [1995] as

$$\text{NIS} = \mathbf{v}_k^T \mathbf{S}_{k|k-1}^{-1} \mathbf{v}_k, \tag{6.1}$$

where $\mathbf{v}_k \in \mathbb{R}^{m_k}$ is the innovation vector, and $\mathbf{S}_{k|k-1} \in \mathbb{R}^{m_k \times m_k}$ is the innovation covariance matrix. Since it is assumed that $\mathbf{v}_k \sim \mathcal{N}(\mathbf{v}_k; 0, \mathbf{S}_{k|k-1})$, the NIS should follow a χ^2-distribution. Hence there exists a $(1-\alpha)$-confidence interval similar to the NEES$_k$ interval. In the original definition in (6.1) it will have m_k degrees of freedom, where m_k is the length of $\mathbf{v}_k$. However, when we have multiple innovations $\mathbf{v}_k^{i,j} \in \mathbb{R}^{2j}$ in the C-GPDA filter, we get different degrees of freedom $2j$ for each hypothesis E_i^j. If we try to reduce it to one χ^2-distribution by mixture reduction, we get an unknown distribution. Hence we look at the NIS for the hypothesis that have the maximal association weight, and only in the time steps where $\beta_k^{max} > 0.99$. We define this quantity as $\beta_k^{max} = \max_{i,j} \beta_k^{i,j}$, and use the innovation $\mathbf{v}_k^{i,j}$ and covariance matrix $\mathbf{S}_{k|k-1}^j$ in (6.1). Here $\mathbf{v}_k^{i,j} = \tilde{\mathbf{z}}_k^{i,j} - \tilde{\mathbf{y}}_k^j$ is the innovation vector for the time steps k where $\beta_k^{max} > 0.99$ and $\mathbf{S}_{k|k-1}^j$ is the corresponding innovation covariance matrix given in (4.17).

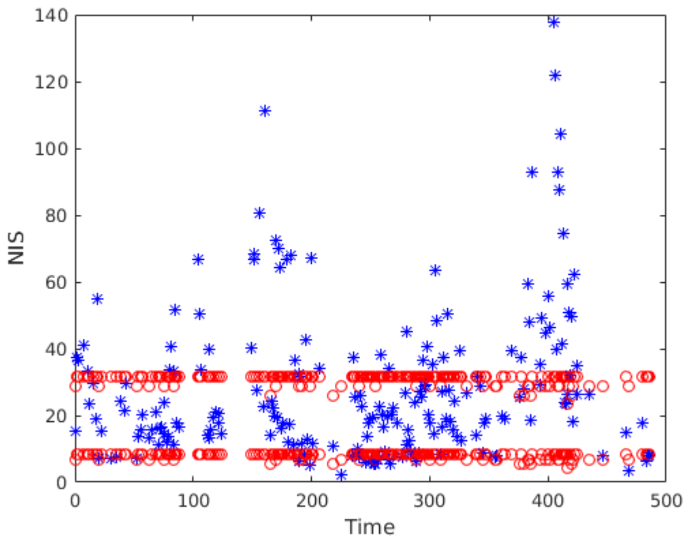


Figure 6.10: Plot of the NIS (blue stars) for the C-GPDA estimated state vector $\mathbf{m}_k$, where $\beta_k^{max} > 0.99$. The red circles represent the χ^2_{2j}-confidence intervals that have $2j$ degrees of freedom depending on E_i^j.

In Figure 6.10 the NIS is plotted for the C-GPDA track of the Munkholm boat together with the confidence intervals as red dots. There are 263 time steps where the NIS is defined for β_k^{max}, and it is inside the confidence interval for 56.22 % of the time. The intervals have varying upper and lower limit because of the different degrees of freedom $2j$, but we see that most of the intervals are $[8.23, 31.53]$ which is the 95-% confidence interval for a χ^2_{18}-variable. This means that the C-GPDA filter chooses a hypothesis with $j = 9$ points from target in most of the time steps. We observe that the NIS have some peaks above the intervals, during the turning maneuver ($k = 180, ..., 200$) and in the final time steps around $k = 400$ when the boat is leaving the harbor.

