Linköping studies in science and technology. Dissertations.
No. 1476

Extended target tracking using PHD filters

Karl Granström

Department of Electrical Engineering
Linköping University, SE–581 83 Linköping, Sweden

Linköping 2012

Cover illustration: Laser range data used in extended target tracking experiments. Two persons were measured at waist height as they walked around in front of the laser range sensor. The data is plotted in three dimensions, with time along the vertical axis. Tracking results for this data set are presented in Figure 11 in Paper B, and in Figure 9 in Paper D.

Nota bene: The illustration features no less than 400 different colors. No colors were harmed in the making of this cover illustration.

Linköping studies in science and technology. Dissertations.
No. 1476

Extended target tracking using PHD filters

Karl Granström

`karl@isy.liu.se`
`www.control.isy.liu.se`
Division of Automatic Control
Department of Electrical Engineering
Linköping University
SE–581 83 Linköping
Sweden

ISBN 978-91-7519-796-8 ISSN 0345-7524

Printed by LiU-Tryck, Linköping, Sweden 2012

Till mina föräldrar

Abstract

The world in which we live is becoming more and more automated, exemplified by the numerous robots, or autonomous vehicles, that operate in air, on land, or in water. These robots perform a wide array of different tasks, ranging from the dangerous, such as underground mining, to the boring, such as vacuum cleaning. In common for all different robots is that they must possess a certain degree of awareness, both of themselves and of the world in which they operate. This thesis considers aspects of two research problems associated with this, more specifically the Simultaneous Localization and Mapping (SLAM) problem and the Multiple Target Tracking (MTT) problem.

The SLAM problem consists of having the robot create a map of an environment and simultaneously localize itself in the same map. One way to reduce the effect of small errors that inevitably accumulate over time, and could significantly distort the SLAM result, is to detect loop closure. In this thesis loop closure detection is considered for robots equipped with laser range sensors. Machine learning is used to construct a loop closure detection classifier, and experiments show that the classifier compares well to related work.

The resulting SLAM map should only contain stationary objects, however the world also contains moving objects, and to function well a robot should be able to handle both types of objects. The MTT problem consists of having the robot keep track of where the moving objects, called targets, are located, and how these targets are moving. This function has a wide range of applications, including tracking of pedestrians, bicycles and cars in urban environments. Solving the MTT problem can be decomposed into two parts: one part is finding out the number of targets, the other part is finding out what the states of the individual targets are.

In this thesis the emphasis is on tracking of so called extended targets. An extended target is a target that can generate any number of measurements, as opposed to a point target that generates at most one measurement. More than one measurement per target raise interesting possibilities to estimate the size and the shape of the target. One way to model the number of targets and the target states is to use random finite sets, which leads to the Probability Hypothesis Density (PHD) filters. Two implementations of an extended target PHD filter are given, one using Gaussian mixtures and one using Gaussian inverse Wishart (GIW) mixtures. Two models for the size and shape of an extended target measured with laser range sensors are suggested. A framework for estimation of the number of measurements generated by the targets is presented, and reduction of GIW mixtures is addressed. Prediction, spawning and combination of extended targets modeled using GIW distributions is also presented. The extended target tracking functions are evaluated in simulations and in experiments with laser range data.

Populärvetenskaplig sammanfattning

Den värld i vilken vi lever har med tiden blivit allt mer automatiserad. Ett av många tecken på detta är det stora antal robotar, eller autonoma farkoster, som verkar bland annat i luften, på land, eller i vatten. De här robotarna kan utföra ett brett spektrum av olika uppgifter, allt ifrån direkt farliga, som underjordisk gruvdrift och sanering av havererade kärnreaktorer, till alldagliga och tråkiga, som dammsugning och gräsklippning. På samma sätt som en människa behöver använda sina sinnen och sitt medvetande för att hantera vardagen, måste alla typer av robotar ha en viss medvetenhet för att kunna utföra sina uppgifter. Det krävs bland annat att robotarna kan uppfatta och förstå sin arbetsmiljö.

I den här avhandlingen behandlas ett antal delar av två stycken övergripande forskningsproblem som är relaterade till detta. Det första forskningsproblemet kallas för samtidig positionering och kartering, vilket på engelska heter Simultaneous Localization and Mapping och förkortas SLAM. Det andra forskningsproblemet kallas för målföljning.

SLAM-problemet går ut på att låta roboten skapa en karta av ett område, och samtidigt som kartan skapas positionera sig i den. Exakt vad som menas med karta i det här sammanhanget varierar beroende på robotens specifika arbetsuppgift. Exempelvis kan det, för en inomhusrobot, röra sig om en virtuell modell av var golv, väggar och möbler finns i ett hus. En oundviklig del av SLAM-problemet är att roboten hela tiden gör små fel, vilket påverkar kartan som skapas, samt hur väl roboten kan positionera sig. Enskilda fel har inte särskilt stor inverkan, men om felen ackumuleras under en längre tid kan det leda till att kartan förvrängs, eller att roboten helt enkelt inte kan finna sin position i kartan.

Ett sätt att undvika att så sker är att utrusta roboten med en funktion vilken gör det möjligt för roboten att känna igen platser som den har besökt tidigare, vilket kallas platsigenkänning. När roboten känner igen en plats kan den jämföra med vad kartan och positionen säger. Om kartan och positionen inte säger att roboten är tillbaka på en plats som tidigare besökts kan denna diskrepans korrigeras. Resultatet är en karta och en position som bättre representerar verkligheten. I den här avhandlingen har platsigenkänning studerats för robotar som är utrustade med laserscanners, och en funktion för platsigenkänning har skapats. I en serie experiment har det visats att funktionen kan känna igen platser såväl inomhus i kontorsmiljö, som utomhus i stadsmiljö. Det har även visats att funktionens egenskaper jämför sig väl med tidigare arbete på området.

Den resulterande SLAM-kartan bör av naturliga skäl endast innehålla stationära föremål. Vår värld innehåller dock även rörliga föremål, och för att en robot ska kunna arbeta på ett säkert sätt måste den även hålla reda på alla rörliga föremål som finns i dess närhet. Det andra forskningsproblemet som behandlats i avhandlingen, målföljning, går ut på att utrusta roboten med funktioner som gör det möjligt för den att hålla reda på var de rörliga målen är, samt vart de är på väg att röra sig. Exempelvis kan den här typen av funktioner användas till att hålla reda på fotgängare, cyklister och bilar i en stadsmiljö.

Tidigare har forskningen inom målföljning varit fokuserad på så kallade punktmål. Vid följning av punktmål kan följningsproblemet sägas ha två delar: den ena är att räkna ut hur många rörliga mål det finns, den andra är att räkna ut var varje enskilt mål befinner sig, samt vart det är på väg.

Här har fokus istället legat på följning av vad som kallas för utsträckta mål, en typ av mål som rönt ökande uppmärksamhet i forskningsvärlden de senaste fem till tio åren. Med utsträckta mål får följningsproblemet en tredje del: att för varje enskilt mål räkna ut storleken och formen på målet, det vill säga den spatiala utsträckningen. Att känna till utsträckningen på de rörliga målen är viktigt exempelvis för en robot som ska ta sig genom ett rum där många person befinner sig. För att göra det krävs att roboten rör sig nära personerna, utan att för den skull krocka med någon. Att lösa detta på ett bra sätt kräver att roboten har kunskap inte bara om var personerna befinner sig, utan även hur mycket plats de tar upp.

I avhandlingen har ett antal aspekter av följning av utsträckta mål studerats. En viktig och komplicerande aspekt av följning av såväl punktmål, som utsträckta mål, är att roboten på förhand inte vet hur många mål som finns i dess närhet. En funktion för att hantera osäkerheterna kring antalet mål som finns, samt osäkerheterna kring var varje mål befinner sig, har implementerats.

I många situationer är det nödvändigt att kunna prediktera, eller förutsäga, var de olika målen kommer att befinna sig i den närmaste framtiden. Det kan exempelvis röra sig om en robot som ska köra genom en vägkorsning, och då måste undvika att krocka med övrig trafik. För detta ändamål har en prediktionsfunktion tagits fram.

När ett större antal mål rör sig i robotens närhet kan det bli svårt att följa varje enskilt mål. Istället kan roboten följa grupper av mål. Det blir då nödvändigt att hålla reda på vad som sker när mål lämnar gruppen, eller nya mål ansluter till gruppen. Fritt översatt från engelska till svenska kan dessa två händelser kallas för målproduktion och målkombination. Funktioner för att hantera produktion och kombination av utsträckta mål har tagits fram.

För att roboten ska kunna beräkna ett måls spatiala utsträckning krävs modeller för formen på målen. När laserscanners används kan formen på en bil sägas vara approximativt rektangulär, och formen på en person kan sägas vara approximativt elliptisk. Beräkning av storleken på rektangulära och elliptiska mål har studerats för robotar utrustade med laserscanners.

Målföljningsfunktionerna som nämnts ovan har utvärderats med hjälp av såväl simulerade data, som experimentella data insamlade med laserscanners. Resultaten visar att det arbete som har utförts jämför sig väl med tidigare arbete på området.

Acknowledgments

Whenever I get my hands on a thesis, there are two things that I always read: the dedication and the acknowledgments. There, in the middle of the objective, and often dry, academic writing lies the only chance to get a glimpse of the authors personality, a glimpse of the living and breathing human being behind the theories and the hypotheses, behind the theorems and the proofs. I shall therefore try to seize this opportunity, and make an attempt to show you my personality – especially my sense of humor – as I give my thanks to the persons to whom my thanks are due. However, please do not make yourself the illusion that I do not take this seriously. What is written below, is written with the utmost respect for all those that are mentioned.

With that said, in the interest of letting first things be first, I will start at the beginning. This thesis, and the research that it contains, is a product of my time at the Automatic Control group at Linköping University. If you should ever find yourself in Linköping, you may wish to visit the central library. Built in 2000, after a fire ravaged the old library in 1996, it features large glass windows that offer great views of Linköping Cathedral, and it boast in excess of 359604 book-titles. On one of the many shelves you can find Jack Kerouac's *On the road*, a story in which the main character claims to have

nothing to offer anybody except my own confusion.

While I would like to hope that I had something more than just my confusion to offer during those first months as a member of Automatic Control, I will gladly admit that starting a voyage towards a PhD-thesis indeed can be confusing.

In ironing out the worst kinks in my sheet of confusion, I received tremendous help from my supervisor Dr Thomas Schön. Thomas made sure that I understood the importance of structure, be it for a paper, a presentation, undergraduate teaching, or just for planning which graduate courses to take, and when to take them. He always served as a great source of enthusiasm and encouragement, and during the last parts of my PhD studies he was gracious enough to let me venture off in my own research direction.

Fortunately, on my continued path I found myself not alone, but under the guidance and patient support of Dr Umut Orguner. Had he not been there to straighten my steps, I probably would have

looked like a crab scurrying across the sand looking for bacon.

Our collaboration has been most fruitful, and it has been very inspiring for me to be part of it. I am forever indebted to Umut for all the knowledge and wisdom he has bestowed upon me. I have enjoyed all our discussions, and I truly hope that we will continue to write papers together.

If Thomas and Umut are to thank for helping me survive these four years, Dr Fabio Ramos and Dr Juan Nieto are to thank for making me believe that I could pull it all together. I may have left Sweden thinking that I was just going to write a

Master's thesis, but thanks to them I came back from Australia with my mind set on writing a PhD thesis. It has been a while since we last saw each other, but I hope our paths will cross again soon, so that we once again can enjoy the finer pleasures of academic life: the exquisite wines, the delicious barbeques, and the late night karaoke.

My days with Automatic Control would not have been, had it not been for the job offer I received from Prof Lennart Ljung and Prof Fredrik Gustafsson. I may have hesitated for a moment before accepting it, but once I joined the group I never looked back. Fredrik is the epicenter of the vibrant a bustling Sensor Fusion group, which I am proud to be part of. Lennart stood at the helm of Automatic Control for some thirty years, and managed to create a very impressive research environment before handing over the ship to Prof Svante Gunnarsson. Svante may have had big shoes to fill, but he filled them with an ease and elegance that is second to none. The only thing I could never quite understand is why he, towards the end of Mjärdevistafetten 2011, took that

fateful, pear shaped, left turn.

But then again, I am hardly in the position to question the judgment of a winner of the teaching award *Gyllene Moroten*. However, I am in the position to extend my sincere gratitude to Automatic Control's secretaries Ulla Salaneck, Åsa Karmelind and Ninna Stensgård, who have made sure that the administrative machinery has run without interruption.

This thesis was written using the thesis template constructed by TEX-gurus-at-large, Dr Gustaf Hendeby and Dr Henrik Tidefelt. Thanks to their meticulous attention to detail, the process of writing became smooth as silk. The thesis was proofread by Thomas, Umut, Fredrik, Dr Christian Lundquist and Lic Jonas Callmer, who all contributed with invaluable comments. Any and all remaining errors are – naturally – mine.

Jeffrey Bernard, a British *causeur* famous for the exhortation

aim low – and miss,

ended his days on earth by ingesting a toxic amount of bananas, an event he referred to as a *banana split*. In his obituary – written by none other than himself – he professed to having developed a fantasy that, starting tomorrow, things would finally take a turn for the better. Thinking that a geographical relocation would solve his problems, he longed for various dream cottages on the countryside. The experience was always ruined when he found himself living at the same location.

Personally, I have never longed for dream cottages – my grandmother already owns one on the Swedish west coast – but I have longed for dream colleagues. At Automatic Control in Linköping I found them, and I have enjoyed every day I have spent with them. I can only hope that their experience was not ruined when they found myself working at the same location.

Dr Christian Lundquist was kind enough to let me tag along as his PHD-train left

the station, a journey I have never regretted being part of. Our collaboration has always be inspiring to me, and I have always appreciated his constructive and invigorating critique of my work. It seems that lately the writing of two certain PhD theses has halted our forward motion. However, I am confident that we will pick up speed soon enough, provided that I do not stop completely to barrage him with another assortment of *sill*-related jokes.

One of my closest friends, Lic Jonas Callmer, has been along for the ride ever since we, in between spending time on Sydney's beaches and drinking free beer at The Gaff, managed to write a Master's thesis. I always enjoy our discussions, regardless of whether they are about serious politics, or about office related gossip. Alas, I never did understand his interest in getting fit by kicking other men in the balls, but then again, a great friendship is only strengthened by differences.

Lic *Morgan* Skoglund, *First Lord Protector of the Order of the Rävhjälm*, has always been there to remind me that it is time for coffee, or time for cross-country skiing, or time for our lunchtime 5K run, or simply time to head to the lab so that all the microphones can be hung – not like horses – but from the ceiling. No matter what we do, with Martin it always is, and always has been, a good time.

When I first started working at Automatic Control, there was this tall guy talking loudly in the *fika*-room. Often, as I walked back to my office after finishing my coffee, I could not help but wonder,

> *Wha' happen'!?!*

To tell you the truth, I still do not know, but I do know this: Lic Zoran Sjanic is not only tall and loud, he is also super-hilarious and fun, and above all he is a great friend. I can always trust Zoran when I want to have a bit of fun, regardless of whether we are in Linköping, on the Balkans, or in some godforsaken corner of the globe.

The *travelling-banana-salesman-and-monkey-dompteur*, Lic André Carvalho Bittencourt, was an awesome travel companion for two weeks in Taiwan. I feel very fortunate that I had the chance to get to know him, and I hope he is not too sad about the fact that I turned out to be the most handsome one of us.

If there is anyone you can trust when it is time to round up the troops for a pub crawl, it is Automatic Control's resident *BBQ-connoisseur*, Lic Sina Khoshfetrat Pakazad. He has a most infectious laughter, he always serves copious amounts of delicious meat, and he has absolutely despicable taste in movies.

As one goes through life, one meets an innumerable number of persons. Sometimes these encounters are like two ships, steaming past each other on a stormy ocean, to the sound of thunder and lightning. Other times the encounters are more like

> *två gistna ekor, guppandes förbi varandra på en försurad*
> *sjö i Småland, till ljudet av storlommens lockrop.*

The siren call of a black-throated loon?! What is it? Well, it's a chirp signal with logarithmically increasing frequency, but that's not important right now.

What is important, are all the friends that I have met on my path through life. This path has stretched over my childhood in Enebyberg, via a year as a foreign exchange student in Cottage Grove, to what has become a decade in Linköping, with a brief *caesura* for a years worth of surfing in New Zealand. You are far too many to mention, and far too important to forget. I hold the memories of the fun times we have had very dear, and I smile at the thought of all the fun that is yet to come.

Of all the gifts I received as a child, the finest one must be to have the best possible siblings I could have ever wanted. Together, Emma, Erik and I have made *kajsor av godisnappar*, fished for crab at the *campingbryggan*, *snorklat* in the *poolen*, and helped each other avoid temptation to

> *the greatest of the cardinal sins: landkrabberi.*

By all means, measures, metrics and standards, it has been very fun, and I'm certain it will continue to be so.

In a letter to Robert Hooke, Isaac Newton expressed that "If I have seen further it is by standing on ye sholders of Giants." Blessed with my grandfathers height, one might think that I do not need shoulders, or anything at all for that matter, to stand on. Nevertheless, my parents have provided me with all the shoulders I have ever needed to see far. With love and care they have instilled in me a feeling of being capable of anything that I set my mind to. They have supported me in wet and dry, they have celebrated my accomplishments, and they have accepted my mistakes. *För allt ni har gjort, och allt ni har betytt – tack!*

Last but not least, I wish to acknowledge the financial support that I have received from the Linnaeus research environment CADICS and the frame project grant Extended Target Tracking (621-2010-4301), both funded by the Swedish Research Council, and from the project Collaborative Unmanned Aircraft Systems (CUAS), funded by the Swedish Foundation for Strategic Research (SSF).

Linköping, October 2012
Karl Granström

Ladies and gentlemen!
The tall man with the glasses,
has now left the patio!

Contents

II Publications

Notation

Abbreviations

Abbreviation	Meaning
CPHD	Cardinalized Probability Hypothesis Density
CRF	Conditional Random Field
D	Detection
EKF	Extended Kalman Filter
ESDF	Exactly Sparse Delayed-state Filter
FA	False Alarm
FISST	Finite Set Statistics
GG	Gamma Gaussian
GIW	Gaussian inverse Wishart
GGIW	Gamma Gaussian inverse Wishart
GM-PHD	Gaussian Mixture Probability Hypothesis Density
GNN	Global Nearest Neighbor
GPS	Global Positioning System
ICP	Iterative Closest Point
IOU	Intersection Over Union
JPDA	Joint Probabilistic Data Association
MHT	Multi Hypothesis Tracking
NDT	Normal Distributions Transform
NEES	Normalized Estimation Error Square
NN	Nearest Neighbor
OSPA	Optimal Subpattern Assignment
PDA	Probabilistic Data Association
PHD	Probability Hypothesis Density
RFS	Random Finite Set
RMSE	Root Mean Square Error
ROC	Receiver Operating Characteristic
SLAM	Simultaneous Localization and Mapping

The laser range sensor

Notation	Meaning
$\mathbb{R}$	Set of real numbers
$\mathbf{p}_k$	Point cloud acquired at time t_k
p_i^k	Point i from $\mathbf{p}_k$
$\mathbf{s}$	Sensor position
$\mathbf{M}$	Environment
$\mathcal{T}_{\mathrm{p2c}}$	Transformation from polar to Cartesian coordinates
$\mathcal{T}_{\mathrm{s2c}}$	Transformation from spherical to Cartesian coordinates
x	Cartesian x-coordinate
y	Cartesian y-coordinate
z	Cartesian z-coordinate
r	Range
φ	Horizontal angle
ψ	Vertical angle
r_{max}	Maximum measurable range for laser range sensor
R	Rotation matrix
t	Translation vector

Classification

Notation	Meaning
$\mathcal{C}_{\mathrm{p}}$	Positive class
$\mathcal{C}_{\mathrm{n}}$	Negative class
$\mathbf{f}$	Data vector
f_j	Component j of data vector
y	Class label
N_{p}	Number of training data in positive class
N_{n}	Number of training data in negative class
$c(\cdot)$	Weak classifier
$\mathbf{c}(\cdot)$	Strong classifier
T	Number of training iterations
w_t^j	Weight of data j at iteration t
t_p	True positive rate
t_n	True negative rate
f_p	False positive rate
f_n	False negative rate
$N_{\mathrm{p}}^{\mathrm{t}}$	Number of test data in positive class
$N_{\mathrm{n}}^{\mathrm{t}}$	Number of test data in negative class
D	Detection rate
FA	False alarm rate

Estimation

Notation	Meaning
$\mathbf{x}(t)$, $\boldsymbol{\theta}(t)$, $\mathbf{z}(t)$, $\mathbf{w}(t)$, $\mathbf{e}(t)$	State, parameter, measurement, process noise, and measurement noise, respectively, at time t
$\mathbf{x}_k$, $\boldsymbol{\theta}_k$, $\mathbf{z}_k$, $\mathbf{w}_k$, $\mathbf{e}_k$	State, parameter, measurement, process noise, and measurement noise, respectively, at time step k
$\mathbf{z}^k$	Set of all measurements up to, and including, time step k
T_s	Sampling time
$\hat{\mathbf{x}}_{k\|\ell}$, $\mathbf{P}_{t\|\ell}$	State estimate and covariance at time t_k given measurements up to and including time t_ℓ
$\mathcal{N}(\mathbf{x}\,;\,\mu,\Sigma)$	Gaussian pdf defined over the random vector $\mathbf{x}$ with mean vector μ and covariance matrix Σ
$\mathbf{Q}$, $\mathbf{R}$	Process noise covariance and measurement noise covariance
$p\left(\mathbf{x}_k \| \mathbf{z}^\ell\right)$	Probability density over $\mathbf{x}_k$ given $\mathbf{z}^\ell$
$\xrightarrow{\text{c}}$, $\xrightarrow{\text{p}}$	Correction and prediction, i.e. measurement update and time update
$\mathcal{E}$	Estimation error
η	Normalized estimation error square
ρ	Root mean square error
${}^r\mathbf{x}_k$	Robot pose k
${}^r\mathbf{x}_c$	Current robot pose
${}^t\mathbf{x}$	Robot trajectory, i.e. history of poses
$\mathbf{M}$	Map state, i.e. environment
$\mathbf{m}_i$	Landmark i

Target tracking

Notation	Meaning
$\mathbf{x}_k^{(i)}$	State of target i at time k
$N_{x,k}$	Number of targets at time k
$\mathbf{X}_k$	Set of present targets at time k
$\mathbf{z}_k^{(i)}$	Measurement i at time k
$N_{z,k}$	Number of measurements at time k
$\mathbf{Z}_k$	Set of measurements at time k
$\mathbf{Z}^k$	Set of all measurement sets up to time k
$\mathcal{H}_i$	Hypothesis i
$\mathcal{P}(\cdot)$	Probability
$\bar{d}_p^c(\cdot\,,\,\cdot)$	OSPA metric of order p with cut-off c

Random finite sets and the probability hypothesis density

Notation	Meaning
Ξ	Random finite set variable
$\mathbf{X}$	Finite set realization of Ξ
$\lvert\mathbf{X}\rvert$	Number of elements in set $\mathbf{X}$
$\mathcal{X}$	Hyperspace of underlying space $\mathcal{X}_0$
$P_{\mathbf{x}} S$	Probability mass function of random variable $\mathbf{x}$
$p_{\mathbf{x}}(x)$	Probability density function of random variable $\mathbf{x}$
$\beta_{\Xi}(S)$	Belief-mass function of random set Ξ
$p_{\Xi}(\mathbf{X})$	Probability density function of random set Ξ
$p_{\Xi}(n)$	Cardinality distribution of random set Ξ
$\mathrm{E}[\,\cdot\,]$	Expected value
$D_{k\mid\ell}(\mathbf{x})$	Probability hypothesis density
$D^{\mathrm{b}}_{k+1\mid k}(\mathbf{x})$	Birth PHD
$p_{\mathrm{S}}(\mathbf{x})$	Probability of survival
$p_{k+1\mid k}(\mathbf{x}\mid\mathbf{x}')$	Transition density
$p^{\mathrm{s}}_{k+1\mid k}(\mathbf{x}\mid\mathbf{x}')$	Spawning likelihood
$p_{\mathrm{D}}(\mathbf{x})$	Probability of detection
$p_{k+1}(\mathbf{z}\mid\mathbf{x})$	Sensor likelihood function
$c(\mathbf{z})$	Distribution of false alarms
$P_{k\mid\ell}(n)$	Estimated cardinality distribution

Extended target tracking

Notation	Meaning
ξ_k	Extended target state at time k
$\mathbb{R}^n$	Set of real vectors of length n
$\mathbb{S}^d_{++}$	Set of symmetric positive definite $d \times d$ matrices
$\mathbf{x}_k$	Random vector at time k
X_k	Random matrix at time k
γ_k	Random measurement rate at time k
$\mathcal{PS}(N;\ \lambda)$	Poisson pmf defined over the random non-negative integer N with rate parameter λ
$\mathcal{IW}(X\,;\ v, V)$	Inverse Wishart pdf defined over the random matrix X with degrees of freedom v and parameter matrix V
$\mathcal{GAM}(\gamma\,;\ \alpha, \beta)$	Gamma pdf defined over scalar γ with scalar shape parameter α and scalar inverse scale parameter β

Part I

Background

1 Introduction

This chapter introduces the research topics that are considered in this thesis, and summarizes the research contributions. In Section 1.1 a motivation to the research topics is given, and in Section 1.2 and Section 1.3 the topics are described in more detail. A list of published work is given in Section 1.4, and the main contributions of the thesis are summarized in section 1.5. The chapter is ended with a thesis outline in Section 1.6.

1.1 Motivation

The research presented in this thesis was undertaken at the Division of Automatic Control, Department of Electrical Engineering, at Linköping university. Automatic control is a research area that can be given the following definition, deliberately intended to be as broad as possible:

Definition 1.1 (Automatic control). To automatically make a system behave as desired.

In this context *automatically* is to be understood as *without human intervention*. *A system* may refer to anything whose behavior can be controlled, however this thesis will be limited to mobile robots, also called autonomous vehicles.

In the January 2007 issue of the magazine Scientific American, Bill Gates, co-founder and former CEO of Microsoft, predicted that the next hot research field would be robotics (Gates, 2007). About four years later, a free online course in artificial intelligence, given by Stanford University during the fall semester of 2011, attracted more than 58000 students globally (Markoff, 2011). Both these examples serve as a testament to the interest in, and relevance of, robotics research.

Indeed, the past decades have seen a large research effort in the field of robotics. In order for a mobile robot to *behave as desired* in the dynamic and complex world within which humans live, the robot must be aware of itself and its surroundings. In this thesis we refer to this as knowing the state of the robot and the state of its surroundings. The state of the robot includes its location, knowledge of which requires the ability to recognize places that the robot has visited earlier, also called loop detection. The state of the robots' surroundings includes the location of moving objects, called targets, knowledge of which requires the ability to track the targets as they move.

The main research topics considered in this thesis are

1. loop closure detection, i.e. recognizing places that have been visited before;
2. multiple target tracking, i.e. estimating how many targets there are and estimating each target's state.

To solve both these problem, the robot needs to sense the environment around it, similarly to how humans use their five senses[1] to be able to go about their days. In this thesis data from laser range sensors is used for both loop closure detection and target tracking. A reconstruction of 2317 2D laser range scans, acquired inside a shopping mall, is shown in Figure 1.1. Figure 1.2, a small portion of a full scene constructed by 34 3D laser range scans, shows an outdoor environment with some buildings and vegetation.

Laser range sensors are versatile sensors that provide data rich in information content, and the sensor data can be used for many different tasks. In the Defense Advanced Research Projects Agency's (DARPA) Urban Challenge, e.g., laser range sensors were used for task such as staying in lane; maintain vehicle separation; vehicles at an intersection; leaving lane to pass; U-turn; following a vehicle; queuing at an intersection; negotiate obstacle field; road blockages; merging to traffic circle; sparse waypoints (straight); road follow: GPS outages; merging at T intersection; merging at 4-way intersection; left turns at intersections; emergency vehicle avoid; and blocked intersection (Campbell et al., 2007).

1.2 Loop closure detection

Loop closure detection is an important part of the Simultaneous Localization and Mapping (SLAM) problem. The SLAM problem consists of finding out where the robot is (localization), while simultaneously finding out what the surrounding environment looks like (mapping), see e.g. the two part SLAM tutorial by Durrant-Whyte and Bailey (2006) and Bailey and Durrant-Whyte (2006). To solve the SLAM problem the acquired sensor data must be organized such that, when the individual pieces of data are put together, they together constitute a coherent map. However, one of the fundamental properties of the SLAM problem is that small errors, due to sensor inaccuracies, are constantly inserted into the localiza-

[1] Vision, hearing, smell, taste, and touch.

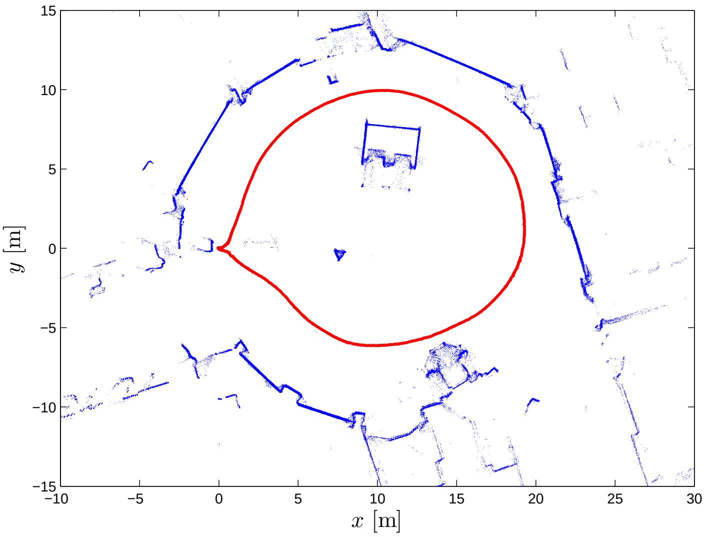


Figure 1.1: *Laser range data in 2D, from the ground floor of the shopping mall "Gränden" in central Linköping, Sweden. The laser range data is shown in blue, the robot trajectory is shown in red. Data courtesy of Petter Torle, C3 Technologies.*

tion and mapping process. As time passes the errors accumulate, and eventually the map is no longer a good representation of the world. One method to correct this problem is to detect when the robot returns to a place that it has previously visited, i.e. detect that the robot has closed a loop. Because loop closure detection is used to correct errors in the SLAM process, it is of high importance that incorrect, or false, loops are not closed, because this would only increase the errors.

In this thesis the SLAM map consists of individual laser range scans, so called point clouds, acquired at different locations. Loop closure is detected by comparing different point clouds to each other in a pairwise fashion, and classifiying them as either being from the same location, or not.

1.3 Multiple target tracking

Multiple target tracking is needed for the robot to be able to move around without constantly running into other objects, and also to make the robot able to follow a moving object while the object moves. The research area target tracking dates back at least to the mid 1900's, when radar stations were built for the purpose of tracking airplanes, see e.g. the books by Bar-Shalom and Fortmann (1987), Bar-Shalom (1992), Bar-Shalom and Rong Li (1995), Bar-Shalom et al. (2001), and Bar-Shalom et al. (2011). In the typical target tracking scenario it is unknown how many targets there are, it is unknown which target caused which measurement,

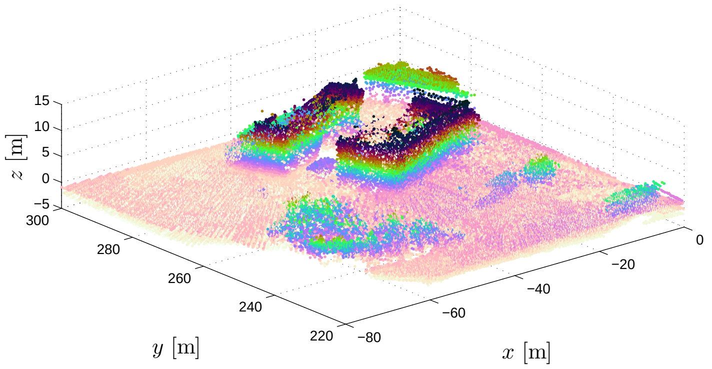


Figure 1.2: *Laser range data in 3D, only a portion of the full data set is shown. The scene features a couple of buildings and some vegetation. The data was acquired using laser range sensors mounted on a helicopter. Data courtesy of Piotr Rudol and Mariusz Wzorek, at the Knowledge Processing lab (KPLAB) at the division of Artificial Intelligence and Integrated Computer Systems (AIICS) at the Department of Computer and Information Science (IDA) at Linköping university (LiU).*

it is unknown if all targets caused any measurement at all, and there are false, so called clutter, measurements that were not caused by any target at all. In some target tracking scenarios, the target and sensor setup is such that each target generates at most one measurement per time step, in other scenarios each target may generate more than one measurement. In the former case the targets are called point targets, in the latter case the targets are called extended targets.

In this thesis we consider tracking of multiple extended targets, where both the number of targets and each target's state must be found.

1.4 Publications

The following papers, listed in reverse chronological order, have been published:

K. Granström, C. Lundquist, and U. Orguner. Extended Target Tracking using a Gaussian Mixture PHD filter. *IEEE Transactions on Aerospace and Electronic Systems*, 2012.

K. Granström and U. Orguner. A PHD filter for tracking multiple extended targets using random matrices. *IEEE Transactions on Signal Processing*, 2012a. doi: 10.1109/TSP.2012.2212888.

K. Granström and U. Orguner. On the Reduction of Gaussian inverse Wishart mixtures. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 2162–2169, Singapore, July 2012d.

K. Granström and U. Orguner. Estimation and Maintenance of Measurement Rates for Multiple Extended Target Tracking. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 2170–2176, Singapore, July 2012c.

K. Granström, C. Lundquist, F. Gustafsson, and U. Orguner. On extended target tracking using PHD filters. In *Workshop on Stochastic Geometry in SLAM at IEEE International Conference on Robotics and Automation (ICRA)*, St. Paul, Minnesota, USA, May 2012.

K. Granström, T. B. Schön, J. I. Nieto, and F. T. Ramos. Learning to close loops from range data. *The International Journal of Robotics Research*, 30(14):1728–1754, December 2011.

U. Orguner, C. Lundquist, and K. Granström. Extended Target Tracking with a Cardinalized Probability Hypothesis Density Filter. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 65–72, Chicago, IL, USA, July 2011.

C. Lundquist, K. Granström, and U. Orguner. Estimating the shape of targets with a PHD filter. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 49–56, Chicago, IL, USA, July 2011a.

K. Granström, C. Lundquist, and U. Orguner. Tracking Rectangular and Elliptical Extended Targets Using Laser Measurements. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 592–599, Chicago, IL, USA, July 2011.

K. Granström and T. B. Schön. Learning to Close the Loop from 3D Point Clouds. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2089–2095, Taipei, Taiwan, October 2010.

K. Granström, C. Lundquist, and U. Orguner. A Gaussian Mixture PHD filter for Extended Target Tracking. In *Proceedings of the International Conference on Information Fusion (FUSION)*, Edinburgh, UK, July 2010.

K. Granström, J. Callmer, F. T. Ramos, and J. I. Nieto. Learning to Detect Loop Closure from Range Data. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22, Kobe, Japan, May 2009.

J. Callmer, K. Granström, J. I. Nieto, and F. T. Ramos. Tree of Words for Visual Loop Closure Detection in Urban SLAM. In *Proceedings of the Australian Conference on Robotics & Automation (ACRA)*, Canberra, Australia, December 2008.

The following paper has been provisionally accepted for publication:

> K. Granström and U. Orguner. On Spawning and Combination of Extended/Group Targets Modeled with Random Matrices. *IEEE Transactions on Signal Processing*, 2012e.

The following paper has been revised and resubmitted:

> K. Granström and U. Orguner. A New Prediction Update for Extended Target Tracking with Random Matrices. *IEEE Transactions on Aerospace and Electronic Systems*, 2012b.

The following paper has been submitted:

> C. Lundquist, K. Granström, and U. Orguner. An extended target CPHD filter and a gamma Gaussian inverse Wishart implementation. *Journal of Selected Topics in Signal Processing*, 2012a.

Finally, the following paper about pedagogic and didactic aspects of undergraduate teaching, has also been published:

> C. Lundquist, M. A. Skoglund, K. Granström, and T. Glad. Insights from implementing a system for peer review. *IEEE Transactions on Education*, 2012b. doi: 10.1109/TE.2012.2211876.

1.5 Main contributions

The second part of this thesis contains edited versions of eight of the above listed papers. The scientific contributions contained in these eight papers are summarized in this section.

1.5.1 Loop closure detection

Learning to close loops from range data

Paper A,

> K. Granström, T. B. Schön, J. I. Nieto, and F. T. Ramos. Learning to close loops from range data. *The International Journal of Robotics Research*, 30(14):1728–1754, December 2011.

presents a loop closure detection classifier that works for point cloud data in both 2D and 3D. A thorough implementational description is given, and the classifier's properties are evaluated in several different experiments using publicly available data. The pros and cons compared to related work are discussed, and the classifier is shown to compare well to other loop closure detection methods.

1.5.2 Target tracking

Extended Target Tracking using a Gaussian Mixture PHD filter

Paper B,

> K. Granström, C. Lundquist, and U. Orguner. Extended Target Tracking using a Gaussian Mixture PHD filter. *IEEE Transactions on Aerospace and Electronic Systems*, 2012.

presents a Gaussian mixture implementation of an extended target PHD filter. The optimal filter requires a summation over all possible measurement set partitions, which is computationally infeasible in all but the simplest of cases. Suitable partitioning methods are presented, such that the number of partitions that are considered can be kept to a minimum without sacrificing too much tracking performance. The filter is evaluated in both simulations and experiments.

Tracking Rectangular and Elliptical Extended Targets Using Laser Measurements

To keep the presentation simple, the sizes and shapes of the extended targets are not estimated in Paper B. Paper C,

> K. Granström, C. Lundquist, and U. Orguner. Tracking Rectangular and Elliptical Extended Targets Using Laser Measurements. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 592–599, Chicago, IL, USA, July 2011.

presents a version of the Gaussian mixture PHD filter that is designed for laser range measurement, with capability to estimate the shape and size of the targets. Two different types of targets are considered, rectangular and elliptical, and the filter is also capable of estimating the target type. Furthermore, the paper shows that the Gaussian mixture PHD filter is not limited to linear motion and measurement models, as in Paper B, it also works for non-linear models. The filter is evaluated in both simulations and experiments.

A PHD filter for tracking multiple extended targets using random matrices

In Paper B and Paper C Gaussian distributions are used to model the extended targets. An alternative to the Gaussian model is to use Gaussian inverse Wishart distributions to model the extended targets, a model in which the extended target shape is assumed to be elliptical. Paper D,

> K. Granström and U. Orguner. A PHD filter for tracking multiple extended targets using random matrices. *IEEE Transactions on Signal Processing*, 2012a. doi: 10.1109/TSP.2012.2212888.

presents a Gaussian inverse Wishart implementation of the extended target PHD filter. A likelihood function is derived, and the necessary assumptions and approximations are given. Two partitioning methods are presented, in addition to the methods given in Paper B. The filter is evaluated in both simulations and experiments, and the results show the benefits of estimating the target size and shape, in addition to estimating the position.

Estimation and Maintenance of Measurement Rates for Multiple Extended Target Tracking

The extended target PHD filters in Paper B, Paper C and Paper D model the number of measurements generated by an extended target as Poisson distributed. In Paper B and Paper D it is noted that correctly setting the filter parameter corresponding to the Poisson rate is necessary in certain circumstances. With an incorrect parameter setting, the filter might estimate the number of targets incorrectly. Paper E,

> K. Granström and U. Orguner. Estimation and Maintenance of Measurement Rates for Multiple Extended Target Tracking. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 2170–2176, Singapore, July 2012c.

presents a framework for estimating an individual, possibly time varying, Poisson rate for each extended target. In addition to the already known measurement update, a simple time update is suggested, and a method for mixture reduction is also given. Simulations show that the filter can estimate multiple Poisson rates simultaneously.

On the Reduction of Gaussian inverse Wishart Mixtures

In Paper D a heuristic is used for reduction of Gaussian inverse Wishart mixtures, however it is noted that a better and less approximative method is needed. Paper F,

> K. Granström and U. Orguner. On the Reduction of Gaussian inverse Wishart mixtures. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 2162–2169, Singapore, July 2012d.

presents a merging method that can be used for reduction of Gaussian inverse Wishart distribution mixtures. It is shown how a weighted sum of distributions can be approximated with a single distribution, and a criterion is suggested that can be used to determine whether or not two distributions should be merged. Simulations show the merits of the presented merging method.

A New Prediction Update for Extended Target Tracking with Random Matrices

In Paper D it is noted that when two spatially close targets maneuver, the filter often cannot keep the two targets resolved, and as a consequence underestimates the number of targets. Part of the problem is the heuristic prediction method that is used. Paper G,

> K. Granström and U. Orguner. A New Prediction Update for Extended Target Tracking with Random Matrices. *IEEE Transactions on Aerospace and Electronic Systems*, 2012b.

presents a prediction method that can handle maneuvering extended targets better, while considering all uncertainty sources. Simulation results show that the presented method outperforms related work on this subject.

On Spawning and Combination of Extended/Group Targets Modeled with Random Matrices

The event that a target launches another target, or that a larger target separates into multiple smaller targets, is called target spawning. The opposite, i.e. that multiple targets merge into a single target, is called target combination. Target spawning is not explicitly modeled in Paper D, however it is noted there that a spawning function could be useful. Paper H,

> K. Granström and U. Orguner. On Spawning and Combination of Extended/Group Targets Modeled with Random Matrices. *IEEE Transactions on Signal Processing*, 2012e.

presents spawning and combination of extended targets whose state are modeled as Gaussian inverse Wishart distributed. Limited to the two target case, a model for target combination is first derived. The combination model is then used to derive a model for spawning into two targets. Simulation results show the benefits of the presented functions.