6.3 Random matrix GPDA results

The random matrix GPDA filter results from the same LIDAR data as in the previous section are shown in Figure 6.11, and it confirms what we would expect from the simulation chapter. The parameters used in the filter are given by

$$z = 1/2, \quad \tau = 0.21, \quad \eta = 2000, \quad \sigma_a = 0.1, \tag{6.2}$$

$$\mathbf{R}^{\text{RM}} = \text{diag}[0.2^2, 0.1^2], \tag{6.3}$$

which are set by tuning the filter. Initially the ellipse is close to the initialization estimate in Figure 6.6. In time step $k = 172$ it seems to have lost the track, and it has an irregular motion together with a wrong estimate of heading and extent. However, it gets better in $k = 342$ were it has captured the true extent and position better, but there are still obvious errors, especially in the heading. In the last time step it is clearly too small and wrong positioned, but it is hard to obtain a good estimate in this case as we saw for the C-GPDA too.

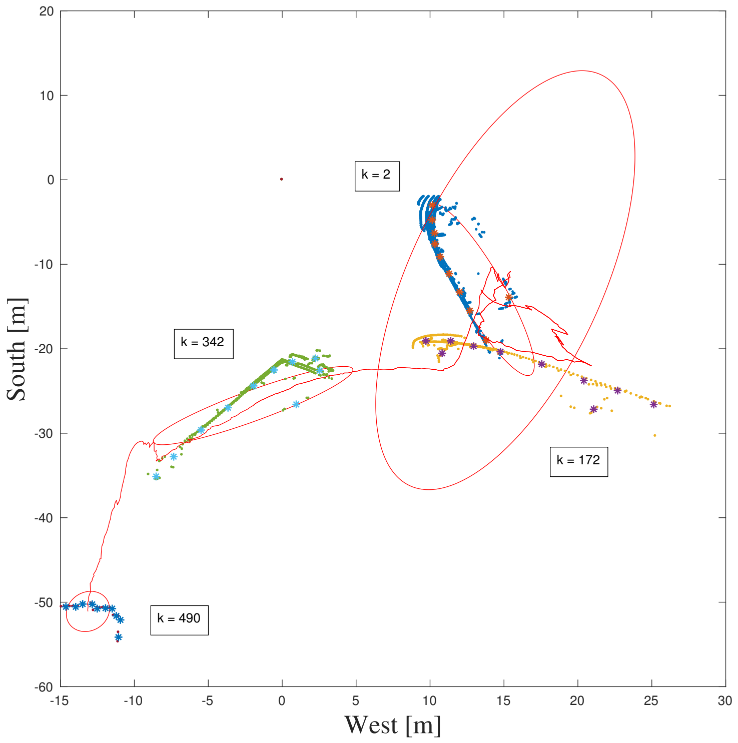
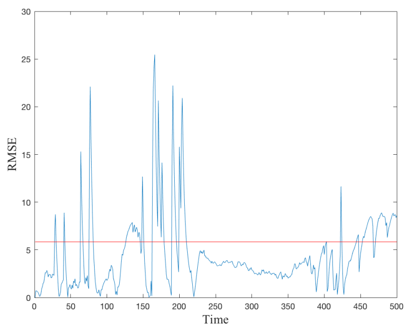
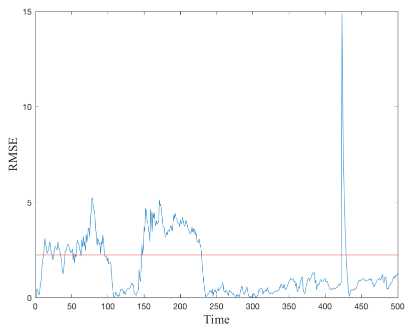


Figure 6.11: Plot of random matrix GPDA estimated track and target ellipse on the cluster centroids. Measurements are from $k = 2$ (blue, red), 172 (beige, purple), 342 (green, cyan), and 490 (blue).