1.6 Thesis outline

The thesis is divided into two parts, with background material in the first part and edited versions of the eight published papers in the second part. It should be noted that while the chapters in the first part give relevant background required for the second part of the thesis, the amount of detail is intentionally kept to a minimum. The reason is that each of the papers in the second part includes background material – repeating this material would cause unnecessary redundancy.

The first part of the thesis is organized as follows. Chapter 2 presents the laser range sensor and the data it produces. Classification, with an emphasis on so called boosting, is the topic of Chapter 3. The machine learning method used in this thesis, called AdaBoost, is presented and illustrated using a number of examples. The estimation problem is introduced in Chapter 4, and filtering solutions and performance metrics are mentioned. Chapter 5 is about the target tracking problem. Data association methods are over-viewed, and performance evaluation is discussed. Random finite sets and the probability hypothesis density filter is introduced in Chapter 6, and extended target tracking is the topic of Chapter 7. The first part of the thesis is ended with Chapter 8, which presents conclusions and discusses future work.

2 The laser range sensor

This chapter presents a brief overview of the laser range sensor and the data it produces. The sensor is described in Section 2.1, and examples of 2D and 3D data are given in Section 2.2 and Section 2.3, respectively. The occlusion problem is described in Section 2.4, and registration of laser range data is discussed in Section 2.5.

2.1 Introduction

In the past 10 year, a vast amount of research has been performed using data from laser range sensors, e.g. mapping, localization and target tracking. There exist different types of laser sensors that produce slightly different types of data, this thesis will be limited to so called sweeping laser sensors. This sensor type works by measuring the distance to the nearest object at different angles, provided that the nearest objects' reflectance properties are good enough. A simulation example of laser range data is given in Figure 2.1.

For an estimation or classification application, the laser range sensor's probabilistic properties need to be modeled. In this thesis, a brief introduction to modeling of the laser range sensor is given. For a more thorough description, chapter 6 in the book by Thrun et al. (2005) is a good starting point. A description of the underlying mechanical and electrical properties goes beyond the scope of this thesis.

In the remainder of this thesis, the output from laser range sensors will be referred to as point clouds, with the following definition:

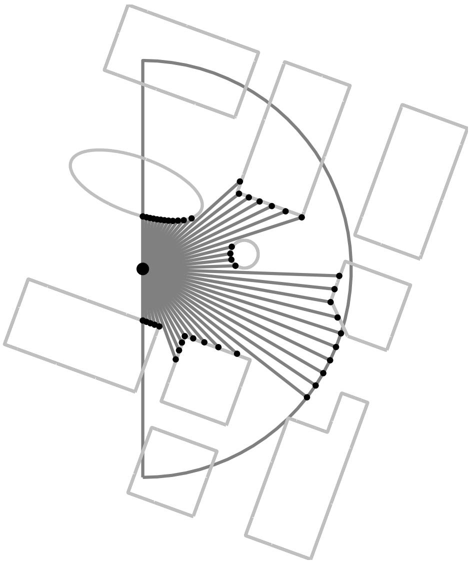


Figure 2.1: *Example of laser range data in a 2D simulation environment, where the objects are shown in light gray. The sensor is located at the large black dot, the sensor field of view (180° wide) is shown by the semi-circle. The sensor sweeps right to left, and measures the nearest object every fourth degree. When the nearest object is further away than the boundary of the field of view, the sensor returns maximum range.*

Definition 2.1 (Point cloud $\mathbf{p}_k$). A collection of points in space,

$$\mathbf{p}_k = \left\{p_i^k\right\}_{i=1}^{N}, \quad p_i^k \in \mathbb{R}^{\mathsf{D}}. \tag{2.1}$$

Here, k refers to the acquisition time t_k, N is the number of points p_i^k in the cloud and D is the dimensionality of the data.

The name point cloud is inherited from the fact that the sensor measurement defines a point in space which is occupied by an object. It should be noted though that the name point cloud does not capture the so called negative information, i.e. the information about the free space along the laser measurement ray[1]. In an application, this negative information about the free-space is important to consider along with the points themselves.

In the applications presented in this thesis, the dimensionality of the data is either D = 2 or D = 3. Many sensors however, in addition to measuring range, also measure the remission value of the measured point. If the laser range data is fused with image data from a camera, each point may also contain RGB color values. Thus the dimensionality D of the data could be larger than 3. Each measured

[1] cf. the gray rays from the sensor to the measurement points in Figure 2.1

(a) (b) (c)

Figure 2.2: *Examples of laser range sensors. (a): the SICK LMS200-series. (b): the Hokuyo URG-04Lx-series. (c): the Velodyne HDL-64E. Images are from* `www.sick.com`, `www.hokuyo-aut.jp` *and* `www.velodyne.com/lidar/`, *respectively.*

point p is a function $\mathcal{L}$ of the sensor position $\mathbf{s}$ and the surrounding environment $\mathbf{M}$,

$$p = \mathcal{L}\left(\mathbf{s}, \mathbf{M}, \mathbf{e}_p\right), \tag{2.2}$$

where $\mathbf{e}_p$ is random noise. Typically $\mathbf{e}_p$ is modeled as a Gaussian with zero mean and covariance Σ_p,

$$\mathcal{N}\left(\mathbf{e}_p\,;\,\mathbf{0}, \Sigma_p\right). \tag{2.3}$$

The properties of laser range sensors vary substantially from sensor to sensor. Maximum measurable range $r_{\max}$ varies from several meters to several kilometers, angular resolution varies from being in the order of one degree to the order of one thousand of a degree. Examples of popular sensors are the LMS200-series sensors manufactured by SICK, see Figure 2.2a, and the sensors manufactured by Hokuyo, see Figure 2.2b. Both these sensors produces planar laser range scans, i.e. they sense the surrounding environment in 2D. Using different pan/tilt units, several 2D scans can be combined to provide 3D laser range data. There are also dedicated 3D laser range sensors, e.g. the HDL-series sensors from Velodyne, see Figure 2.2c.

2.2 Laser range data in 2D

In 2D the points in the point cloud are typically given in polar coordinates as

$$p_i^k = \begin{bmatrix} r_i^k & \varphi_i^k \end{bmatrix}^{\mathrm{T}}, \tag{2.4}$$

where r is the range and φ is the horizontal angle, or bearing, to the measured point. Using the polar to Cartesian transformation

$$\begin{bmatrix} \mathrm{x} \\ \mathrm{y} \end{bmatrix} = \mathcal{T}_{\mathrm{p2c}}\left(r, \varphi\right) = \begin{bmatrix} r\cos\left(\varphi\right) \\ r\sin\left(\varphi\right) \end{bmatrix}, \tag{2.5}$$

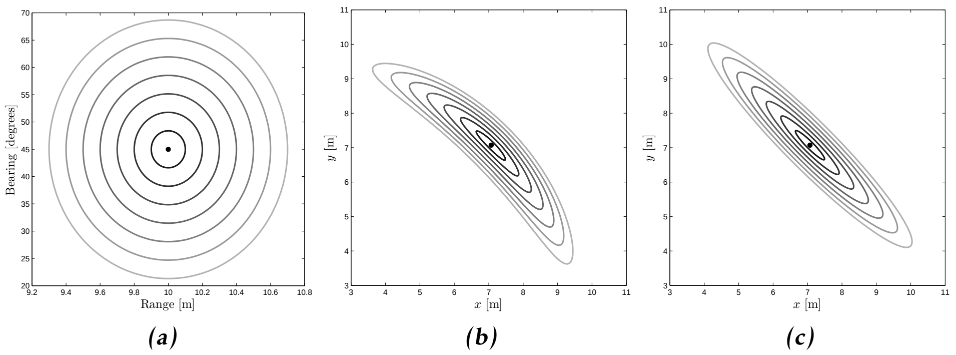


Figure 2.3: *Illustration of laser point uncertainty. (a): Uncertainty in polar coordinates. (b): The same uncertainty ellipses as in (a), transformed to Cartesian coordinates using the transform* (2.5)*. (c): First order approximation of transformed uncertainty in (b) using* (2.9)*.*

the points can be expressed in Cartesian coordinates as

$$p_i^k = \begin{bmatrix} \mathrm{x}_i^k & \mathrm{y}_i^k \end{bmatrix}^{\mathrm{T}}. \tag{2.6}$$

The measurement noise covariance matrix is typically a diagonal matrix, where the range and bearing standard deviations can be modeled as functions of range and bearing,

$$\Sigma_p = \begin{bmatrix} \sigma_r^2(r,\varphi) & 0 \\ 0 & \sigma_\varphi^2(r,\varphi) \end{bmatrix}. \tag{2.7}$$

Using the Jacobian J_{p2c} of the polar to Cartesian transformation (2.5),

$$J_{\mathrm{p2c}} = \begin{bmatrix} \cos(\varphi) & -r\sin(\varphi) \\ \sin(\varphi) & r\cos(\varphi) \end{bmatrix} \tag{2.8}$$

the covariance can be approximated to first order in Cartesian coordinates as

$$\Sigma_p^{\mathrm{c}} = J_{\mathrm{p2c}} \Sigma_p J_{\mathrm{p2c}}^{\mathrm{T}}. \tag{2.9}$$

An illustration of the modeled uncertainty is shown in Figure 2.3, where a measurement at range $r = 10$m and bearing $\varphi = 45°$, and its corresponding uncertainty, are shown in Figure 2.3a. The results for the non-linear transformation from polar to Cartesian coordinates (2.5) is shown in Figure 2.3b. Compare with the first order approximation in Figure 2.3c.

Figure 2.4 shows a typical outdoor point cloud, both in polar and Cartesian coordinates, Figure 2.4a and Figure 2.4b, respectively. Typical indoor data was shown in Figure 1.1 in Chapter 1. The figures also feature the corresponding 3σ covariance ellipses for every tenth point. For this example, the measurement noise

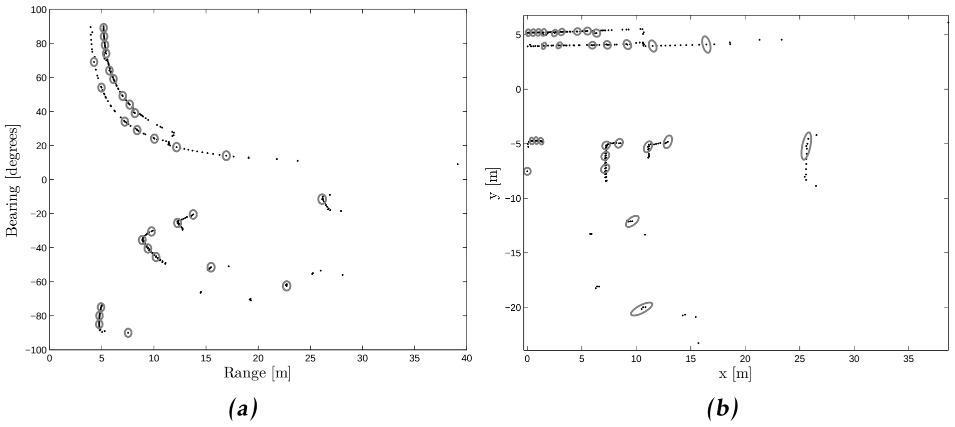


Figure 2.4: *Example 2D point cloud, acquired in an outdoor environment. (a): Polar coordinates. (b): Cartesian coordinates. All points measuring range above the maximum range $r_{max} = 50\text{m}$ have been filtered out. The points are shown in black, for every tenth point the corresponding 3σ covariance ellipse is shown in gray.*

covariance is modeled as

$$\Sigma_p = \begin{bmatrix} \sigma_r^2(r,\varphi) & 0 \\ 0 & \sigma_\varphi^2(r,\varphi) \end{bmatrix} = \begin{bmatrix} \left(0.01\left(1+\frac{r}{r_{\max}}\right)\right)^2 & 0 \\ 0 & \left(\frac{0.01\pi}{180}\left(1+\frac{r}{r_{\max}}\right)\right)^2 \end{bmatrix}. \quad (2.10)$$

Thus the uncertainty in range grows larger as the distance to the nearest object along the laser ray grows larger. The bearing noise model can be understood as modeling the laser ray as being shaped as a triangle, with the tip located at the sensor. Thus, the measured point is located on the bottom edge of the triangle.

2.3 Laser range data in 3D

In 3D the points in the point cloud are given in spherical coordinates as

$$p_i^k = \begin{bmatrix} r_i^k & \varphi_i^k & \psi_i^k \end{bmatrix}^{\mathrm{T}}, \quad (2.11)$$

where r is the range, φ is the horizontal angle and ψ is the vertical angle to the measured point. Using the spherical to Cartesian transformation

$$\begin{bmatrix} \mathsf{x} \\ \mathsf{y} \\ \mathsf{z} \end{bmatrix} = \mathcal{T}_{\mathsf{s2c}}(r,\varphi,\psi) = \begin{bmatrix} r\sin(\psi)\cos(\varphi) \\ r\sin(\psi)\sin(\varphi) \\ r\cos(\psi) \end{bmatrix}, \quad (2.12)$$

the points can be expressed in Cartesian coordinates as

$$p_i^k = \begin{bmatrix} \mathsf{x}_i^k & \mathsf{y}_i^k & \mathsf{z}_i^k \end{bmatrix}^{\mathrm{T}}. \quad (2.13)$$

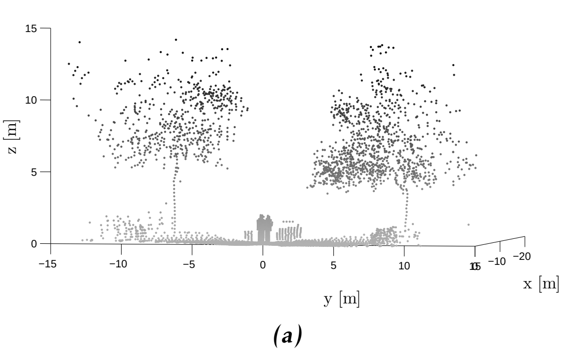
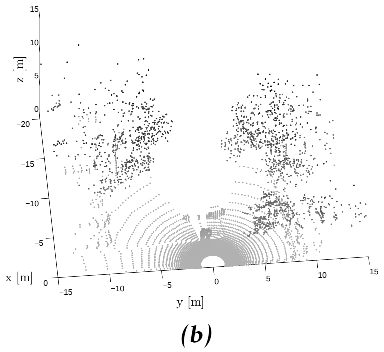


Figure 2.5: *Example 3D point cloud in Cartesian coordinates, shown from two different view points in (a) and (b). Grey-scale is used to accentuate height. In the center of the point cloud there are two persons and behind them a car can be seen. To the left and right of the persons and the car there are two trees.*



The measurement noise covariance matrix is typically a diagonal matrix, where the range, horizontal angle and vertical angle standard deviations can be modeled as functions of range, horizontal angle and vertical angle,

$$\Sigma_p = \begin{bmatrix} \sigma_r^2(r,\varphi,\psi) & 0 & 0 \\ 0 & \sigma_\varphi^2(r,\varphi,\psi) & 0 \\ 0 & 0 & \sigma_\psi^2(r,\varphi,\psi) \end{bmatrix}. \tag{2.14}$$

Using the Jacobian J_{s2c} of the spherical to Cartesian transformation (2.12),

$$J_{\mathrm{s2c}} = \begin{bmatrix} \sin(\psi)\cos(\varphi) & -r\sin(\psi)\sin(\varphi) & r\cos(\psi)\cos(\varphi) \\ \sin(\psi)\sin(\varphi) & r\sin(\psi)\cos(\varphi) & r\cos(\psi)\sin(\varphi) \\ \cos(\psi) & 0 & -r\sin(\psi) \end{bmatrix} \tag{2.15}$$

the covariance can be approximated to first order in Cartesian coordinates as

$$\Sigma_p^{\mathrm{c}} = J_{\mathrm{s2c}}\Sigma_p J_{\mathrm{s2c}}^{\mathrm{T}}. \tag{2.16}$$

An example point cloud is shown in Figure 2.5. In the figure, gray-scale is used to accentuate height. A 3D point cloud was also shown in Figure 1.2 in Chapter 1.

2.4 Occlusion

Similarly to how a camera needs direct line of sight to the object that is being sensed, so does a laser range sensor. Thus, if an object A is located at the same bearing as another object B, but at larger range than B, then A is occluded by B. Depending on the shape and size of the two objects, A is either partially of fully occluded by B. An example of occlusion is given in Figure 2.6. Occlusion presents

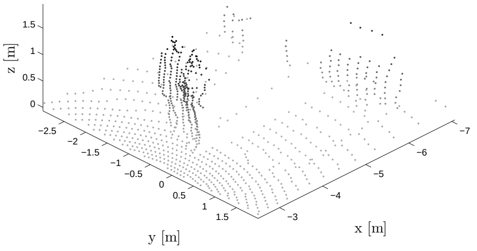


Figure 2.6: *Example of the occlusion problem for laser range sensors. The 3D point cloud shows two persons in the foreground, and behind them is a partially occluded vehicle. This point cloud is a portion of the point cloud shown in Figure 2.5. Grey-scale is used to accentuate height, however the gray-scale is different from Figure 2.5.*

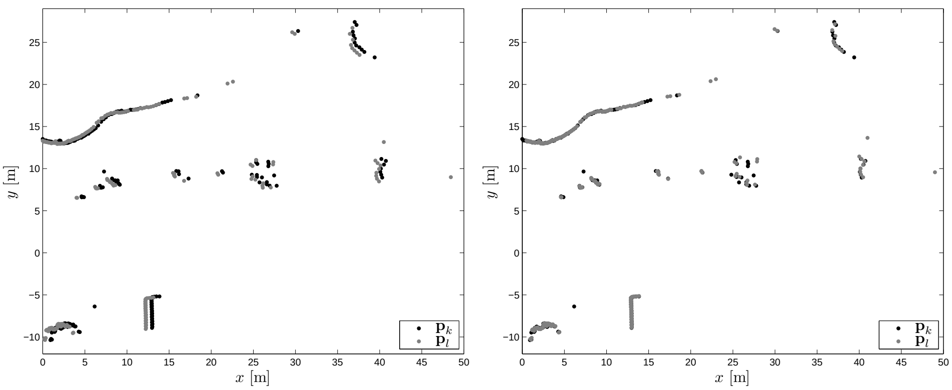
a considerable challenge in estimation and classification problems where laser range data is used. In the context of target tracking, targets may be lost when they move behind other objects and thus do not generate any measurements. For loop closure detection, occlusions by dynamic objects means that the appearance of the point cloud can be significantly changed.

2.5 Registration

Registration is the process by which two point clouds $\mathbf{p}_k$ and $\mathbf{p}_l$ are fitted to each other with respect to some measure, or cost function, $\mathbf{C}(\mathbf{p}_k, \mathbf{p}_l)$. Typically, the problem is solved by finding a rigid body transformation (R, t), where R is a rotation matrix and t is a translation vector, such that the sum of distances between different point correspondences in the two point clouds is minimized. Point cloud registration is in the literature also referred to as scan matching. Several different methods for finding this rigid body transformation have been suggested, among them the (probably) most popular and well used is the so called Iterative Closest Point (ICP) algorithm (Besl and McKay, 1992; Chen and Medioni, 1992; Zhang, 1994). ICP works by solving the following optimization problem

$$\min_{(R,\mathrm{t})} \mathbf{C}(\mathbf{p}_k, \mathbf{p}_l) = \min_{(R,\mathrm{t})} \sum_{i=1}^{N_k} \sum_{j=1}^{N_l} w_{i,j} \left\| p_i^k - \left(R p_j^l + \mathrm{t} \right) \right\|^2, \tag{2.17}$$

where $w_{i,j}$ is 1 if point p_i^k and point p_j^l describe the same point in space, and 0 otherwise. Finding these point correspondences is typically performed by a nearest neighbor search, and a solution (R, t) is found by iterating between finding nearest neighbor point pairs and computing the corresponding rigid body transformation. The cost function in (2.17) has many local minimas, and the ICP



(a) *Before* ICP. ***(b)*** *After* ICP.

Figure 2.7: *Point cloud registration using the* ICP *algorithm. (a): two point clouds from an outdoor environment before the* ICP *algorithm is applied. (b): after the* ICP *algorithm is applied. Note how the rotation and translation that aligns the two point clouds are rather small, thus initializing the* ICP *algorithm in* $\left(R^0, \mathrm{t}^0\right) = (\mathbf{I}_2, \mathbf{0}_{2\times1})$ *is sufficient.*

algorithm is thus dependent on being initialized in a good point $\left(R^0, \mathrm{t}^0\right)$ in order to converge. There are a few different ways to implement ICP, and a full overview goes beyond the scope of this thesis. Chapter 4 in the book by Nüchter (2009) is a good starting point for the interested reader.

An illustrative example of the ICP algorithm is given in Figure 2.7, where two point clouds are shown before and after the ICP algorithm is applied. A 3D example of registration is given in Figure 1.2 in Chapter 1, where the point cloud was constructed from 34 smaller point clouds which were registered to each other.

As mentioned above, ICP is a local algorithm in the sense that it, if initialized poorly, often gets stuck in local minimas of the cost function (2.17). To remedy this problem, a method that is able to find a rigid body transformation (R, t) that is close to the ground truth without relying on a good initial guess is needed. Some examples of methods that attempt to improve upon the performance of ICP are the Normal Distributions Transform (NDT) in 2D (Biber and Strasser, 2003) or 3D (Magnusson et al., 2007), CRF-Match in 2D by Ramos et al. (2007), or the approach using histograms in 2D by Bosse and Zlot (2008).

3 Classification

This chapter introduces the classification problem, with an emphasis on boosting methods for finding decision boundaries. The classification problem is defined in Section 3.1, and boosting is presented in Section 3.2. The boosting method of choice, called adaptive boosting, is presented algorithmically, and some key properties of adaptive boosting are highlighted in a series of examples.

3.1 The classification problem

The purpose of a classification method is to take an input data vector

$$\mathbf{f} = [f_1, \ldots, f_{n_f}]^{\mathrm{T}} \in \mathbb{R}^{n_f} \tag{3.1}$$

and assign it to one of K classes. Let $\mathcal{C}_k$ denote the class domain, where $k \in \{1, \ldots, K\}$ is a class index. In some classification scenarios, the K classes are assumed to be disjoint,

$$\mathcal{C}_i \cap \mathcal{C}_j = \emptyset, \quad \forall i \neq j, \tag{3.2}$$

and the input space can therefore be divided into decision regions which are separated by boundaries. These are called decision boundaries, or decision surfaces, see e.g. Bishop (2006). When the decision boundaries are affine functions of the input data $\mathbf{f}$, the corresponding classifiers are called linear. There are also non-linear classifiers, i.e. classifiers which define decision boundaries that are non-linear functions of the input data. Classes that are disjoint can be separated by linear/non-linear decision boundaries, and are therefore called linearly/non-linearly separable.

However, many problems in classification are neither linearly separable, nor are the true, underlying, data domains $\mathcal{C}_k$ disjoint. For data sets which can not be separated by linear decision boundaries, methods which combine multiple models may be used. Such methods are sometimes called committees, examples include bagging and boosting, see e.g. Bishop (2006). Bagging classifiers are formed by generating M bootstrap data sets from a single data set, and then using each bootstrap data set to train a classifier. Bootstrap data sets are generated by randomly drawing points with replacement from the original data set. Some points from the original data set may thus be drawn more than once in a bootstrap data set, while other points are not drawn at all. The bagging classification is then formed by taking the average of the M bootstrap classifications. In Paper A, boosting, presented in Section 3.2, is used to compute non-linear classifiers.

3.2 Boosting

Boosting is a machine learning method for finding combinations of simple base classifiers in order to produce a form of committee whose performance can be significantly better than any one of the base classifiers used alone. The simple base classifiers need to be just slightly better than a random guess, hence they are often called weak classifiers, see e.g. Bishop (2006). The resulting combination is (typically) better than the best individual weak classifier, and analogously the resulting classifier learned by boosting is thus called strong. The principal difference between boosting and other committee methods such as bagging, is that the training is performed sequentially. Each weak classifier is learned using a weighted form of the data set, where the weighting of each data point depends on the performance of the previous weak classifiers, see e.g. Bishop (2006). There exists a few different boosting methods, here we will limit ourselves to considering adaptive boosting.

3.2.1 Adaptive boosting

A widely used form of boosting is adaptive boosting, abbreviated AdaBoost. It is a machine learning procedure which greedily builds a strong classifier by a linear combination of weak classifiers (Freund and Shapire, 1995). When the weak classifiers are combined into a strong classifier, the resulting decision boundary is non-linear. As more weak classifiers are added, the classification error on the training data converges towards zero, and eventually becomes zero. Although this might be interpreted as over-fitting, AdaBoost has been shown to generalize well on testing data (Freund and Shapire, 1995). A more detailed overview and examination of boosting than can be given here is found in Chapter 10 in the book by Hastie et al. (2009).

Although later generalized to multiple classes, AdaBoost was originally designed for problems with two classes, i.e. $K = 2$. Rather than denoting the two classes as 1 and 2, here they are referred to as the positive class and negative class, or p and n, respectively. As input to the AdaBoost learning algorithm, N hand-labeled training data pairs are provided,

$$\left(\mathbf{f}^1, y_1\right), \ldots, \left(\mathbf{f}^i, y_i\right), \ldots, \left(\mathbf{f}^N, y_N\right), \tag{3.3}$$

where each data point $\mathbf{f}^i$ has a corresponding class label y_i. To learn a classifier using AdaBoost, data points from each class are needed. Let N_p and N_n be the number of training data points belonging to $\mathcal{C}_\text{p}$ and $\mathcal{C}_\text{n}$, respectively, i.e. $N = N_\text{n} + N_\text{p}$. The data labels in the two class problem are defined as

$$y_i = \begin{cases} 1 & \text{if } \mathbf{f}^i \in \mathcal{C}_\text{p}, \\ 0 & \text{if } \mathbf{f}^i \in \mathcal{C}_\text{n}. \end{cases} \tag{3.4}$$

In the AdaBoost algorithm, each data pair $\left(\mathbf{f}^i, y_i\right)$ is given a weight w_t^i, where t denotes the specific iteration of the algorithm. The weights are initialized as $w_1^i = \frac{1}{2N_\text{n}}$ if $y_i = 0$, or $w_1^i = \frac{1}{2N_\text{p}}$ if $y_i = 1$. This initialization ensures that each class is given half the weight of the data, and all data pairs within a class are given an equal weight.

After initialization, AdaBoost iteratively adds weak classifiers to a set of previously added weak classifiers, to find a good combination that together constitutes a strong classifier. The weak classifiers used in this thesis are decision stumps, i.e. one node decision trees, defined as

$$c\left(\mathbf{f}^i, \theta\right) = \begin{cases} 1 & \text{if } p f_j^i < p\lambda \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

with parameter $\theta = \{j, p, \lambda\}$, where j is the particular component of $\mathbf{f}^i$ selected, f_j^i, p is the polarity ($p = \pm 1$), and $\lambda \in \mathbb{R}$ is a threshold. The result of a weak classifier (3.5) is that the input space is partitioned into two half spaces, separated by an affine decision boundary which is parallel to one of the input axes.

In each iteration t, the weak classifier that minimizes the weighted classification error with respect to θ is chosen. This is performed by solving an optimization problem. Given the parameters of the best weak classifier θ_t, the training data is classified and the weights of the mis-classified data are increased (or, conversely, the weights of the correctly classified data are decreased). Further, using the classification error ε_t a weight α_t is computed for the best weak classifier. Details on how the weights are computed are given below.

This procedure is repeated until T weak classifiers $c\left(\mathbf{f}^i, \theta_t\right)$ have been computed. Weak classifiers can be added several times in each dimension of $\mathbb{R}^{n_f}$, each time with a new polarity and threshold, i.e. same j and new p and λ. The normalized weighted combination of T weak classifier together create the strong classifier $\mathbf{c}\left(\mathbf{f}^i\right)$. The output of the strong classifier is a likelihood, $\mathbf{c}\left(\mathbf{f}^i\right) \in [0,\ 1]$. To obtain a binary classification decision, a threshold $\tau \in [0,\ 1]$ is used, where the standard choice is $\tau = 0.5$. A detailed presentation of AdaBoost is given in Algorithm 1, and the learning iterations are illustrated in Example 3.1.

Algorithm 1 AdaBoost

```
Input: Labeled data pairs: $\left(\mathbf{f}^1, y_1\right), \ldots, \left(\mathbf{f}^N, y_N\right)$. Number of training iterations: $T$.
Initialize weights: $w_1^i = \frac{1}{2N_\mathrm{n}}$ if $y_i = 0$, $w_1^i = \frac{1}{2N_\mathrm{p}}$ if $y_i = 1$
1: for $t = 1, \ldots, T$ do
2:     Normalize the weights:
           $$\widetilde{w}_t^i = \frac{w_t^i}{\sum_{j=1}^{N_\mathrm{n}+N_\mathrm{p}} w_t^j}, \quad i = 1, \ldots, N_\mathrm{n} + N_\mathrm{p} \tag{3.6}$$
3:     Select the best weak classifier with respect to $\theta$,
           $$\theta_t = \arg\min_{\theta} \sum_{i=1}^{N} \widetilde{w}_t^i \left| c\left(\mathbf{f}^i, \theta\right) - y_i \right| \tag{3.7}$$
4:     Define $c_t\left(\mathbf{f}^i\right) = c\left(\mathbf{f}^i, \theta_t\right)$, and $\varepsilon_t = \sum_{i=1}^{N} \widetilde{w}_t^i \left| c_t\left(\mathbf{f}^i\right) - y_i \right|$.
5:     Update the weights:
           $$w_{t+1}^i = \widetilde{w}_t^i \beta_t^{1-e_i}, \quad i = 1, \ldots, N_\mathrm{n} + N_\mathrm{p}, \tag{3.8}$$
       where $e_i = 0$ if $\mathbf{f}^i$ is classified correctly and 1 otherwise, and $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$. Set $\alpha_t = \log \frac{1}{\beta_t}$.
6: end for
Output: Strong classifier
           $$\mathbf{c}\left(\mathbf{f}^i\right) = \frac{\sum_{t=1}^{T} \alpha_t c_t\left(\mathbf{f}^i\right)}{\sum_{t=1}^{T} \alpha_t} \in [0,\ 1] \tag{3.9}$$
```

Example 3.1: AdaBoost learning

A data set was generated in polar coordinates, where the angle f_φ was sampled uniformly in $[0\ 2\pi]$, and the range f_r was sampled from $\mathcal{N}\left(f_r\,;\,0, 0.10\right)$ for the positive class and $\mathcal{N}\left(f_r\,;\,0.50, 0.10\right)$ for the negative class. Using the transform (2.4), the sampled data was transformed into Cartesian coordinates. Note that since the range components f_r for the two classes are randomly sampled from probability distributions that overlap, the underlying classes are not separable and it is therefore difficult to define a "true" decision boundary.

However, with the known Gaussian distributions for the range components of the data, a probabilistic decision boundary can be defined by considering which class has higher probability in any given data point. Here, the probabilistic decision boundary in the range component is defined as the range for which both classes are equally probable,

$$\frac{1}{\sqrt{2\pi 0.10^2}} e^{-\frac{(r-0)^2}{2 \cdot 0.10^2}} = \frac{1}{\sqrt{2\pi 0.10^2}} e^{-\frac{(r-0.50)^2}{2 \cdot 0.10^2}} \quad \Leftrightarrow \quad r = 0.25. \tag{3.10}$$

Thus, for this example, in Cartesian coordinates the probabilistic decision boundary is a circle with radius 0.25. This probabilistic decision boundary is compared with the decision boundary learned by AdaBoost. The data set and a number of learning iterations are shown in Figure 3.1.

In Figures 3.1b to 3.1f, the decision boundaries of the weak classifiers are shown as black lines and the resulting decision regions are shown in white for $\mathcal{C}_\text{p}$ and light purple for $\mathcal{C}_\text{n}$. The increasing weight of the misclassified data is illustrated by making the markers larger.

3.2.2 Examples

In this section we show some of the properties of AdaBoost through some examples. The first two examples are for data from classes that are separable, either linearly of non-linearly. Thus, for these examples true decision boundaries can be defined. The third example is with data from classes that are non-separable, similarly to the data in Example 3.1. The last example bears the largest resemblance to the real world classification problem addressed in this thesis in Paper A.

A very basic requirement for a classification method is that it can handle data which is linearly separable. The AdaBoost algorithm is tested on such data in Example 3.2.

Example 3.2: Linearly separable data

Data points $\mathbf{f}^i$ are generated by uniform sampling in $[0\ 1] \times [0\ 1]$. In total $N = 1000$ data points are generated, and sorted into classes according to

$$\begin{cases} \mathbf{f}^i \in \mathcal{C}_\text{p} & \text{if } f_1^i < f_2^i, \\ \mathbf{f}^i \in \mathcal{C}_\text{n} & \text{otherwise.} \end{cases} \tag{3.11}$$

Thus, the class regions are well defined in the data space, $\mathbf{f} \in \mathbb{R}^2$, and a true decision boundary can be defined as the line $f_2^i = f_1^i$. Figure 3.2 shows the data and the results. AdaBoost is able to compute a good estimate of the true decision boundary, using $T = 100$ weak classifiers.

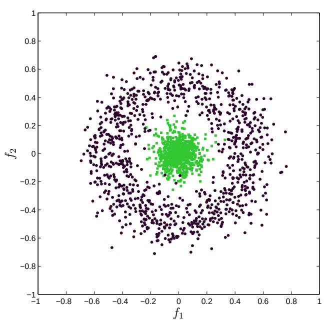
For linearly separable data, using AdaBoost can be inefficient from a computational point of view. In Example 3.2 a linear classifier, e.g. Fischer's linear discriminant (see e.g. Bishop (2006)), would be a better choice due to its lower computational demands. An example of data which is separable by a non-linear decision boundary is given in Example 3.3.

Example 3.3: Non-linearly separable data

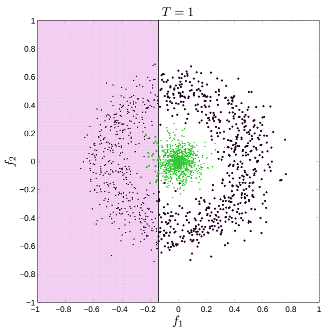
Data points $\mathbf{f}^i$ are generated by uniform sampling in $[0\ 1] \times [0\ 1]$. In total $N = 1000$ data points are generated, and sorted into classes according to

$$\begin{cases} \mathbf{f}^i \in \mathcal{C}_\text{p} & \text{if } \left|f_1^i - 0.5\right| \leq 0.25 \text{ AND } \left|f_2^i - 0.5\right| \leq 0.25, \\ \mathbf{f}^i \in \mathcal{C}_\text{n} & \text{otherwise.} \end{cases} \tag{3.12}$$

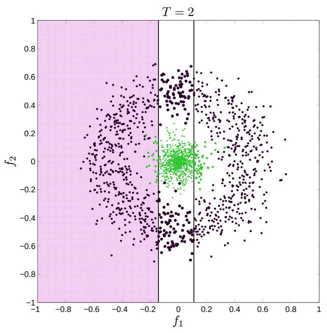
Similarly to Example 3.2, the class regions are well defined and the true decision boundary is defined as a square box. The data and the results are shown in Figure 3.3. For the particular example, using just $T = 9$ weak classifiers, AdaBoost finds a good estimate of the true decision boundary.



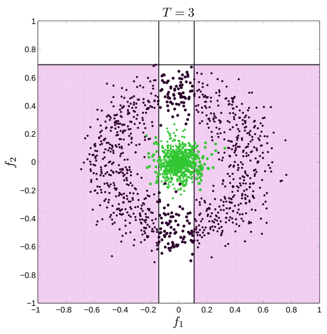
(a) *Polar data*



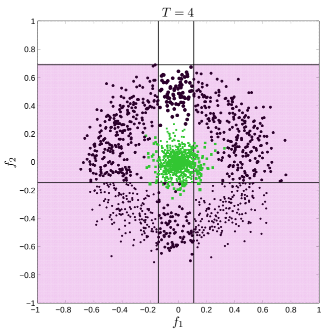
(b) *$T = 1$*



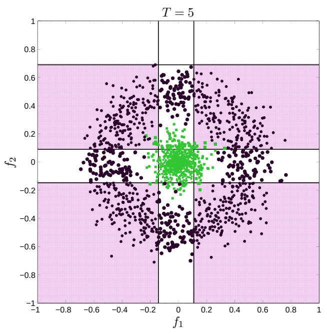
(c) *$T = 2$*



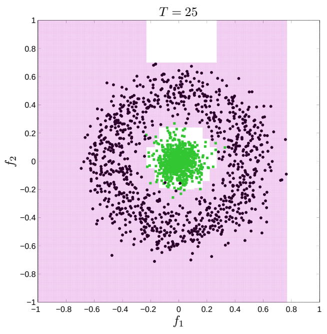
(d) *$T = 3$*



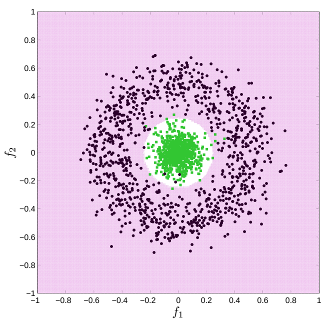
(e) *$T = 4$*



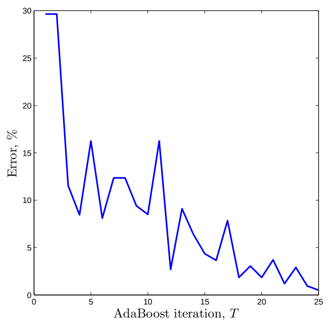
(f) *$T = 5$*



(g) *Decision region*



(h) *Probabilistic decision region.*



(i) *Error*

Figure 3.1: *AdaBoost learning. (a): polar data set, $\mathbf{f} \in \mathcal{C}_p$ in green, $\mathbf{f} \in \mathcal{C}_n$ in dark purple. (b) to (f): the first five training iterations. (g) and (h): final decision region, $T = 25$, and true probabilistic decision region. (i): Training error versus number of training iterations T. The error is defined as the percentage of data points that are mis-classified.*

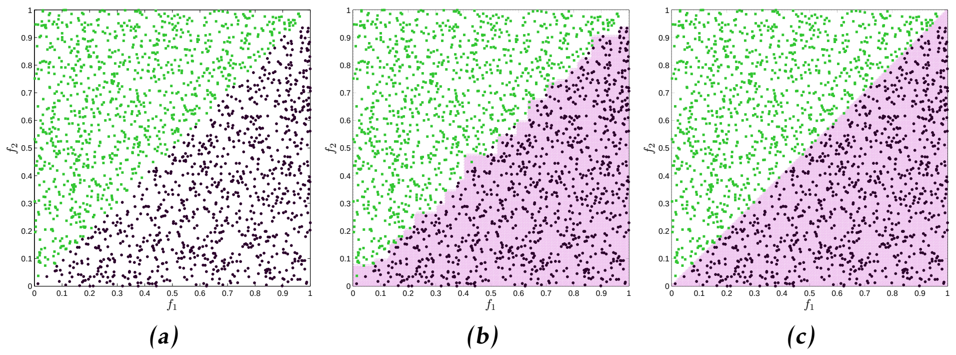


Figure 3.2: *AdaBoost with data that is linearly separable. (a): $\mathcal{C}_p$ in green, $\mathcal{C}_n$ in dark purple. (b): AdaBoost decision region, $T = 100$. (c): True decision region.*

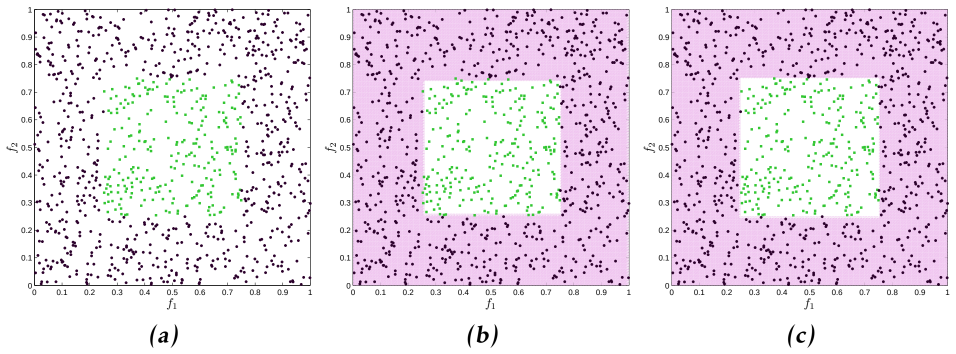


Figure 3.3: *AdaBoost with data that is separable by a non-linear decision boundary. (a): $\mathcal{C}_p$ in green, $\mathcal{C}_n$ in dark purple. (b): AdaBoost decision region, $T = 9$. (c): True decision region.*

Most practical classification problems however, are with data sampled from classes that are likely to be non-separable. Example 3.4 presents data, where each class is represented by a Gaussian distribution. Thus, with knowledge of the true underlying class distributions, a decision boundary can be computed from the probability density functions analogously to how a decision boundary was computed in Example 3.1. In the example, there is a large resemblance between the learned decision boundary and the probabilistic decision boundary.