The RMSE plots in Figure 6.12a and 6.12b show what we would expect from the track plot, which is high extent errors for the a_k and b_k estimates. These are found in the same way as in the simulation studies, by calculating the eigenvalues of $\mathbf{M}_k$. The magnitude of the errors are much higher than in the C-GPDA filter, but in some time steps it drops to zero. The mean values plotted as the red lines are also high because of the many peaks during the track. We do not have an estimated variance for the a_k and b_k estimates used, only the matrix $\mathbf{V}_k$ which is the variance for $\mathbf{M}_k$, and hence we have no NEES for the random matrix estimates.



(a) a_k RMSE



(b) b_k RMSE

Figure 6.12: RMSE plots for the extent estimation in the random matrix GPDA on LIDAR data from the Munkholm boat.

The NIS measure for random matrix GPDA is analogous to the C-GPDA NIS, but is defined as

$$\text{NIS}_k = (\bar{\mathbf{z}}_k^{i,j} - \mathbf{Hm}_{k|k-1})^T \mathbf{S}_{k|k-1}^{-1} (\bar{\mathbf{z}}_k^{i,j} - \mathbf{Hm}_{k|k-1}) \sim \chi_2^2. \tag{6.4}$$

This measure has 2 degrees of freedom in each hypothesis E_i^j because the innovation vector has length 2 regardless of which hypothesis we investigate. However, we will only use the NIS in time steps where $\beta_k^{max} > 0.99$, so it will only be used in time steps were the filter is certain that it has chosen the right hypothesis. This we do to make it comparable with the NIS for C-GPDA.

The NIS measure of the random matrix GPDA is plotted in Figure 6.13 and the measure is defined in 42 time steps, which is far less than the C-GPDA filter. This indicates that the random matrix GPDA filter is more uncertain in choosing one hypothesis compared to the C-GPDA. The NIS is inside the confidence interval 66.67 % of the time, which indicates that the filter think the innovations are consistent in most of the time steps. By considering the tracking results and extent absolute errors together with the NIS we can say that the random matrix filter give deviant estimates and think it is estimating the target state correctly.

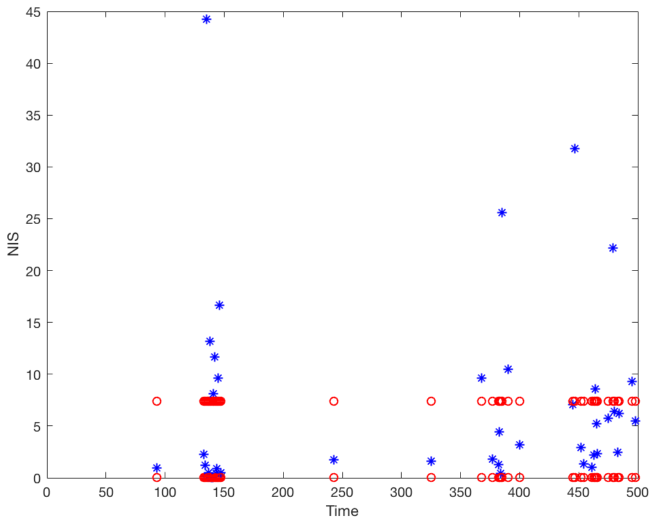


Figure 6.13: Plot of the NIS (blue stars) for the random matrix GPDA estimated state vector $\mathbf{m}_k$, where $\beta_k^{max} > 0.99$. The red dots represent the χ_2^2-confidence intervals.

Chapter 7

Discussion

In this chapter we will discuss the tracking results from the simulation experiments and real LIDAR data. Different implementation methods and other modelling choices are also discussed.

7.1 Simulation experiments

From the experiments presented in chapter 5, there is no doubt that the CEKF and C-GPDA filters are superior to the random matrix filters. Considering the theoretical background for each method, as presented in Chapter 3, this comes as no surprise because the CEKF method is specifically constructed to handle LIDAR data with predicted measurements that are distributed along the predicted target contour.