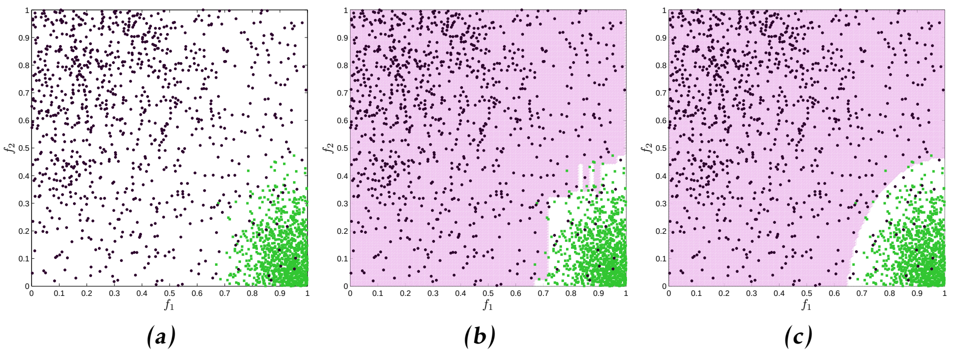


Figure 3.4: *AdaBoost with data generated by two Gaussian distributions. (a): $\mathcal{C}_p$ in green, $\mathcal{C}_n$ in dark purple. (b): AdaBoost decision boundary, $T = 100$. (c): Probabilistic decision region.*

Example 3.4: Gaussian data

This example illustrates how AdaBoost finds a decision boundary for data from two non-separable classes. The data points $\mathbf{f}^i$ are generated by sampling from two Gaussian distributions, and keeping only samples that fall in $[0\ 1] \times [0\ 1]$. Here, $N_\text{p} = N_\text{n} = 1000$ data points are generated from the following Gaussian distributions

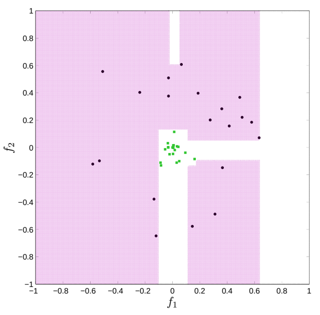
$$\begin{cases} \mathcal{C}_\text{p}: & \text{Samples from } \mathcal{N}\left(\mathbf{f}\,;\begin{bmatrix}0.95\\0.05\end{bmatrix},\begin{bmatrix}0.01 & 0\\ 0 & 0.02\end{bmatrix}\right), \\ \\ \mathcal{C}_\text{n}: & \text{Samples from } \mathcal{N}\left(\mathbf{f}\,;\begin{bmatrix}0.25\\0.75\end{bmatrix},\begin{bmatrix}0.20 & -0.005\\ -0.005 & 0.20\end{bmatrix}\right). \end{cases} \tag{3.13}$$

A probabilistic decision boundary is computed analogously to Example 3.1, i.e. by computing the value of each Gaussian's corresponding probability density function, and for each point in the data space consider which class has higher probability. In Figure 3.4 the data is shown together with the decision boundary learned by AdaBoost and the probabilistic decision boundary.

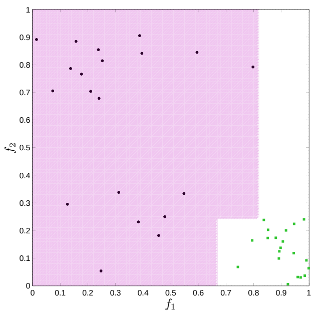
3.2.3 Properties

Above it was shown that AdaBoost has strong capabilities of finding good non-linear decision boundaries. In each of the three examples however, the number of training data was quite large. In this section, we show what happens when there is few training data available (i.e. N_p and N_n are small), or when the training data are unbalanced ($N_\text{p} \ll N_\text{n}$ or $N_\text{p} \gg N_\text{n}$). The important issue of over-fitting is also addressed. Paper A contains experiments where the data is unbalanced, and also contains experiments where the learned classifier is tested for overfitting.

Performance when data is scarce is shown in Example 3.5.



(a)



(b)

Figure 3.5: *AdaBoost when both N_p and N_n are small. A decision boundary has to be learned from an insufficient amount of data. (a): Polar data, $T = 25$. (b): Gaussian data, $T = 3$.*

Example 3.5: Few data

Data is generated by randomly selecting $N_\mathrm{p} = 20$ and $N_\mathrm{n} = 20$ data points from the polar data in Figure 3.1a, and from the Gaussian data in Figure 3.4a. These data sets were previously used in Example 3.1 and Example 3.4, thus the probabilistic decision boundaries are the same as previously. Figure 3.5 shows the results, compare to the true decision boundaries given in Figure 3.1g and Figure 3.4c. It is quite clear from the results that the decision boundary learned by AdaBoost is a poor estimate of the probabilistic decision boundaries. It can be noted though, that such few data gives a rather poor representation of the underlying true distributions, and finding a good decision boundary can be expected to be difficult using any method.

Unbalanced data is presented in Example 3.6, where the number of positive data N_p is 100 times fewer than the number of negative data N_n.

Example 3.6: Unbalanced data

Data is generated by randomly selecting $N_\mathrm{p} = 10$ data points from the positive class, and using all $N_\mathrm{n} = 1000$ data points from the negative class. As in Example 3.5, both the polar and the Gaussian data sets were used, thus the probabilistic decision boundaries are the same as previously. Figure 3.6 shows the results, compare to the probabilistic decision boundaries given in Figure 3.1g and Figure 3.4c. It is evident that with unbalanced data, AdaBoost no longer finds a decision boundary which resembles the probabilistic one. Instead the learned decision boundary adapts too much to the data, which can be interpreted as over-fitting, an issue which is addressed in the next example.

When learning models for classification, care should be taken to avoid the problem of over-fitting. Over-fitting is when the learned model adapts too much to the

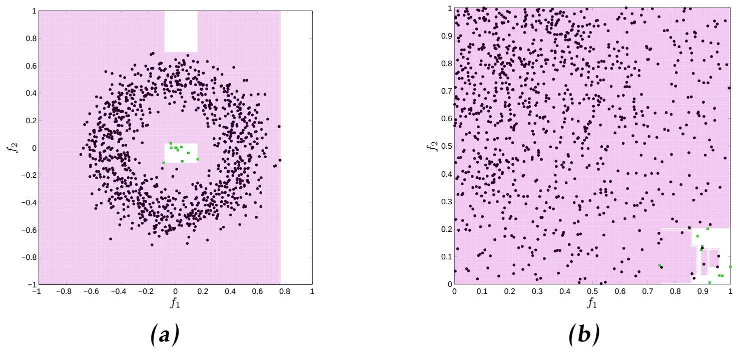


Figure 3.6: AdaBoost when the numbers of data in each class are unbalanced, here $N_p \ll N_n$. (a): Polar data, $T = 25$. (b): Gaussian data, $T = 100$.

training data, and thus does not generalize well to validation data, or the true underlying decision regions. Over-fitting, in the context of AdaBoost, corresponds to using a number of weak classifiers that is too large. Work by Freund and Shapire (1995) has shown that AdaBoost has a strong resistance to over-fitting, indeed experimental results in Paper A confirm this. Attempts have been made to explain AdaBoost, and its reported resistance to over-fitting, in terms of logistic regression, see the paper by Friedman et al. (2000). However, an exhaustive and full technical explanation has to the best of the author's knowledge not been given.

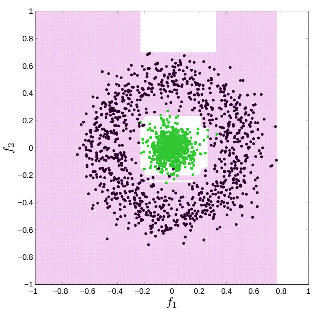
Example 3.7: Overfitting

In this example, the polar data from Example 3.1 and Gaussian data from Example 3.4 were used. For both data sets a classifier was learned using AdaBoost for $T = 1000$ iterations. Figure 3.7 shows the results, compare to the true decision boundaries given in Figure 3.1g and Figure 3.4c.

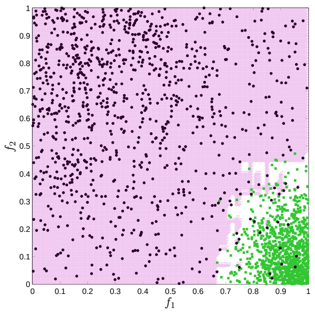
The results in Example 3.7 show that despite T being excessively large, the resulting decision boundary has not over-fitted to the training data. However, Example 3.5 did show that when data are scarce, the resulting decision boundary adapts too much to training data, i.e. the resistance to overfitting appears to be dependent on the total number of training data.

3.2.4 S-fold cross validation

When solving a classification problem it is important to keep the training data separate from the validation data. If data is scarce, a common practice is to use S-fold cross validation. With this approach, the data is partitioned into S-folds, or subsets. In each of S runs, the S:th fold is reserved for validation, and remaining $S-1$ folds are used for training. The results from each round are then pooled. This procedure allows a portion $\frac{S-1}{S}$ of the data to be used for training, while the



(a)



(b)

Figure 3.7: *Overfitting test. (a): Polar data,* $T = 1000$*. (b): Gaussian data,* $T = 1000$*.*

whole data set can be used for performance evaluation. While performing S-fold cross validation, it is important to keep the training and validation data fully disjoint. A drawback of S-fold cross validation is that the training procedure has to be repeated S times, which can prove time consuming when the training is computationally expensive, see e.g. Bishop (2006).

A sub-problem of S-fold cross validation is how to partition the data into folds. When data is a sequence over time, training data and validation data can be chosen as different time sequences. This is common practise in e.g. system identification, see e.g. Ljung (1999). In Paper A the data used in classification is not ordered temporally, and thus training and validation data can not be taken as different time sequences. Instead, the data is partitioned into folds by randomly permuting the order of the data, and then dividing the re-ordered data into folds. If the random permutation is performed correctly, each fold should be a good representation of the entire data set. It is also important to consider the labels of the data such that one, or more, of the folds do not represent an unbalanced subset of the whole data set.

3.3 Performance evaluation

This section contains the definition of some quantities related to performance evaluation of binary classifiers. It is shown how these quantities can be used to evaluate classifiers, and compare classifiers to each other. The performance evaluation metrics are used to evaluate and compare classifiers in Paper A.

3.3.1 Basic concepts

For a binary classifier, the true positive rate t_p is the number of positive test data correctly classified as positive. Similarly, the true negative rate t_n is the number of negative test data correctly classified as negative. The false positive

Table 3.1: Binary classifier contingency table.

		Predicted class		
		p	n	**Tot.**
Actual class	p	True Positive	False Negative	$t_p + f_n = N_\mathrm{p}^\mathrm{t}$
	n	False Positive	True Negative	$f_p + t_n = N_\mathrm{n}^\mathrm{t}$
	Tot.	$t_p + f_p$	$f_n + t_n$	

rate f_p is the number of negative test data incorrectly classified as positive, and the false negative rate f_n is the number of positive test data incorrectly classified as negative.

Let there be N_p^t and N_n^t number of positive and negative test data, respectively. The four outcomes of a binary classifier can be formulated in a 2×2 contingency table, or confusion matrix, as shown in Table 3.1.

3.3.2 Detection and false alarm

The detection, or true positive, rate D and false alarm, or false positive, rate FA are defined as

$$D = \frac{t_p}{t_p + f_n} = \frac{t_p}{N_\mathrm{p}^\mathrm{t}}, \tag{3.14a}$$

$$FA = \frac{f_p}{f_p + t_n} = \frac{f_p}{N_\mathrm{n}^\mathrm{t}}. \tag{3.14b}$$

For an AdaBoost learned classifier, and a given set of test data, the detection and false alarms rates can be computed for different thresholds τ, see Example 3.8.

Example 3.8: Detection and false alarms rates

A set of $N_\mathrm{p}^\mathrm{t} = N_\mathrm{n}^\mathrm{t} = 10^4$ test data was generated in the same way as the training data in Example 3.1. Figure 3.8 shows the detection and false alarm rates for different thresholds when the test data is classified using the classifiers learned in Example 3.1 (all training data), Example 3.5 (few training data), Example 3.6 (unbalanced training data), and Example 3.7 (overtraining).

The figure shows that both the detection rate and the false alarm rate decrease with an increasing threshold. This is intuitive, because a higher threshold implies that a higher likelihood is required for the test data point to be classified as belonging to the positive class.

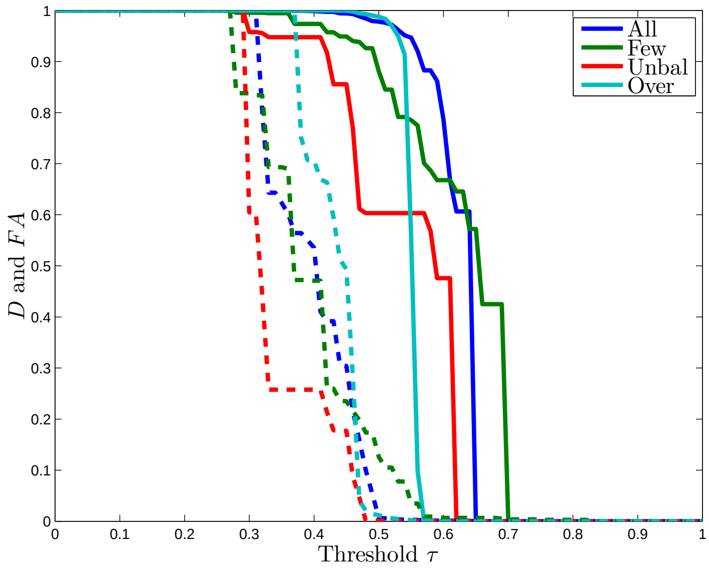


Figure 3.8: *Detection (solid) and false alarm (dashed) rates for different thresholds. The legend refers to classifiers learned using the data in Example 3.1 (All), Example 3.5 (Few), Example 3.6 (Unbal), and Example 3.7 (Over).*

3.3.3 Receiver operating characteristic

For any binary classifier the detection rate should be high and the false alarm rate should be low. However, as shown in Example 3.8, these two objectives are in conflict. A higher detection rate implies that a lower threshold should be used, but a lower threshold in turn implies a higher false alarm rate. A receiver operating characteristic (ROC) curve is an illustration of a binary classifier's trade off between detection and false alarm. Example 3.9 gives ROC curves for the detection and false alarm curves in Example 3.8.

Example 3.9: Receiver operating characteristic

The detection and false alarm rates in Figure 3.8 are plotted against each other as ROC curves in Figure 3.9. At any given false alarm rate, the detection rate should be as high as possible, which implies that the ROC curve should be as close as possible to the upper left corner of the plot. It can be seen that the overtrained classifier has similar performance as the classifier learned using all data. The classifier learned using few data points has worse performance, but is slightly better than the classifier learned using unbalanced data.

In addition to visually comparing ROC curves in order to compare different classifiers, the area under the ROC curve can be taken as a performance measure. In the best case scenario, the detection rate is 100% for any threshold that gives a

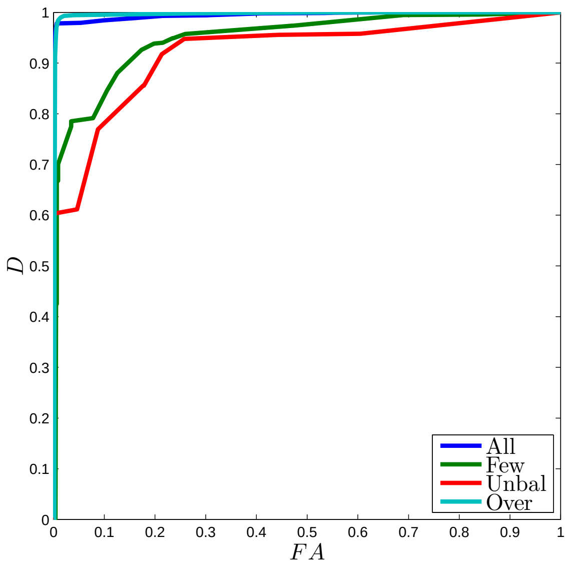


Figure 3.9: *ROC curves corresponding to the detection and false alarm curves in Figure 3.8. The legend refers to classifiers learned using the data in Example 3.1 (All), Example 3.5 (Few), Example 3.6 (Unbal), and Example 3.7 (Over).*

Table 3.2: *Area under the ROC curves in Figure 3.9.*

Classifier	All	Few	Unbal	Over
Area under ROC	99.45%	94.85%	92.07%	99.62%

non-zero false alarm rate. In this case the area under the ROC curve would be 1, which means that the larger the area under the ROC curve is, the better the classifier is. The areas under the ROC curves in Figure 3.9 are given in Table 3.2. The results confirm that the classifier learned using the unbalanced training data is the worst, and that the overtrained classifier has equal performance with the classifier learned using all training data.

In certain classification problems it is more important to have a low FA rate than to have a high D rate, one such example is given in Paper A. In this case, different classifiers can be compared by considering the D rate at a specific FA rate. Naturally, the opposite could also be true, i.e. high D rate is more important than a low FA rate. In this case, a comparison of FA rates for specific D rates can be made.

In Table 3.3 the D rates at 0% and 1% FA rate are given for the ROC curves in Figure 3.9. Judging by the D rate at 1% FA rate, the classifier learned using the unbalanced training data is again the worst. However, judging by the D rate at 0% FA rate, the classifier learned using the unbalanced training data is the only one that achieves a non-zero D rate.

Table 3.3: *D rate at* 0% *and* 1% *FA rate for the* ROC *curves in Figure 3.9.*

Classifier	All	Few	Unbal	Over
0% FA	0%	0%	46.94%	0%
1% FA	97.71%	70.75%	60.36%	98.20%

4 Estimation

This chapter is about estimation, a signal processing problem in which an unobserved signal is approximated using an observed signal containing noise. Estimation is an important part of many scientific fields, e.g. sensor fusion and robotics. A common requirement for practical estimation is a mathematical model of the observed and unobserved signals, and the relationship between the signals. In the cases where the mathematical models are linear, and the noise is Gaussian distributed, the Kalman filter is the optimal solution to the estimation problem. In many applications the state-space description is non-linear, for these cases the extended Kalman filter might be an option.

The chapter is organized as follows: the estimation problem is presented and defined in Section 4.1. Section 4.2 is about dynamical models and measurement models, with examples in both continuous and discrete time. Recursive single state Bayes filtering is overviewed in Section 4.3. Linear and non-linear estimation methods are described in Section 4.4 and performance evaluation is presented in Section 4.5.

4.1 The estimation problem

Estimation can, in a general sense, be defined as the problem of approximating, or estimating, a state $\mathbf{x}$, or a parameter $\boldsymbol{\theta}$, using the noisy measurement $\mathbf{z}$. In an estimation problem, the quantity of interest, either the state $\mathbf{x}$ or the parameter $\boldsymbol{\theta}$, or the pair $\mathbf{x}$ and $\boldsymbol{\theta}$, is unknown. To keep things simple and uncluttered, in the remainder of the chapter we will assume that it is the state $\mathbf{x}$ that is being estimated. However, note that the presented theory applies equally well to the parameter $\boldsymbol{\theta}$.

Assume, for the sake of simplicity, that the state, parameter and measurement are all vector valued variables, $\mathbf{x} = [x^1, \dots, x^{n_x}]^{\mathrm{T}} \in \mathbb{R}^{n_x}$, $\boldsymbol{\theta} = [\theta^1, \dots, \theta^{n_\theta}]^{\mathrm{T}} \in \mathbb{R}^{n_\theta}$ and $\mathbf{z} = [z^1, \dots, z^{n_z}]^{\mathrm{T}} \in \mathbb{R}^{n_z}$. Estimation can be performed in either continuous or discrete time. Let $\mathbf{x}(t)$ denote the true state in continuous time t, and let $\mathbf{x}_k$ denote the true state at discrete time instant t_k, i.e. $\mathbf{x}_k = \mathbf{x}(t_k)$. In discrete time, let $\hat{\mathbf{x}}_{k|\ell}$ denote the estimate at discrete time t_k, given all measurements up to, and including, time t_ℓ. When $t_\ell > t_k$ ($\ell > k$), the estimation problem is called smoothing, and when $t_\ell < t_k$ ($\ell < k$) the problem is called prediction. However, in this chapter we will limit ourselves to the filtering problem, i.e. when $t_\ell = t_k$ ($\ell = k$).

4.2 Dynamic models and measurement models

To solve the estimation problem it is necessary to model how the estimated quantity $\mathbf{x}$ evolves over time, i.e. to model the state dynamics, which is typically done using differential equations in continuous time. Let

$$\dot{\mathbf{x}}(t) = a\left(\mathbf{x}(t), \mathbf{u}(t), \mathbf{w}(t), \boldsymbol{\theta}(t)\right) \tag{4.1}$$

denote the dynamic motion model in continuous time. Here, $\dot{\mathbf{x}}(t)$ is the derivative of $\mathbf{x}(t)$ w.r.t. time t, $\mathbf{u}(t)$ is an exogenous input variable, and $\mathbf{w}(t)$ is random noise, often called process noise. Note that estimation of the state $\mathbf{x}$ often assumes that the parameter $\boldsymbol{\theta}$ is known. Often estimation of the state $\mathbf{x}$ cannot be performed in continuous time, instead it has to be performed in discrete time. Let

$$\mathbf{x}_{k+1} = f\left(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k, \boldsymbol{\theta}_k\right) \tag{4.2}$$

be the discrete time counterpart of (4.1). The discrete time steps are related as follows,

$$t_{k+1} = t_k + T_s(k) \tag{4.3}$$

where $T_s(k)$ is the sampling time at time step k. The sampling time can be a function of time, i.e. it can be time varying. However, in the remainder of this chapter we assume constant sampling time and simply write T_s.

In addition to modeling the state dynamics, it is necessary to model the relationship between the measurements $\mathbf{z}$ and the state $\mathbf{x}$. Let

$$\mathbf{z}(t) = c\left(\mathbf{x}(t), \mathbf{e}(t), \boldsymbol{\theta}(t)\right) \tag{4.4}$$

denote the measurement model in continuous time. Here $\mathbf{e}(t)$ is random noise, often called measurement noise. Analogously to the continuous-discrete relationship between (4.1) and (4.2), a measurement model can be given in discrete time as

$$\mathbf{z}_k = h\left(\mathbf{x}_k, \mathbf{e}_k, \boldsymbol{\theta}_k\right). \tag{4.5}$$

In the most simple case, the motion and measurement models are both linear, and the process and measurement noises are additive zero mean Gaussian. However, far from all systems can be modeled as linear and Gaussian. In the following, we will give some simple model examples in continuous and discrete time.

4.2.1 Example in continuous time

Example 4.1 gives a simple linear and noiseless state space system in continuous time.

Example 4.1: Linear state space system, continuous time

Let the state vector contain the one dimensional position p and velocity v of an object, i.e. $\mathbf{x}(t) = [p(t)\ v(t)]^{\mathrm{T}}$. The motion model can be defined as

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \begin{bmatrix}\dot{p}(t)\\ \dot{v}(t)\end{bmatrix} = \begin{bmatrix}v(t)\\ a(t)\end{bmatrix}\\ &= \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}p(t)\\ v(t)\end{bmatrix} + \begin{bmatrix}0\\ 1\end{bmatrix}\mathbf{w}(t)\\ &= A\mathbf{x}(t) + B\mathbf{w}(t) \end{aligned} \tag{4.6}$$

where $\mathbf{w}(t) = a(t)$ is the acceleration. This motion model is often called constant velocity model. Let the measurement be the position, thus the measurement model is

$$\begin{aligned}\mathbf{z}(t) &= z(t) = p(t)\\ &= \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}p(t)\\ v(t)\end{bmatrix}\\ &= C\mathbf{x}(t). \end{aligned} \tag{4.7}$$

Note that the motion and measurement models (4.6) and (4.7) are modeled as noiseless. This is an atypical choice, because most often systems are assumed to be noisy.

The motion and measurement models used in this thesis are all in discrete time, and in the next section, which presents discrete time models, noise is included. A presentation of random signals in continuous time goes beyond the scope of this thesis, instead we refer the reader to the literature, see e.g. (Jazwinski, 1970).

4.2.2 Examples in discrete time

As was noted above, some estimation problems cannot be solved in continuous time, and instead discrete time models have to be derived for the dynamic motion and measurements. A possible way to discretize a continuous model is to approximate the continuous time derivatives as

$$\dot{\mathbf{x}}(t) \approx \frac{\mathbf{x}(t+T_s) - \mathbf{x}(t)}{T_s} \tag{4.8a}$$

$$= \frac{\mathbf{x}_{k+1} - \mathbf{x}_k}{T_s}, \tag{4.8b}$$

where T_s is the sample time, and we assume that $t = kT_s$. Note that in the limit, $\lim_{T_s \to 0}$, the approximation is exact. The approximation (4.8) is also called Euler's approximation. In Example 4.2 Euler's approximation is used to find a discrete time counterpart to the continuous time system presented in Example 4.1.

Example 4.2: Linear Gaussian state space system, discrete time

Using the approximation given in (4.8), the motion model in (4.6) is given in discrete time as

$$\begin{aligned}\frac{\mathbf{x}_{k+1} - \mathbf{x}_k}{T_s} &= \begin{bmatrix} \frac{p_{k+1}-p_k}{T_s} \\ \frac{v_{k+1}-v_k}{T_s} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} p_k \\ v_k \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathbf{w}_k \\ &\Leftrightarrow \\ \mathbf{x}_{k+1} &= \begin{bmatrix} p_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_k \\ v_k \end{bmatrix} + \begin{bmatrix} 0 \\ T_s \end{bmatrix} \mathbf{w}_k \\ &= F\mathbf{x}_k + G\mathbf{w}_k, \end{aligned} \tag{4.9}$$

with discrete time process noise $p(\mathbf{w}_k) = \mathcal{N}(\mathbf{w}_k\,;\,\mathbf{0}, \mathbf{Q}_k)$. The measurement model is given in discrete time as

$$\begin{aligned}\mathbf{z}_k &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} p_k \\ v_k \end{bmatrix} + \mathbf{e}_k \\ &= H\mathbf{x}_k + \mathbf{e}_k, \end{aligned} \tag{4.10}$$

with discrete time measurement noise $p(\mathbf{e}_k) = \mathcal{N}(\mathbf{e}_k\,;\,\mathbf{0}, \mathbf{R}_k)$.

Other continuous to discrete approximations are also possible, see e.g. Gustafsson (2010). Depending on the approximation that is used, the discretization of the continuous constant velocity model in Example 4.1 could be different than the one given in Example 4.2.

However, in many cases neither the state dynamics nor the measurements can be modeled accurately as linear systems. Instead non-linear models have to be used. Similarly, the noise processes are not necessarily zero mean Gaussian, but may belong to any other probability distribution. An example of a non-linear, non-Gaussian, state space system is given in Example 4.3.

Example 4.3: Non-linear non-Gaussian state space system, discrete time

A common non-linear motion model is the coordinated turn model with polar velocity. The state is

$$\mathbf{x}_k = \begin{bmatrix} p_k^{\mathrm{x}} & p_k^{\mathrm{y}} & v_k & \phi_k & \omega_k \end{bmatrix}^{\mathrm{T}}, \tag{4.11}$$

where p_k^{x} and p_k^{y} are the positions in two dimension, v_k is the velocity, ϕ_k is the heading and ω_k is the turn rate.

The motion model, see e.g. Rong Li and Jilkov (2003), is

$$\mathbf{x}_{k+1} = \begin{bmatrix} p_k^{\mathrm{x}} + \frac{2v_k}{\omega_k}\sin\left(\frac{\omega_k T_s}{2}\right)\cos\left(\phi_k + \frac{\omega_k T_s}{2}\right) \\ p_k^{\mathrm{y}} + \frac{2v_k}{\omega_k}\sin\left(\frac{\omega_k T_s}{2}\right)\sin\left(\phi_k + \frac{\omega_k T_s}{2}\right) \\ v_k \\ \phi_k + \omega_k T_s \\ \omega_k \end{bmatrix} + \mathbf{w}_k, \tag{4.12}$$

where $\mathbf{w}_k$ is random process noise with covariance matrix (Rong Li and Jilkov, 2003)

$$\mathbf{Q}_k = \mathrm{blkdiag}\left(\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, T_s\sigma_v^2, \begin{bmatrix} \frac{T_s^3\sigma_\omega^2}{3} & \frac{T_s^3\sigma_\omega^2}{2} \\ \frac{T_s^3\sigma_\omega^2}{2} & T_s^3\sigma_\omega^2 \end{bmatrix}\right). \tag{4.13}$$

The coordinated turn motion model is often used for airplane tracking in air traffic control, and the measurement is then typically the range to the airplane measured by a radar station. If the radar station is located in $\mathbf{s} = [s^{\mathrm{x}}\ s^{\mathrm{y}}]^{\mathrm{T}}$, the measurement model for the state vector in (4.11) is

$$\mathbf{z}_k = \sqrt{\left(p_k^{\mathrm{x}} - s^{\mathrm{x}}\right)^2 + \left(p_k^{\mathrm{y}} - s^{\mathrm{y}}\right)^2} + \mathbf{e}_k, \tag{4.14}$$

where $\mathbf{e}_k$ is random measurement noise.

4.3 Recursive single state Bayes filter

It is often of interest to use dynamic and measurement models to describe the time evolution of a state. Because of the uncertainties involved, such as process and measurement noise, the knowledge of the state is often described using probability distributions.

The time evolution of the distribution of the state $\mathbf{x}$ can be described in a recursive Bayesian framework. At time step k, assume that we have a prior distribution for the state variable,

$$p\left(\mathbf{x}_k \left| \mathbf{z}^k \right.\right), \tag{4.15}$$

where $\mathbf{z}^k$ is the set of all measurements from time step 0 to time step k

$$\mathbf{z}^k = \{\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_{k-1}, \mathbf{z}_k\}. \tag{4.16}$$

The prior can be predicted (i.e. time updated) to the next time step using the Chapman-Kolmogorov equation,

$$p\left(\mathbf{x}_{k+1} \left| \mathbf{z}^k \right.\right) = \int p\left(\mathbf{x}_{k+1} \left| \mathbf{x}_k \right.\right) p\left(\mathbf{x}_k \left| \mathbf{z}^k \right.\right) \mathrm{d}\mathbf{x}_k, \tag{4.17}$$

where $p\left(\mathbf{x}_{k+1} \left| \mathbf{x}_k \right.\right)$ is the transition density from time k to time $k+1$.

Let $p\left(\mathbf{z}_{k+1}\left|\mathbf{x}_{k+1}\right.\right)$ be the measurement likelihood. Then, given the predicted prior, the corrected (i.e. measurement updated) posterior is

$$p\left(\mathbf{x}_{k+1}\left|\mathbf{z}^{k+1}\right.\right) = \frac{p\left(\mathbf{z}_{k+1}\left|\mathbf{x}_{k+1}\right.\right) p\left(\mathbf{x}_{k+1}\left|\mathbf{z}^{k}\right.\right)}{\int p\left(\mathbf{z}_{k+1}\left|\mathbf{x}_{k+1}\right.\right) p\left(\mathbf{x}_{k+1}\left|\mathbf{z}^{k}\right.\right) \mathrm{d}\mathbf{x}_{k+1}}. \tag{4.18}$$

Let $p(\mathbf{x}_0)$ be the prior at the initial time step. The prior $p(\mathbf{x}_0)$, the prediction (4.17), and the correction (4.18), are sufficient to describe the time evolution of the distribution of the state $\mathbf{x}$ given measurements $\mathbf{z}$,

$$\begin{aligned} p(\mathbf{x}_0) &\overset{\mathrm{c}}{\to} p\left(\mathbf{x}_0\left|\mathbf{z}^0\right.\right) \overset{\mathrm{p}}{\to} p\left(\mathbf{x}_1\left|\mathbf{z}^0\right.\right) \overset{\mathrm{c}}{\to} p\left(\mathbf{x}_1\left|\mathbf{z}^1\right.\right) \overset{\mathrm{p}}{\to} \dots \\ \dots &\overset{\mathrm{c}}{\to} p\left(\mathbf{x}_k\left|\mathbf{z}^k\right.\right) \overset{\mathrm{p}}{\to} p\left(\mathbf{x}_{k+1}\left|\mathbf{z}^k\right.\right) \overset{\mathrm{c}}{\to} p\left(\mathbf{x}_{k+1}\left|\mathbf{z}^{k+1}\right.\right) \overset{\mathrm{p}}{\to} \dots \end{aligned} \tag{4.19}$$

where $\overset{\mathrm{p}}{\to}$ denotes prediction and $\overset{\mathrm{c}}{\to}$ denotes correction.

It is often desired that the propagated distribution over $\mathbf{x}$ has the same functional form, i.e. that $p\left(\mathbf{x}_k\left|\mathbf{z}^{k-1}\right.\right)$ and $p\left(\mathbf{x}_k\left|\mathbf{z}^{k}\right.\right)$ are of the same functional form for all k. For example, if the initial prior is a Gaussian distribution

$$p(\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0\,;\, m_0, P_0), \tag{4.20}$$

it is desired that $p\left(\mathbf{x}_k\left|\mathbf{z}^{k-1}\right.\right)$ and $p\left(\mathbf{x}_k\left|\mathbf{z}_k\right.\right)$ are Gaussian for all k, i.e.

$$p\left(\mathbf{x}_k\left|\mathbf{z}^{k-1}\right.\right) = \mathcal{N}\left(\mathbf{x}_k\,;\, m_{k|k-1}, P_{k|k-1}\right), \tag{4.21a}$$

$$p\left(\mathbf{x}_k\left|\mathbf{z}^{k}\right.\right) = \mathcal{N}\left(\mathbf{x}_k\,;\, m_{k|k}, P_{k|k}\right). \tag{4.21b}$$

The property that the posterior distribution $p\left(\mathbf{x}_k\left|\mathbf{z}^{k}\right.\right)$ has the same functional form as the prior distribution $p\left(\mathbf{x}_k\left|\mathbf{z}^{k-1}\right.\right)$ is called conjugacy; for a given measurement likelihood $p\left(\mathbf{z}_k\left|\mathbf{x}_k\right.\right)$ the prior that gives the same posterior is called conjugate prior.

Worthy of mention for their relevance to this thesis are the conjugate pairs given in Table 4.1. A comprehensive study of conjugate pairs can be found in e.g. the book by Gelman et al. (2004).

Table 4.1: *Conjugate prior pairs*

Measurement likelihood	Conjugate Prior (variable of interest)
Poisson	Gamma (Poisson rate)
Gaussian	Gaussian (mean vector)
Multivariate Gaussian	Inverse Wishart (covariance matrix)

4.4 Some solutions to the estimation problem

There exists a variety of methods to solve estimation problems, in this section we will briefly review some of them. For linear estimation with Gaussian noise, the Kalman filter provides the optimal solution. For non-linear, non-Gaussian, problems, the extended Kalman filter and the particle filter are two possible methods. The Kalman filter is used in Paper B, Paper D and Paper H, and the extended Kalman filter is used in Paper C and Paper G.

4.4.1 Linear estimation with the Kalman filter

For the case presented in Example 4.2, when the motion and measurement models are linear and the process and measurement noise are Gaussian and independent, the estimation problem can be solved in closed form using the Kalman filter (Kalman, 1960). The Kalman filter propagates in time the first moment $m_{k|k}$ and the second moment $\mathbf{P}_{k|k}$ of the state $\mathbf{x}_k$,

$$\begin{aligned} \ldots \overset{\text{c}}{\rightarrow} \quad & \mathcal{N}\left(\mathbf{x}_k\,;\, m_{k|k}, P_{k|k}\right) \overset{\text{p}}{\rightarrow} \mathcal{N}\left(\mathbf{x}_{k+1}\,;\, m_{k+1|k}, P_{k+1|k}\right) \\ \overset{\text{c}}{\rightarrow} \quad & \mathcal{N}\left(\mathbf{x}_{k+1}\,;\, m_{k+1|k+1}, P_{k+1|k+1}\right) \overset{\text{p}}{\rightarrow} \quad \ldots \end{aligned} \tag{4.22}$$

The Kalman filter prediction and correction equations are given in Algorithm 2. Example 4.4 shows how the Kalman filter is used to estimate the states of the discrete time system given in Example 4.2.

Algorithm 2 Kalman filter

Input: Measurements: $\{\mathbf{z}_k\}_{k=0}^{N}$. Initial state estimate and covariance: $\{m_0, \mathbf{P}_0\}$. Models: F, G, and H. Parameters: $\mathbf{R}_k$ and $\mathbf{Q}_k$.

```
1: for $k = 0, \ldots, N$ do
2:     Correction (measurement update)
```

$$\hat{\mathbf{z}}_{k|k-1} = H m_{k|k-1} \tag{4.23a}$$

$$S_k = H\mathbf{P}_{k|k-1}H^{\mathrm{T}} + \mathbf{R}_k \tag{4.23b}$$

$$K_k = \mathbf{P}_{k|k-1}H^{\mathrm{T}}S_k^{-1} \tag{4.23c}$$

$$m_{k|k} = m_{k|k-1} + K_k\left(\mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1}\right) \tag{4.23d}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - K_k H \mathbf{P}_{k|k-1} \tag{4.23e}$$

```
3:     if $k < N$ then
4:         Prediction (time update)
```

$$m_{k+1|k} = F m_{k|k} \tag{4.24a}$$

$$\mathbf{P}_{k+1|k} = F\mathbf{P}_{k|k}F^{\mathrm{T}} + G\mathbf{Q}_k G^{\mathrm{T}} \tag{4.24b}$$

```
5:     end if
6: end for
```

Output: State estimates and covariances $\left\{m_{k|k}, \mathbf{P}_{k|k}\right\}_{k=1}^{N}$

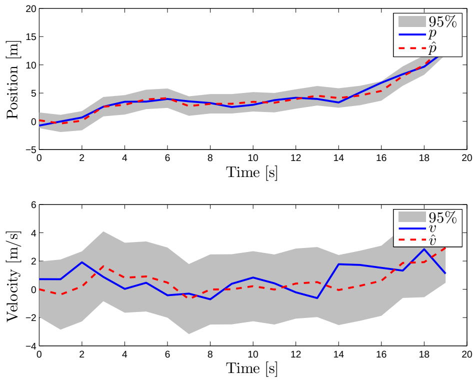


Figure 4.1: *Kalman filter state estimates, with* 95% *confidence intervals.*

Example 4.4: Kalman filter

The system in Example 4.2 was initialized at the state $\mathbf{x}_0 = [-0.75, 0.72]^{\mathrm{T}}$, and then simulated with true process noise covariance $\bar{\mathbf{Q}}_k = 1$. $N = 19$ measurements were generated with true measurement noise covariance $\bar{\mathbf{R}}_k = 1$. The solution from the Kalman filter, initialized with

$$m_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \tag{4.25a}$$

$$\mathbf{P}_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \tag{4.25b}$$

is given in Figure 4.1. In the Kalman filtering the same models were used as were used to simulate the system, and process and measurement noise covariance parameter were set to $\mathbf{Q}_k = \bar{\mathbf{Q}}_k$ and $\mathbf{R}_k = \bar{\mathbf{R}}_k$, respectively.

For certain dynamic motion models, e.g. the constant velocity model in Example 4.2, the state covariance $\mathbf{P}_{k|k}$ will converge to a steady state value, see e.g. Bar-Shalom and Fortmann (1987). In such cases, a class of stationary filters known as α-β filters can be used. The filter recursion consists of the following prediction and correction steps,

$$m_{k+1|k} = F m_{k|k}, \tag{4.26a}$$

$$m_{k+1|k+1} = m_{k+1|k} + \begin{bmatrix} \alpha \\ \frac{\beta}{T_s} \end{bmatrix} \left(\mathbf{z}_{k+1} - \hat{\mathbf{z}}_{k+1|k} \right), \tag{4.26b}$$

which, analogously to (4.22), can be expressed as

$$\ldots \xrightarrow{\mathrm{c}} m_{k|k} \xrightarrow{\mathrm{p}} m_{k+1|k} \xrightarrow{\mathrm{c}} m_{k+1|k+1} \xrightarrow{\mathrm{p}} \ldots \tag{4.27}$$

The steady state Kalman filter can be used to determine appropriate values for the parameters α and β, see e.g. Bar-Shalom and Fortmann (1987).

4.4.2 Non-linear estimation

As mentioned above, in many estimation problems the system is neither linear, nor are the noise processes Gaussian. One possible way to solve a non-linear estimation problem is to assume that the process and measurement noises are zero mean Gaussian, and to apply the so called Extended Kalman Filter (EKF), see e.g. Jazwinski (1970). As the name hints, the EKF is an extension of the Kalman filter to non-linear systems. The EKF works by, at each time step, linearizing the non-linear equations around the state estimate via first order Taylor expansion. This linearization introduces linearization errors though. Note that there is no guarantee of convergence of the EKF, however there is much practical experience showing that, if initialized properly, the solution of the EKF often converges.

In the cases when the noise distributions cannot, with reasonable accuracy, be assumed to be Gaussian, the so called Particle Filter (Gordon et al., 1993) is a good alternative to the EKF. In brief, the particle filter provides an approximation of the distribution of the state $\mathbf{x}_k$ conditioned on $\mathbf{z}^k$. The approximation of the distribution is based on a number of particles, or samples, with associated weights. There are also filters that combine the Kalman filter and the particle filter, these are called Marginalized Particle Filters or Rao-Blackwellized particle filters, see e.g. Schön et al. (2005). In brief, this type of filter is suitable for systems whose state-space models can be separated into linear and nonlinear parts. There are also non-linear filtering methods, good references on non-linear filtering are Simon (2006) and Gustafsson (2010).

4.5 Performance evaluation

When a system is simulated, as in Example 4.4, the true state $\mathbf{x}_k$ is available and can be compared to the state estimate $\hat{\mathbf{x}}_{k|k}$. In contrast to simulations, the true state is often not directly available in experiments, however sometimes an approximation of the true state can be obtained. One such example is outdoor positioning of mobile robots, in which case the position given by a GPS sensor can be used as an approximation of the true position. In other cases the true state can not be approximated, and other more or less subjective performance measures have to be used.

Performance evaluation in the absence of the true state is briefly addressed in Paper A, Paper B, Paper C, and Paper D. In the remainder of this section, it is assumed that the true state is available. When evaluating estimation results, it is important to have a well defined notion of the performance of the estimate. Two performance evaluation methods are presented in this section.