In the random matrix framework, the data are assumed to be scattered on and around the surface, which is typically the pattern we get from a radar. The centroid calculation rule in (3.61) which is supposed to make the random matrix method able to track a single target from LIDAR data, is not robust at all. When looking at different LIDAR data images from an elliptical target, it is obvious that this rule will fail to capture the true ellipse center in some cases. By comparing the estimates with the ground truth we have found that the random matrix method generally has so high errors that it is not competitive with the CEKF or the C-GPDA method in any case tested. Hence the decision of choosing the best method is easy, and we will focus on discussing the CEKF and C-GPDA experiments from here.

It is evident that the C-GPDA filter is a consistent tracking method when the true state is given initially. The absolute errors are low, and the NEES lies inside the confidence interval in almost all time steps regardless of the clutter pattern that is around the target. However, in the last simulation experiment with simulated wakes, it started to get too optimistic about its estimates, but the RMSE was still low. This showed us that the C-GPDA is able to give low absolute errors, and consistent results up to a certain amount of clutter measurements. However, in a real life tracking scenario we can not expect to know the ground truth initially, and thus the initialization algorithm were created.

In the automated C-GPDA filter the initialization in Algorithm 5 were used, and the results gave higher estimation errors and was less consistent than with ground truth initialization. This is as expected, but it is interesting to see how the method performs with no prior knowledge of the target state. To prepare the C-GPDA algorithm for the real LIDAR data case, it was important to check if the initialization method worked on simulated data. In the case with no clutter measurements in subsection 5.2.1 the initial errors were high, but eventually they got smaller throughout the the tracking period. The filter also got more consistent in the last time steps. With the more difficult case in subsection 5.3.2 the automated C-GPDA struggled both with initialization and filtering correctly. It is a hard problem to fit an ellipse correctly in a noisy environment, and this is shown in the initial errors of the experiments. To find the ground truth state in the time steps after a wrong initialization is also a hard problem when clutter measurements are present, and the filter did not manage to do this.

When we compare the extent estimation results in Table 5.2 with the real data RMSE plots in Figure 6.12a and 6.8b, we see that the mean values for a and b have approximately the same values. This is interesting because it shows that the automated C-GPDA gives approximately the same extent errors in all the 100 simulated tracks in addition to the Munkholm boat track. This indicates a robustness in the algorithm, but the errors could of course have been lower.

To say that the automated C-GPDA filter was given no information is not entirely correct, because it is provided with the subtraction parameter g to find n^{min}, the true covariance matrices $\mathbf{Q}$ and $\mathbf{R}$ and clutter parameter λ. The λ- and g-parameters are useful when we have prior information about how much clutter there is in the data. In the simulation experiments we knew how much clutter we would expect to get, and these parameters were set accordingly. Also, if $n^{min} > n_k^t$ in some time steps, the subtraction parameter was lowered such that it did not exclude the correct hypothesis. As we have discussed earlier, the true parameters $\mathbf{Q}$ and $\mathbf{R}$ will give the filter a wrong update of the covariances in $\mathbf{P}_k$, which is way lower than they should be. This is why we introduced the tuning parameters $\mathbf{Q}^{tun}$ and $\mathbf{R}^{tun}$, which are set with higher values so that the filter gets a more realistic covariance update. This also increased the chance of finding the ground truth state after the deviant initialization, because of a higher variance in the estimates.

The parameter λ is set to a low value in the measurement simulations because we would avoid getting too many association hypotheses. When using this in the C-GPDA filter it will lead to high prior probabilities for hypotheses E_i^j with high j, because of the prior term $j!/(\lambda V_k)^j$ from (4.13). By looking at the actual choice the algorithm makes in Figure 5.12 we found that it chooses the correct hypothesis in most cases, even though there are several clutter points in the validation region. This indicates that the spatial density $p_{sp}(\Theta_k|\cdot)$ dominates the association weight expression in (4.18), and removes the bias that the prior gives towards higher j.