4.5.1 The root mean square error

The estimation error $\mathcal{E}_k$ is defined as

$$\mathcal{E}_k = \mathbf{x}_k - \hat{\mathbf{x}}_{k|k}. \tag{4.28}$$

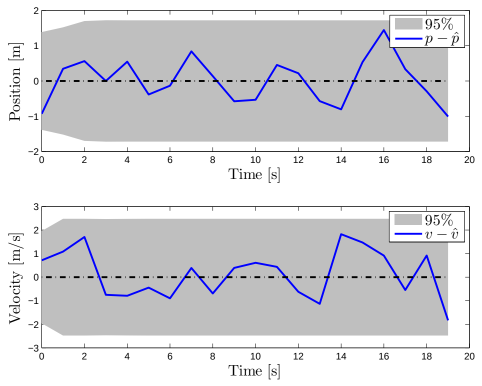


Figure 4.2: *Kalman filter estimation errors, with* 95% *confidence intervals.*

The estimation error is thus a vector of the same dimension as the underlying state $\mathbf{x}_k$, i.e. $\mathcal{E}_k = [\varepsilon_k^1, \ldots, \varepsilon_k^{n_x}]^\mathrm{T} \in \mathbb{R}^{n_x}$, and each component of the estimation error has the same unit as the corresponding state. If the states are position and velocity, as in Example 4.2, the components of the estimation error are given in meters and meter per second, respectively, see Example 4.5.

Example 4.5: Kalman filter estimation errors

The estimation errors corresponding to the Kalman filter results in Example 4.4 are given in Figure 4.2. The figure shows the estimation errors and the 95% confidence intervals.

A standard performance metric for the estimation error is the root mean square error (RMSE) ρ. Given a time sequence of states $\mathbf{x}_k$, and the corresponding state estimates $\hat{\mathbf{x}}_{k|k}$, the RMSE's are defined, for each component of the estimation error vector, as

$$\rho^i \triangleq \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(\varepsilon_k^i\right)^2}. \tag{4.29}$$

Note that the summation is taken over time for each component of the estimation error vector. The RMSE of the estimation error $\mathcal{E}_k$, i.e. the Euclidean norm of the vector, can be difficult to interpret because often the states have different dimensions[1]. An exception to this is when the state vector contains multiple states of the same dimension, in which case an RMSE can be calculated in each time step for those states[2].

When simulated data is used, Monte Carlo techniques can be used to realize the system with different process and measurement noise sequences. In such a case, the RMSE can be evaluated at each time step over the different simulations. Let

[1] cf. Example 4.2: what is the unit of a sum of position squared and velocity squared?

[2] When $\mathbf{x}$ contains x-position and y-position, the 2D Euclidean norm of $\mathbf{x}$ is the position error.

$\mathcal{E}_k^m$ be the estimation error at time step k for the m:th Monte Carlo simulation run. The RMSE at each discrete time step k is then calculated, for each component i of the estimation error vector, as

$$\rho_k^{i,\mathrm{MC}} = \sqrt{\frac{1}{n_{\mathrm{MC}}} \sum_{m=1}^{n_{\mathrm{MC}}} \left(\varepsilon_k^{i,m}\right)^2}. \tag{4.30}$$

where $\varepsilon_k^{i,m}$ is component i of $\mathcal{E}_k^m$, and n_{MC} is the number of Monte Carlo runs. Note that since the estimation error can be negative, calculating the mean estimation error should be avoided. Evaluating the RMSE over time may be of interest when the true target track contains different types of motion, e.g. linear motion and curving motion. In such cases, it is often difficult to model both types of motion using just one motion model.

Note that techniques exist that find the estimate that minimizes the squared RMSE, the so called minimum mean square error (MMSE) estimate $\hat{\mathbf{x}}^{\mathrm{MMSE}}$, see e.g. Bar-Shalom and Fortmann (1987).

4.5.2 The normalized estimation error square

An alternative to the estimation error is the normalized estimation error square (NEES) η_k, defined as

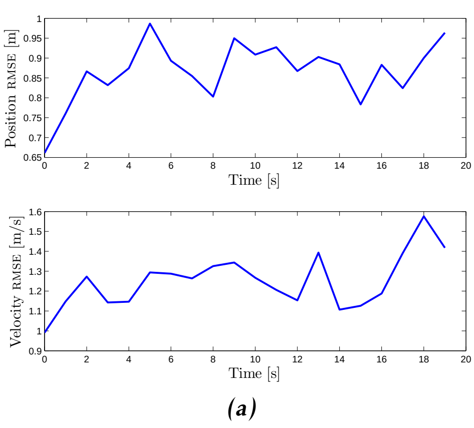
$$\eta_k \triangleq \left(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}\right)^{\mathrm{T}} \mathbf{P}_{k|k}^{-1} \left(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}\right). \tag{4.31}$$

The NEES can be understood as being a weighted average of the individual state errors, where the weights are given by the inverse state covariance. Thus, if the variance of a state estimate is high, its inverse weight will be small and a large error will have a smaller contribution to the NEES. Conversely, if the variance of a state is low, its inverse weight will be large and a large error will have a larger contribution to the NEES. Note that the NEES, in contrast to the estimation error and RMSE, is a dimensionless quantity.

When the state estimate $\hat{\mathbf{x}}_k$ is Gaussian distributed, the NEES can be shown to be χ^2-distributed with n_x degrees of freedom (Bar-Shalom et al., 2001). Thus, using the $\chi^2(n_x)$-distribution, probability gates can be computed for the NEES. Similarly to (4.30), the NEES can be averaged over Monte Carlo runs. The NEES however, is always positive by definition, and thus the sum can be calculated over the Monte Carlo runs,

$$\eta_k^{\mathrm{MC}} = \sum_{j=1}^{n_{\mathrm{MC}}} \eta_k^j = \sum_{j=1}^{n_{\mathrm{MC}}} \left(\mathbf{x}_k^j - \hat{\mathbf{x}}_{k|k}^j\right)^{\mathrm{T}} \left(\mathbf{P}_{k|k}^j\right)^{-1} \left(\mathbf{x}_k^j - \hat{\mathbf{x}}_{k|k}^j\right). \tag{4.32}$$

The Monte Carlo NEES sum η_k^{MC} is χ^2-distributed with $n_x \times n_{\mathrm{MC}}$ degrees of freedom. Probability gates $g_\gamma^{\max}$ and $g_\gamma^{\min}$, corresponding to γ% of the probability mass, can be computed using the $\chi^2(n_x n_{\mathrm{MC}})$-distribution. For a given discrete time sequence, η_k^{MC} should be within $\left[g_\gamma^{\min},\ g_\gamma^{\max}\right]$ γ% of the time instances. If it is not, it is a sign that the estimates may be inconsistent. In practical implementa-



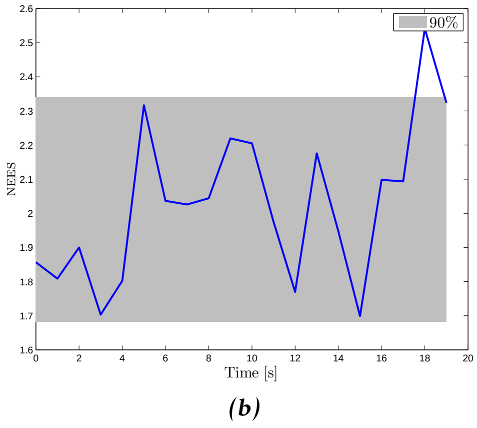


Figure 4.3: *Estimation performance evaluation.* ***(a)****: the RMSE for the position and velocity, respectively.* ***(b)****: the NEES with 90% confidence interval.*

tions, η_k^{MC} is often divided by the number of Monte Carlo runs n_{MC}. When this is performed, $g_\gamma^{\max}$ and $g_\gamma^{\min}$ must also be divided by n_{MC}. Example 4.6 shows the Monte Carlo average RMSE and NEES for the system in Example 4.4.

Example 4.6: RMSE and NEES

The system presented in Example 4.4 is simulated with 100 unique process and measurement noise sequences, and the Kalman filter was used to compute state estimates. The corresponding RMSE and NEES are shown in Figure 4.3. The NEES is within the 90% confidence interval in 19 out of 20 time steps, or 95% of the time steps.

4.6 Simultaneous localization and mapping

Filtering can be applied to solve a broad variety of problems. An example that is of high relevance to this thesis, is the Simultaneous Localization and Mapping (SLAM) problem. The SLAM problem is a robotics problem that consists of the joint estimation of the robot state ${}^r\mathbf{x}$ and the map state $\mathbf{M}$. Thus, the full state vector is

$$\mathbf{x} = \begin{bmatrix} {}^r\mathbf{x} \\ \mathbf{M} \end{bmatrix}. \tag{4.33}$$

The robot state typically consists of position and orientation, which is also called the robot pose. The map is often divided into landmarks, sometimes called features, and thus the map state $\mathbf{M}$ can be decomposed as

$$\mathbf{M} = \left[\left(\mathbf{m}^{(1)}\right)^{\mathrm{T}} \quad \dots \quad \left(\mathbf{m}^{(i)}\right)^{\mathrm{T}} \quad \dots \quad \left(\mathbf{m}^{(m)}\right)^{\mathrm{T}} \right]^{\mathrm{T}}, \tag{4.34}$$

where $\mathbf{m}^{(i)}$ is the i:th landmark state. The landmark state is often given as an (x, y) or $(\mathrm{x}, \mathrm{y}, \mathrm{z})$ position, however the orientation can also be included. In some SLAM applications the robot trajectory, i.e. a history of robot poses, is of interest and therefore included in the state vector. In the estimation, the estimated quantity is the trajectory state

$$ {}^{t}\mathbf{x} = \begin{bmatrix} {}^{r}\mathbf{x}_0^{\mathrm{T}} & {}^{r}\mathbf{x}_1^{\mathrm{T}} & \ldots & {}^{r}\mathbf{x}_k^{\mathrm{T}} & \ldots & {}^{r}\mathbf{x}_K^{\mathrm{T}} & {}^{r}\mathbf{x}_c^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}, \tag{4.35} $$

where ${}^{r}\mathbf{x}_k$ is the kth robot pose and ${}^{r}\mathbf{x}_c$ is the current robot pose. When both the robot trajectory and the whole map is estimated, the problem is called Full-SLAM. The Full-SLAM state vector is

$$ \mathbf{x} = \begin{bmatrix} {}^{t}\mathbf{x} \\ \mathbf{M} \end{bmatrix}. \tag{4.36} $$

Estimating the robot trajectory ${}^{t}\mathbf{x}$, and not the landmarks in the map state $\mathbf{M}$, is called trajectory based SLAM. In this case, the state vector is simply $\mathbf{x}={}^{t}\mathbf{x}$. In trajectory based SLAM, instead of measuring the landmarks in the map state $\mathbf{M}$, the robot measures the relative difference between the current pose ${}^{r}\mathbf{x}_c$ and some previous pose ${}^{r}\mathbf{x}_k$. A simple example of trajectory based SLAM in 2D is given in Example 4.7.

Example 4.7: Trajectory based SLAM

A true trajectory was simulated,[3] and control inputs and measurements were obtained from the simulation. The robot pose is

$$ {}^{r}\mathbf{x}_k = \begin{bmatrix} \mathrm{x}_k & \mathrm{y}_k & \phi_k \end{bmatrix}^{\mathrm{T}}, \tag{4.37} $$

where $(\mathrm{x}_k, \mathrm{y}_k)$ is the position and ϕ_k is the orientation. The relative pose

$$ {}^{r}\mathbf{x}_{k,c} = \begin{bmatrix} \mathrm{x}_{k,c} & \mathrm{y}_{k,c} & \phi_{k,c} \end{bmatrix}^{\mathrm{T}} \tag{4.38} $$

is defined as a rigid body transformation that transforms ${}^{r}\mathbf{x}_k$ to ${}^{r}\mathbf{x}_c$. For a pose defined as in (4.37), the relationship between ${}^{r}\mathbf{x}_k$, ${}^{r}\mathbf{x}_c$ and ${}^{r}\mathbf{x}_{k,c}$ is (Smith et al., 1990)

$$ {}^{r}\mathbf{x}_c = \begin{bmatrix} \mathrm{x}_c \\ \mathrm{y}_c \\ \phi_c \end{bmatrix} = \begin{bmatrix} \mathrm{x}_k + \mathrm{x}_{k,c}\cos(\phi_k) - \mathrm{y}_{k,c}\sin(\phi_k) \\ \mathrm{y}_k + \mathrm{x}_{k,c}\sin(\phi_k) + \mathrm{y}_{k,c}\cos(\phi_k) \\ \phi_k + \phi_{k,c} \end{bmatrix} \tag{4.39a} $$

$$ = \begin{bmatrix} \mathrm{x}_k \\ \mathrm{y}_k \\ \phi_k \end{bmatrix} + \begin{bmatrix} cos(\phi_k) & -\sin(\phi_k) & 0 \\ \sin(\phi_k) & \cos(\phi_k) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathrm{x}_{k,c} \\ \mathrm{y}_{k,c} \\ \phi_{k,c} \end{bmatrix} \tag{4.39b} $$

$$ ={}^{r}\mathbf{x}_k + \begin{bmatrix} R(\phi_k) & \mathbf{0}_{2\times 1} \\ \mathbf{0}_{1\times 2} & 1 \end{bmatrix} {}^{r}\mathbf{x}_{k,c}, \tag{4.39c} $$

where $R(\cdot)$ is a rotation matrix, and $\mathbf{0}_{m\times n}$ is an $m \times n$ all zero matrix.

[3] The trajectory was simulated using the EKF-SLAM toolbox written by Tim Bailey and Juan I. Nieto, Australian Centre for Field Robotics (ACFR), University of Sydney (USYD), Australia.
`http://www-personal.acfr.usyd.edu.au/tbailey/software/index.html`

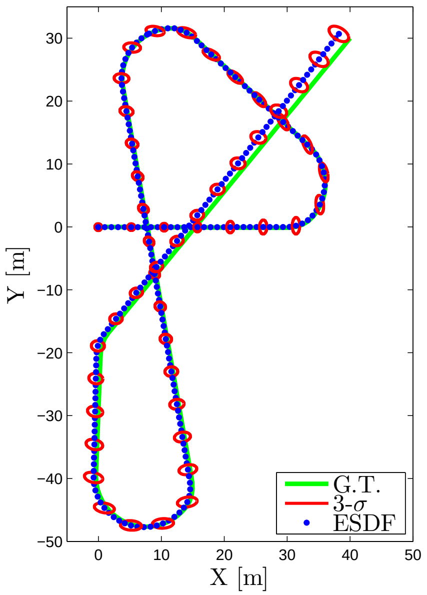


Figure 4.4: *Trajectory based SLAM. The true trajectory is given by the line, estimated poses by the dots and the* 99% *pose uncertainty is shown by the ellipses for every fifth pose.*

In this example, measurements are generated when the true current pose is sufficiently close to a previous pose, i.e. when the robot closes a loop. The measurements are of the true relative pose ${}^{r}\mathbf{x}_{k,c}$, with additive white Gaussian noise. The probability of detecting loop closure was set to one. In reality however, detecting loop closure presents a considerable challenge, and the probability of detection is less than one.

A SLAM trajectory was estimated using an Exactly Sparse Delayed state Filter (ESDF) (Eustice et al., 2006), which is an EKF on information form for state vectors of the type (4.35). The results are shown in Figure 4.4.

The origins of modern SLAM research can be traced back to the mid 80's, when probabilistic methods were first applied to mobile field robotics (Durrant-Whyte and Bailey, 2006). Early work of large impact includes the paper by Smith et al. (1990), which showed that when a robot makes relative observations of landmarks, the estimates of the landmarks are all correlated. This implied that a consistent full solution to the SLAM problem would require a joint state consisting of both robot state ${}^{r}\mathbf{x}$ and map state $\mathbf{M}$. A nice overview of SLAM research is given in the two part tutorial by Durrant-Whyte and Bailey (2006) and Bailey and Durrant-Whyte (2006).

A number of different solutions to the SLAM problem have been proposed. In online solutions, the data is considered incrementally, i.e. processed one at a time, while in offline solutions the data is typically considered in batch, i.e. all data is processed at the same time, see e.g. Thrun and Leonard (2008). Popular solutions to the SLAM problem include EKF-SLAM, see e.g. Dissanayake et al. (2001), which, as the name reveals, solves the problem using an EKF. Another solution is FAST-SLAM (Montemerlo et al., 2002), which is based on the particle filter. FAST-SLAM has been shown to suffer from particle depletion when the robot's mission grows longer in time, which results in inconsistent estimates (Bailey et al., 2006). There are, however, many practical examples where FAST-SLAM has provided good results. A third family of solutions to the SLAM problem are the graph-based solutions, pioneered by Lu and Milios (1997). The graph-based solutions solve the SLAM problem offline in batch, either as trajectory based SLAM, or as Full-SLAM.

An important part of any SLAM solution is the data association, i.e. associating measurements to the right landmark estimates, or associating relative pose estimates to the correct previous pose. Data association in SLAM is very important, because incorrect associations can lead to inconsistent SLAM-estimates. In Paper A data association for trajectory based SLAM is posed as a loop closure detection problem, and a detection classifier is learned using AdaBoost.

5 Target tracking

This chapter is about target tracking, which is a type of estimation problem. The target tracking problem is defined in Section 5.1, and some common data association methods are presented in Section 5.2. The optimal sub-pattern assignment metric for target tracking performance evaluation is defined in Section 5.3.

5.1 The target tracking problem

In this thesis, we will focus on targets that are moving objects, such as airplanes, cars and humans. Early target tracking research was motivated by, among other things, tracking of airplanes using radars, see e.g. Bar-Shalom and Fortmann (1987). When airplanes are tracked using radar stations, the distance between the sensor and target is often such that the target only occupies one resolution cell of the sensor. Due to this, each target generates at most one radar measurement. Because the targets generate at most one measurement per time step, they effectively behave as points in the surveillance space and can thus be modeled as such. This leads to the point target assumption, with the following definition:

Definition 5.1 (Point target). A target that gives rise to at most one measurement per time step.

In the following two subsections some properties of target tracking are listed, and problem formulations are given for both the case of a single target, and the case of multiple targets.

5.1.1 Single target tracking

Single target tracking can be defined as the processing of measurements in order to maintain an estimate of a target's current state. As in any estimation problem, it holds that

I each target generated measurement is corrupted by noise.

However, when the estimation problem was introduced to the reader it was implicitly assumed that in each time step there is a single state generated measurement. In contrast to this, single target tracking is complicated by the fact that

II the probability of detection for each target is less than one, i.e. in each time step it is not known whether or not the target generated a measurement,

III there are false, so called clutter, measurements, and

IV measurement origin is unknown, i.e. it is not known which measurements are target generated, and which measurements are clutter.

Each time step, a sensor delivers $N_{z,k}$ measurements $\mathbf{z}_k^{(j)}$. Let the set of measurements at time k be denoted

$$\mathbf{Z}_k = \left\{ \mathbf{z}_k^{(j)} \right\}_{j=1}^{N_{z,k}} . \tag{5.1}$$

Further, let $\mathbf{Z}^k$ be all sets of measurements from time 1 to time k, i.e. $\mathbf{Z}^k = \{\mathbf{Z}_i\}_{i=1}^k$. The objective of single target tracking is to use $\mathbf{Z}^k$ to determine whether or not there is a target present, and if a target is present, to estimate the target state $\mathbf{x}_k$.

5.1.2 Multiple target tracking

Multiple target tracking is an extension of single target tracking, and can be defined as the processing of multiple measurements obtained from multiple targets in order to maintain estimates of the targets' current states, see e.g. Bar-Shalom and Fortmann (1987). At the heart of multiple target tracking lies a joint estimation problem, namely estimating the number of targets, and estimating the states of each target. In addition to the properties I to IV listed above, in multiple target tracking

V the number of present targets is unknown.

Let $N_{x,k}$ denote the unknown number of targets present at time k, and let $\mathbf{x}_k^{(i)}$ denote the state of target i at time k. At time k the set of all present targets $\mathbf{X}_k$ is given by

$$\mathbf{X}_k = \left\{ \mathbf{x}_k^{(i)} \right\}_{i=1}^{N_{x,k}} . \tag{5.2}$$

The objective of multiple target tracking is to estimate $\mathbf{X}_k$ given $\mathbf{Z}^k$, i.e. to determine how many targets there are, and for the targets that are present, to estimate the target states $\mathbf{x}_k^{(i)}$.

5.2 Data association methods

An important part of solving a target tracking problem is solving the data association problem, sometimes also referred to as the correspondence problem. Data association means to associate each measurement to one of the measurement generating sources, i.e. either to a target or a clutter source.

Data association is an integral part of target tracking, because incorrect data association can result in disastrous tracking performance. In this section, we will briefly overview some data association methods often used for point target tracking. In each time step, each measurement is either clutter, or generated by a target. For the measurements that are generated by targets, a decision has to be made as to which measurements belong to already existing targets, and which measurements belong to new targets. Handling the data association problem is easier when the number of targets is limited to at most one. Hence, we will review single target tracking association methods first before reviewing association methods for multiple target tracking.

5.2.1 Single target tracking

When at most one target is present, i.e. single target tracking, the data association problem comes down to deciding if there is a target present, and if so, which measurement belongs to the target. Since there is at most one target present, the association can be handled locally, i.e. only the measurements closest to the target estimate are considered as potentially having been generated by the target.