To avoid getting the factor $j!/(\lambda V_k)^j$ in the prior expression (4.13), we tried to change the prior density for target generated measurements $P(n_k^t)$. When modelling it as a Poisson distribution with parameter λ_t, we get the prior factor $(\lambda_t/\lambda)^j$ instead in (4.13). By introducing this parametric prior we can control more of the prior association probability by setting a realistic λ^t. However there are one obvious drawback with modelling n_k^t as a Poisson distribution, because it is possible to get a huge amount of detections when simulating from it. This would have made the running time for the simulations much longer, and we would not be able to control the minimum and maximum number of target measurements like we did with the uniform distribution.

Another possibility for the n_k^t prior is a binomial distribution with probability p_t and parameter n_t. By doing this we have an upper limit for the number of target generated measurements, but there is still a chance for getting low a low number of measurements in the simulations. The resulting factor in the prior density (4.13) becomes $\frac{p_t^j}{(1-p_t)^j \lambda V_k (m_k-j)!}$, given that $n_t = m_k$. By setting a realistic value for p_t based on prior knowledge for n_k^t, this prior expression would assign probabilities to hypotheses E_i^j accordingly. However, when choosing the n_k^t-prior it is important to think of the tracking problem we want to solve, and when tracking the Munkholm boat we found that a uniform distribution would be the best. This is because of its simplicity and amount of control on the interval where n_k^t should lie.

7.2 Real LIDAR data experiments

In the LIDAR data experiments we got the results that we expected based on the simulation experiments. The C-GPDA filter performed better than random matrix GPDA, when looking at track plots, extent RMSE and NIS. However, we can not be entirely sure how the methods performed in the kinematic estimation, but by looking at the LIDAR measurements we get the main idea of how they performed. The C-GPDA results were acceptable in the extent estimation with not too high errors. It gave a good reconstruction of the Munkholm boat departure, and did not give estimates that were a long way from the assumed true state.

The K-means clustering algorithm on the original dataset enabled us to determine the number of measurements m_k prior to the filtering process. This is different from the simulation studies where we sampled the clutter measurements from a Poisson distribution which gave different numbers each time step. In fact, the clustering could actually help remove some clutter points, because the majority of the points will lie around the target and thus move the centroids closer to the target. This is not always the case, as we have seen from time step $k = 203$ in Figure 6.5 for instance. When we get a small number of measurements each time step, this clustering procedure should not be used. Then it is better to just use the original measurements, and this can happen in cases where the target is small or the distance from target to LIDAR is longer.

The predicted measurements of the C-GPDA method proved to be a good way to model the cluster centroid positions. Throughout the whole tracking process they seemed to lie close to the real measurements. Especially in the turning maneuver where the cluster centroids only came from the front of the boat, and when measurements from the side began to emerge the extent estimation adapted itself to the generated centroids. However, when centroids were generated from other measurements around the target contour like the middle of the boat or wakes, we observed a shift in the ellipse estimates. It is difficult to deal with these centroids because they are close to the predicted measurements and thus get high association weights $\beta_k^{i,j}$. A solution to this would be to model this type of clutter in the C-GPDA filter, so the ellipse estimate will not be biased towards these points.
When the boat left the canal, and only the rear end was visible, the earlier extent estimates was passed through and made the major axis estimates gradually smaller. This was not the case with the random matrix GPDA extent estimate, which got small almost immediately after the target side was not visible anymore.

7.3 Other implementation techniques

In section 4.4 we tried different implementation methods for initialization and finding n^{min}, and it was difficult to find a general method that worked in all the tracking scenarios we tested. This is mainly caused by the infinitely many measurement configurations that can arise when both target and clutter measurements are present. If we knew which points that were target generated prior to the filtering, the job would be much easier.

The ellipse-fitting method described in section 4.4 is biased towards smaller ellipses, so if the target generated measurements are found, the fitted ellipse may be small compared to the true target size. This proved to be a problem when the random sample consensus (RANSAC) was used as initialization procedure. This is a method that randomly samples different permutations of the measurement set with size j, when $j = 3, ..., m_k$. It then tries to estimate an ellipse to the chosen points, and this is why j starts at 3, which is the minimum number of points required to do this. The algebraic distance $F(\mathbf{a}, \mathbf{c})$ is computed for the fitted ellipse, and this works as a score for how well the ellipse fit the data. If this distance is better than in previous iterations, it is chosen as the new best solution. The points that are sampled are called inliers, and we try to find as many of them as possible. The ellipse with the lowest algebraic distance to the sampled inliers will be the output of the algorithm.

The problem with the RANSAC method is that it often chooses clutter points as inliers because they coincidentally fit in an ellipse with either target or other clutter measurements. It also chooses small ellipses fitted to the target points because they often have smaller algebraic distances than the larger ones. This makes the ellipse initialization deviant from the true target state and it will be difficult for the filters to deal with. Hence we chose the method with MAD outlier removal, before fitting the ellipse to the resulting points. This is not a flawless method, but it generally worked better than RANSAC.

When finding n^{min} we also tried to set it as the number of inliers in the RANSAC method, but it was too often excluding the correct hypothesis or were set too low such that the hypothesis space remained large. Another approach that were tried was to first remove outliers with MAD in the data points at each time step, and fit an ellipse to these points using the least squares method. Then we calculated the tangent points (x_k^t, y_k^t) on the ellipse that had tangent line through the origin, illustrated in Figure 3.4. From these two tangent points we defined a line $y = Ax + B$ that went through both points and separated the measurement points. The number of measurement points that were on the side of the line closest to the LIDAR were chosen as n^{min}. This proved to be a quite deviant method, with estimates both too small, and way too large in the simulations.

In the end we found that none of the methods we tried for initialization and finding n^{min} would give accurate estimates in all cases we tried to throw at them. Hence we settled with a simple outlier detection algorithm using the MAD to remove points from the measurement set. This proved to be an effective method in most cases, and got rid of the worst outliers so that the ellipse-fitting algorithm got either only the target generated points, or some clutter points in addition. The solution for n^{min} with the subtraction parameter g was chosen based on the fact that it is better to set n^{min} too low than too high. This is because the filter only gets slower when it is too low, but it will not find the correct hypothesis E_i^j when it is too high.

Chapter 8

Conclusion

8.1 Concluding remarks

This thesis has presented a new approach for dealing with the single extended object tracking (EOT) problem in clutter by combining the contour extended Kalman filter (CEKF) in Granström et al. [2011b] with the general probabilistic data association filter (GPDA) from Schubert et al. [2012]. The modelling of the GPDA filter was done in a different way by making inference on the association hypothesis prior as opposed to the target and clutter likelihood that were done in the original approach. This new method called the contour GPDA (C-GPDA) filter, was compared with the random matrix GPDA from Schuster and Reuter [2015], which proved to be more deviant in all experiments.

Through simulation experiments of LIDAR measurements from an elliptical target we have shown that the C-GPDA method is accurate and consistent in its estimates of the target position, velocity and extent. The CEKF tracks the single target with low errors without clutter, and the C-GPDA filter handles clutter measurements in a robust way. Compared to the random matrix approach there is no doubt that the C-GPDA is the best filter to use when considering both efficiency and accuracy. This is because it is specifically designed for tracking with LIDAR measurements, which are distributed along the target contour. The random matrix method was originally developed for group target tracking, which gives a measurement spread over the whole group of targets. The modification of random matrix presented in Schuster and Reuter [2015] gives an intuitive rule to handle LIDAR measurements, but we have seen that this rule does not always work.

When tracking with real LIDAR data from the Munkholm boat in Ravnkloa, the C-GPDA filter proved to be the best method. It handled the departure track in a good manner, and estimated the extent with low absolute errors in most of the time steps. The random matrix GPDA filter struggled more with the extent estimation and was not able to give a reasonable track of the boat. Considering the simulation results, this outcome was expected, and shows again that the C-GPDA method is preferable.

In addition we have tried to make the filters more automated by finding an initialization procedure and a way to cut down the association hypothesis space $\mathcal{E}$. This proved to be a difficult task, and the C-GPDA algorithm struggled to give correct estimates in a simulated cluttered environment when using this, but in the real LIDAR data with small amount of clutter the methods worked better. By combining the methods of CEKF and GPDA we have found a simple alternative to the more complex random finite set framework. In this sense we have made a contribution to the extended object tracking research field, by solving the problem of single target tracking in clutter with LIDAR in an easy way.

8.2 Future work

The next step in testing the C-GPDA filter will be to compare it with a point target method using clustering of the measurements. This will show if the EOT modelling is beneficial when dealing with single target tracking in clutter. The C-GPDA filter could also be extended to handle wake clutter by using a similar approach to what was done in Brekke et al. [2012] for the traditional PDA filter. This will show how robust the C-GPDA method is to a more advanced clutter pattern close to the target.

To generalize the GPDA further into a multi-target method the framework of joint PDA (JPDA) or Poisson multi-Bernoulli mixture (PMBM) could be used. The PMBM framework is a generalization of JPDA, which again is a generalization of PDA. Hence it is reasonable to start by generalizing GPDA by using JPDA, and then generalize it further with PMBM which represents the state-of-the-art in multi-target tracking.

To test the C-GPDA filter with different prior densities for the number of measurements from target n_k^t, is also an interesting topic for future work. In this thesis we modelled it as a discrete uniform distribution, but it is possible to use a Poisson or binomial distribution as well. An analysis of how the C-GPDA filter performs when using these prior densities for n_k^t will give more insight into how essential the prior term is in the association weights.

A further study of initialization algorithms and association hypothesis reduction with LIDAR data is a natural extension of this work. It is a hard problem to solve and there are most likely more robust algorithms that can be used instead of the ones used in this thesis. Heuristic approaches for searching after the relevant hypotheses, or sampling methods like the stochastic optimization used in Granström et al. [2018], is probably a better way of solving this problem. If the C-GPDA algorithm were to be implemented on the autonomous ferry it should have a more general procedure to remove all clutter points from the surroundings as well. This was done manually in our real LIDAR data experiment, by cutting the domain.

Bibliography

Bar-Shalom, Y., Li, X., 1995. Multitarget-Multisensor Tracking. YBS publishing, England.

Bar-Shalom, Y., Li, X., Kirubarajan, T., 2001. Estimation with Applications To Tracking and Navigation. John Wiley & Sons, Inc.

Brekke, E., Hallingstad, O., Glattetre, J., 2012. Improved target tracking in the presence of wakes. IEEE Transactions on Aerospace and Electronic Systems 48 (2).

Chilton, A., 2014. The Working Principle and Key Applications of Infrared Sensors. Accessed: 15.05.2018.
URL `https://www.azosensors.com/article.aspx?ArticleID=339`

Drummond, O., Blackman, S., Hell, K., 1988. Multiple sensor tracking of clusters and extended objects. In: Technical Proceedings: 1988 Tri-Service Data Fusion Symposium.

Eide, J., 2016. MS Nidarholm. Accessed: 16.06.2018.
URL `https://no.wikipedia.org/wiki/MS_C2ABNidarholmC2BB`

Feldmann, M., Frnken, D., Koch, J., 2011. Tracking of extended objects and group targets using random matrices. IEEE transactions on signal processing 59 (4), 1409–1420.

Fitzgibbon, A., Pilu, M., Fisher, R. B., 1999. Direct least square fitting of ellipses. IEEE Transactions on pattern analysis and machine intelligence 21 (5), 477–480.

Granström, K., Fatemi, M., Svensson, L., 2017a. Poisson multi-bernoulli conjugate prior for multiple extended object estimation. arXiv:1605.06311v3.

Granström, K., Lundquist, C., Orguner, O., 2011a. Extended target tracking using a gaussian-mixture phd filter. IEEE transactions on aerospace and electronic systems 48 (4), 3268–3286.

Granström, K., Lundquist, C., Orguner, U., 2011b. Tracking rectangular and elliptical extended targets using laser measurements. In: 14th International Conference on Information Fusion. Chicago, Illinois, USA, pp. 1–5.

Granström, K., Orguner, U., 2014. New prediction for extended targets with random matrices. IEEE transactions on aerospace and electronic systems 50 (2), 1577–1589.

Granström, K., Reuter, S., Fatemi, M., Svensson, L., 2017b. Pedestrian tracking using velodyne data - stochastic optimization for extended object tracking. In: Proc. IEEE Intell. Veh. Symp. Redondo Beach, California, USA.

Granström, K., Reuter, S., Meissner, D., Scheel, A., 2014. A multiple model phd approach to tracking of cars under an assumed rectangular shape. In: 17th International Conference on Information Fusion. Salamanca, Spain.

Granström, K., Svensson, L., Reuter, S., Xia, Y., Fatemi, M., 2018. Likelihood-based data association for extended object tracking using sampling methods. IEEE Transactions on intelligent vehicles 3 (1), 30–45.

Gupta, A., Nagar, D. K., 2000. Matrix variate distributions. Chapman & Hall.

Habtemariam, B. K., Tharmarasa, R., Kirubarajan, T., Grimmett, D., Wakayama, C., 2011. Multiple detection probabilistic data association filter for multistatic target tracking. In: 14th International Conference on Information Fusion. Chicago, Illinois, USA.

Kalman, R., 1960. A new approach to linear filtering and prediction problems. Journal of Basic Engineering 82-D (1), 35–45.

Koch, J., 2008. Bayesian approach to extended object and cluster tracking using random matrices. IEEE transactions on aerospace and electronic systems 44 (3), 1042–1059.

Kongsberg, 2017. Autonomous ship project, key facts about yara birkeland. Accessed: 15.05.2018.
URL `https://www.km.kongsberg.com/ks/web/nokbg0240.nsf/AllWeb/4B8113B707A50A4FC125811D00407045?OpenDocument`

Lan, J., Li, X., 2012. Tracking of extended object or target group using random matrix: New model and approach. In: Proceedings of the International Conference on Information Fusion. Singapore, pp. 2177–2184.

Li, X., Vilkov, V., 2003. Survey of maneuvering target tracking. part i. dynamic models. IEEE Transactions on Aerospace and Electronic Systems 39 (4), 1333–1364.

Loan, C., 1978. Computing integrals involving the matrix exponential. IEEE transactions on automatic control AC-23 (3), 395–404.

Mahler, R., 2003. Multitarget bayes filtering via first-order multi target moments. IEEE transactions on aerospace and electronic systems 39 (4), 1152–1178.

Reid, D., 1979. An algorithm for tracking multiple targets. IEEE transactions on automatic control 24 (6), 843–1420.

Salmond, D., Ristic, B., 2004. A study of a nonlinear filtering problem for tracking an extended target. In: 7th International Conference on Information Fusion. Stockholm, Sweden.

Särkkä, S., 2013. Bayesian Filtering and Smoothing. Cambridge University Press, Cambridge, England.

Schubert, R., Adam, C., Richter, E., Bauer, S., Lietz, H., Wanielik, G., 2012. Generalized probabilistic data association for vehicle tracking under clutter. In: Intelligent Vehicles Symposium. Alcal de Henares, Spain, pp. 962–968.

Schuster, M., Reuter, J., 2015. Target tracking in marine environment using automotive radar and laser range sensor. In: IEEE 20th International Conference on Methods and Models in Automation and Robotics. Midzyzdroje, Poland, pp. 965–970.

Rødningsby, A., 2010. Multitarget Multisensor Tracking in the Presence of Wakes. PhD thesis, NTNU.

Wilthil, E., Flåten, A., Brekke, E., 2017. A target tracking system for asv collision avoidance based on the pdaf. Sensing and Control for Autonomous Vehicles, Springer, 269–288.