Nearest neighbor

In nearest neighbor (NN) data association, the target is associated to the nearest measurement such that

$$\left(\mathbf{z}_k^{(i)} - \hat{\mathbf{z}}_{k|k-1}\right)^{\mathrm{T}} S_k^{-1} \left(\mathbf{z}_k^{(i)} - \hat{\mathbf{z}}_{k|k-1}\right) \tag{5.3}$$

is minimized. Here, the notation $\hat{\mathbf{z}}_{k|k-1}$ and S_k was introduced in Algorithm 2, and i is an index which spans over all measurements that fall within the gate. A gate, or validation region, is a part of the measurement space where the specific measurement is found with some (high) probability (Bar-Shalom and Fortmann, 1987). Gating is used as a means to reduce the risk that the target is corrected using a clutter measurement. Each measurement has a probability of being target generated, and a probability of being clutter. The measurements that fall inside the gate have a probability of being target generated that is relatively high.

One of the more common gating methods is ellipsoidal gating, which checks whether the quantity in (5.3) is larger or smaller than some gate threshold g_γ, which corresponds to γ% of the probability mass.[1] Ellipsoidal gating thus corresponds to an ellipsoidal region in the measurement space. Note that NN makes a hard decision in the sense that only the measurement that minimizes (5.3) is

[1]The quantity in (5.3) is χ^2-distributed, cf. the normalized estimation error square in (4.31).

considered, the remaining measurements that fall inside the gate are ignored in the correction step.

Probabilistic data association

Probabilistic data association (PDA) is a soft version of NN. No hard decision is made at all, instead all measurements that fall inside the gate are used to the extent that they suit the prediction. Let $\left\{\mathbf{z}_k^{(i)}\right\}_{i=1}^{m_k}$ be the measurements that fall within the gate. The following hypotheses are considered,

$$\mathcal{H}_0 : \text{All of } \left\{\mathbf{z}_k^{(i)}\right\}_{i=1}^{m_k} \text{ are clutter} \tag{5.4a}$$

$$\mathcal{H}_j : \mathbf{z}_k^{(j)} \text{ was target generated and } \left\{\mathbf{z}_k^{(i)}\right\}_{i\neq j} \text{ are clutter.} \quad j = 1,\ldots,m_k \tag{5.4b}$$

Using the total probability theorem, the probability $\mathcal{P}\left(\mathbf{x}_k \middle| \left\{\mathbf{z}_k^{(i)}\right\}_{i=1}^{m_k}\right)$ is calculated as

$$\mathcal{P}\left(\mathbf{x}_k \middle| \left\{\mathbf{z}_k^{(i)}\right\}_{i=1}^{m_k}\right) = \sum_{j=0}^{m_k} \mathcal{P}\left(\mathbf{x}_k \middle| \mathcal{H}_j, \left\{\mathbf{z}_k^{(i)}\right\}_{i=1}^{m_k}\right) \mathcal{P}\left(\mathcal{H}_j \middle| \left\{\mathbf{z}_k^{(i)}\right\}_{i=1}^{m_k}\right). \tag{5.5}$$

Details on how PDA is implemented can be found in the literature, see e.g. Bar-Shalom and Fortmann (1987).

5.2.2 Multiple target tracking

In multiple target tracking, i.e. when more than one target may be present, data association is more complicated, and using local methods is insufficient. Instead a global association decision must be made, meaning that all measurement-to-target-estimate associations have be to considered jointly.

Global nearest neighbor

Global nearest neighbor (GNN) data association considers all measurement-to-clutter/existing track/new track associations, and selects the best overall hypothesis. In an implementation, the clutter and new target tracks are typically handled by so called track initiators, and they are therefore combined into a category called external sources. At each time step, an association matrix containing measurement-to-source likelihoods is formed, and the assignment problem is then solved as a convex optimization problem, e.g. using the auction algorithm (Blackman and Popoli, 1999). While being global, and thus superior to NN, GNN represents a hard decision for each measurement and only one data association hypothesis is thus considered. In some more complex scenarios, making a hard decision may be insufficient.

Joint probabilistic data association

Joint probabilistic data association (JPDA) is a soft version of GNN analogously to how PDA is a soft version of NN. Measurement-to-target probabilities are computed jointly over all targets, and only measurements from the last time step are considered, see e.g. Bar-Shalom and Fortmann (1987).

Multiple hypothesis tracking

Multiple hypothesis tracking (MHT) is a data association method that considers association of sequences of measurements and evaluates the probabilities of all hypotheses. Quickly, the number of possible hypotheses grows very large, and therefore methods to reduce the number of hypotheses have been suggested. These include clustering to reduce combinatorial complexity, pruning of low probability hypotheses and merging of similar hypotheses, see e.g. Bar-Shalom and Rong Li (1995).

5.3 Performance evaluation

In Section 4.5 performance evaluation methods for the estimation problem were presented. While state estimation is a central part of target tracking, in many scenarios performance indicators, such as RMSE and NEES, can not be applied directly to a target tracking problem. Some typical difficulties are highlighted in Example 5.1.

Example 5.1: Multiple target tracking data association difficulties

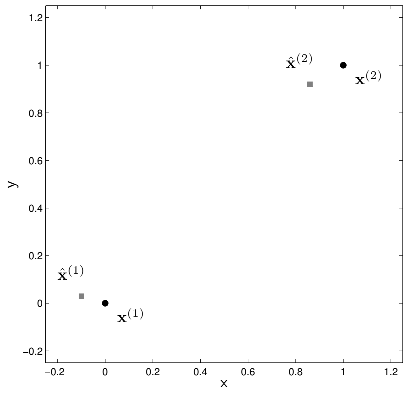
Let the true number of targets be 2, and let each state consist of Cartesian x and y position. Consider the scenarios given in Figure 5.1. Note that the index numbers for the estimates are not used as track labels. In Figure 5.1a the targets and the estimates can be arranged in vectors according to the respective index numbers, and the total position RMSE can be computed in a straightforward manner as the Euclidean norm of the estimation error. In Figure 5.1b, the indexing of the targets and the estimates does not coincide, and a straightforward computation of the RMSE does not make sense. In Figures 5.1c and 5.1d, the problem is complicated further because the number of targets is either under- or overestimated.

Thus, a multiple target tracking performance evaluation method must capture both the error in the estimated number of targets, as well as the state estimation error. Further, the performance evaluation should consider, globally, which estimate is associated to which target. One such method is the optimal subpattern assignment (OSPA) metric (Schuhmacher et al., 2008). Let

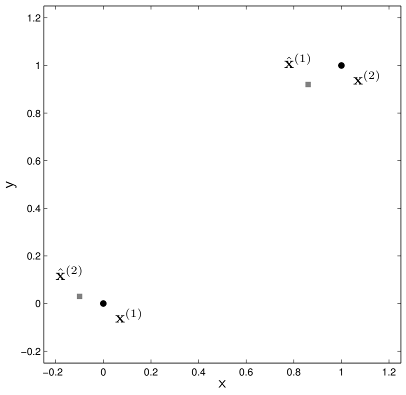
$$d^{(c)}\left(\mathbf{x}^{(k)}, \mathbf{x}^{(l)}\right) \triangleq \min\left(c, d\left(\mathbf{x}^{(k)}, \mathbf{x}^{(l)}\right)\right) \tag{5.6}$$

be the distance between $\mathbf{x}^{(k)}$ and $\mathbf{x}^{(l)}$, cut off at $c > 0$. Here $d(\cdot, \cdot)$ is any metric, in target tracking typically the Euclidean metric. Let $\mathbf{X} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\}$ and $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}^{(1)}, \ldots, \hat{\mathbf{x}}^{(n)}\}$, where $m, n \in \{0, 1, 2, \ldots\}$. Let Π_k denote the set of permutations on $\{1, 2, \ldots, k\}$ for any $k \in \{1, 2, \ldots\}$. For example, if $k = 3$ then Π_3 is

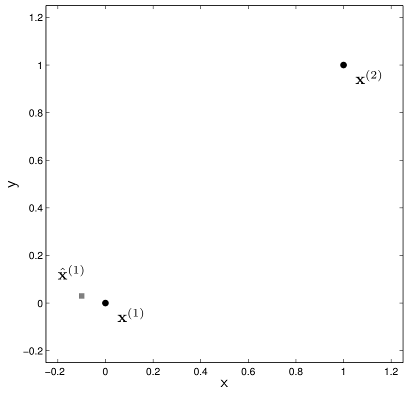
$$\Pi_3 = \Big\{[1,\ 2,\ 3],\ [1,\ 3,\ 2],\ [2,\ 1,\ 3],\ [2,\ 3,\ 1],\ [3,\ 1,\ 2],\ [3,\ 2,\ 1]\Big\}. \tag{5.7}$$



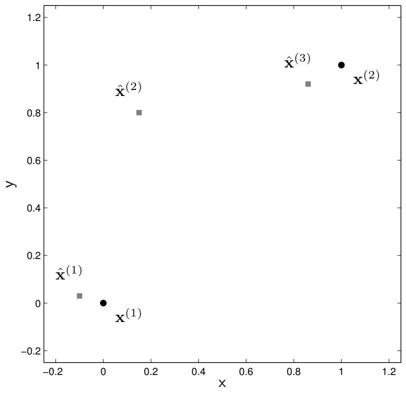
***(a)** RMSE* = 0.19



***(b)** RMSE* = 1.93



***(c)** RMSE* =?



***(d)** RMSE* =?

Figure 5.1: *Multiple target tracking data association difficulties. There are two true targets, marked with black dots, and the estimated targets are marked with gray squares.*

Define $\bar{d}_p^{(c)}\left(\mathbf{X}, \hat{\mathbf{X}}\right)$, called the OSPA metric of order p with cut-off c, as

$$\bar{d}_p^{(c)}\left(\mathbf{X}, \hat{\mathbf{X}}\right) \triangleq \left(\frac{1}{n}\left(\min_{\pi\in\Pi_n}\sum_{i=1}^{m} d^{(c)}\left(\mathbf{x}^{(i)}, \hat{\mathbf{x}}^{(\pi(i))}\right)^p + c^p\left(n-m\right)\right)\right)^{\frac{1}{p}} \qquad \text{if } m \le n \quad (5.8a)$$

$$\bar{d}_p^{(c)}\left(\mathbf{X}, \hat{\mathbf{X}}\right) \triangleq \bar{d}_p^{(c)}\left(\hat{\mathbf{X}}, \mathbf{X}\right) \qquad \text{if } m > n \quad (5.8b)$$

$$\bar{d}_p^{(c)}\left(\mathbf{X}, \hat{\mathbf{X}}\right) \triangleq 0 \qquad \text{if } m = n = 0 \quad (5.8c)$$

Note that if $d\left(\cdot,\cdot\right)$ is the Euclidean metric, $n = m$, $p = 2$, $c = \infty$ and the optimal permutation is $\pi(i) = i$, the OSPA $\bar{d}_2^{(\infty)}$ reduces to the RMSE multiplied with $\sqrt{n_x}$. It is shown by Schuhmacher et al. (2008) that (5.8) is indeed a metric. Similarly to how a minimum mean square error estimate can be found, see Section 4.5, a minimum mean OSPA (MMOSPA) estimate $\hat{\mathbf{x}}^{\text{MMOSPA}}$ can be found (Guerriero et al., 2010). In Example 5.2 the OSPA metric is evaluated for the scenarios presented in Example 5.1.

Example 5.2: OSPA metric

For the multiple target tracking scenarios given in Figure 5.1, the OSPA metric, evaluated with $c = 0.5$ and $p = 2$, is

$$\textbf{(a)}: \bar{d}_2^{(0.5)}\left(\mathbf{X}, \hat{\mathbf{X}}\right) = 0.14 \tag{5.9a}$$

$$\textbf{(b)}: \bar{d}_2^{(0.5)}\left(\mathbf{X}, \hat{\mathbf{X}}\right) = 0.14 \tag{5.9b}$$

$$\textbf{(c)}: \bar{d}_2^{(0.5)}\left(\mathbf{X}, \hat{\mathbf{X}}\right) = 0.36 \tag{5.9c}$$

$$\textbf{(d)}: \bar{d}_2^{(0.5)}\left(\mathbf{X}, \hat{\mathbf{X}}\right) = 0.31 \tag{5.9d}$$

6 Random finite sets and the probability hypothesis density

This chapter is about random finite sets and the probability hypothesis density. Random finite sets and multi-target calculus are reviewed in Section 6.1. These mathematical concepts can be used to derive a recursive multi-state Bayes filter. Multi-state Bayes filters is the topic of Section 6.2, with an emphasis on the probability hypothesis density filter.

6.1 Introduction

When the estimation problem was presented, see Chapter 4, the notion of a random vector $\mathbf{x}$ was implicitly assumed to be known to the reader. In this section the reader is introduced to the perhaps less familiar concept of a random finite set, and is also given a short overview of multi-target calculus.

6.1.1 Random finite sets

In the previous chapter both the measurements and targets were seen as sets, see (5.1) and (5.2). Considering the properties I to V listed in Section 5.1, in multiple target tracking it becomes suitable to model the numbers of elements in the measurement and target sets as random variables, and to model all elements in the sets as random variables.

A random finite set (RFS) is a set where each element in the set is a random variable, and where the number of elements in the set is a non-negative integer valued random variable. Mahler (2007b) defines random finite set as follows.

Definition 6.1 (Random finite set). A random variable Ξ that draws its instantiations $\Xi = \mathbf{X}$ from the hyperspace $\mathcal{X}$ of all finite subsets $\mathbf{X}$ (the null set $\emptyset$ included) of some underlying space $\mathcal{X}_0$.

One of the benefits of using RFS:s to model multiple target tracking is that doing so simplifies the performance evaluation, as highlighted in the examples in Section 5.3. The number of elements in an RFS is also called the set cardinality, and is denoted $|\mathbf{X}|$. Note that an RFS $\mathbf{X}$ is without ordering, which implies that

$$\mathbf{X} = \left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\right\} = \left\{\mathbf{x}^{(2)}, \mathbf{x}^{(1)}\right\} \tag{6.1}$$

for $|\mathbf{X}| = 2$. The RFS concept is used for multiple target tracking in Paper B to Paper H. In Example 6.1 we give an RFS example whose elements are random state vectors.

Example 6.1: Random finite set in Euclidean vector space

In multiple target tracking the target states can typically be defined as vectors in a Euclidean vector space $\mathbb{R}^{n_x}$, e.g. this is the case in the examples in Chapter 4.

Let the underlying state space be a Euclidean vector space, i.e. $\mathcal{X}_0 = \mathbb{R}^{n_x}$. Then the hyperspace $\mathcal{X}$ consists of the following finite sets:

$$\begin{array}{rcl} \mathbf{X} & = & \emptyset \\ \mathbf{X} & = & \mathbf{x}^{(1)} \in \mathbb{R}^{n_x} \\ & \vdots & \\ \mathbf{X} & = & \left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N_x)}\right\}, \quad \mathbf{x}^{(i)} \in \mathbb{R}^{n_x} \quad \forall i \\ & \vdots & \end{array} \tag{6.2}$$

where $N_x \geq 0$ is the set cardinality.

Thus, in multiple target tracking the target set (5.2) can be modeled as a random finite set $\mathbf{X}_k \in \mathcal{X}$, where $\mathcal{X}_0 = \mathbb{R}^{n_x}$. This has a simple interpretation: there could be any number of targets present, and the state of each present target is a vector in $\mathbb{R}^{n_x}$.

6.1.2 A brief overview of multi-target calculus

The probability mass function and probability density function are two important concepts in the context of random variables. In this section the RFS generalizations of these concepts, given by Mahler (2007b), are introduced.

For a random variable $\mathbf{x} \in \mathcal{X}_0$ the probability mass function $p_{\mathbf{x}}(S)$ gives the probability of $\mathbf{x}$ being in some part $S \subseteq \mathcal{X}_0$,

$$P_{\mathbf{x}}(S) = \Pr(\mathbf{x} \in S). \tag{6.3}$$

The probability density function $p_{\mathbf{x}}(x)$ describes the relative likelihood of $\mathbf{x}$ to occur at a given point x, and it is related to the probability mass function by an integral

$$P_{\mathbf{x}}(S) = \int_S p_{\mathbf{x}}(\mathbf{x}) \mathrm{d}\mathbf{x}, \tag{6.4}$$

and a derivative

$$p_{\mathbf{x}}(x) = \left.\frac{\mathrm{d}P_{\mathbf{x}}(S)}{\mathrm{d}\mathbf{x}}\right|_{S=x}. \tag{6.5}$$

The probability mass function for a random vector can be generalized to the belief mass function for an RFS. The belief-mass function is denoted $\beta_\Xi(S)$, and is defined as the probability that the random finite set Ξ on $\mathcal{X}_0$ is within some region S,

$$\beta_\Xi(S) = \Pr(\Xi \subseteq S). \tag{6.6}$$

Similarly, the probability density function for a random vector can be generalized to the probability density function $p_\Xi(\mathbf{X})$ of a random finite set Ξ.

A multi-state density function on the underlying space $\mathcal{X}_0$ is a real valued function $p(\mathbf{X})$ of a finite subset variable $\mathbf{X} \subseteq \mathcal{X}_0$ such that, if $\mathcal{X}_0$ has a unit of measurement u, the unit of measurement of $p(\mathbf{X})$ is $u^{|\mathbf{X}|}$. Furthermore a multi-state density function $p(\mathbf{X})$ is a multi-state probability density function if $p(\mathbf{X}) \geq 0$ for all $\mathbf{X}$ and if

$$\int p(\mathbf{X})\delta\mathbf{X} = 1. \tag{6.7}$$

The probability density function of an RFS Ξ is, if it exists, the function $p_\Xi(\mathbf{X})$, such that

$$\int_S p_\Xi(\mathbf{X})\,\delta\mathbf{X} = \Pr(\Xi \subseteq S),\ \forall S. \tag{6.8}$$

The relation between $p_\Xi(\mathbf{X})$ and $\beta_\Xi(S)$ is thus given by a set integral

$$\beta_\Xi(S) = \int_S p_\Xi(\mathbf{X})\delta\mathbf{X} \tag{6.9}$$

and a set derivative

$$p_\Xi(\mathbf{X}) = \left.\frac{\delta\beta_\Xi(S)}{\delta\mathbf{X}}\right|_{S=\emptyset}. \tag{6.10}$$

Please refer to the book by Mahler (2007b) for the definitions of the set integral and the set derivative. Multi-target distributions can also be expressed in vector notation, e.g.

$$p_\Xi\left(\left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\right\}\right) = 2p_{\mathbf{x}}\left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\right), \tag{6.11}$$

since the probability density must be distributed equally over the two possible vectors $\left[\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\right]$ and $\left[\mathbf{x}^{(2)}, \mathbf{x}^{(1)}\right]$. In general it holds that

$$p_\Xi\left(\left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right\}\right) = n!p_{\mathbf{x}}\left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right). \tag{6.12}$$

As mentioned earlier in the chapter, the cardinality of an RFS is a random variable, and it is necessary to model its distribution. The cardinality distribution of the finite random set Ξ is

$$\begin{aligned} p_{\Xi}(n) &\triangleq p_{|\Xi|}(n) && (6.13a)\\ &\triangleq \Pr(|\Xi| = n) && (6.13b)\\ &= \frac{1}{n!}\int_{\mathcal{X}_0} p\left(\left\{\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(n)}\right\}\right) d\mathbf{x}^{(1)}\ldots d\mathbf{x}^{(n)}. && (6.13c)\end{aligned}$$

Let $p(n)$ be a probability distribution on the non-negative integers and let $p_{\mathbf{x}}(\mathbf{x})$ be a probability density function on $\mathcal{X}_0$. For any $\mathbf{X} = \left\{\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(n)}\right\}$ with $|\mathbf{X}| = n$, define

$$p_{\Xi}(\mathbf{X}) \triangleq n!p(n)\prod_{i=1}^{n} p_{\mathbf{x}}\left(\mathbf{x}^{(i)}\right). \tag{6.14}$$

An i.i.d. cluster process is any random finite set Ξ that has $p_{\Xi}(\mathbf{X})$ as its distribution, for some choice of $p(n)$ and $p_{\mathbf{x}}(\mathbf{x})$. In Example 6.2 a type of i.i.d. cluster process called Poisson process is given.

Example 6.2: Multitarget Poisson process

If $p(n)$ is the Poisson probability mass function with rate parameter γ,

$$\begin{aligned} p(n) =& \mathcal{PS}(n;\ \gamma) && (6.15a)\\ =& \frac{\gamma^n e^{-\gamma}}{n!}, && (6.15b)\end{aligned}$$

then the RFS $\mathbf{X}$ with pdf

$$p_{\Xi}(\mathbf{X}) = e^{-\gamma}\gamma^n\prod_{i=1}^{n} p_{\mathbf{x}}\left(\mathbf{x}^{(i)}\right) \tag{6.16}$$

is a multi-target Poisson process.

The concepts i.i.d. cluster process and Poisson process can be used to derive practical recursive multi-state Bayes filters, e.g. the probability hypothesis density filter and the cardinalized probability hypothesis density filter, which are given in the next section.

For a more in depth description of multi-target calculus, please refer to the book by Mahler (2007b).

6.2 Recursive multi-state Bayes filter

In Section 4.3 we described the time evolution of the probability distribution of a single target. In this section we generalize this to the multi-target case using the concepts and tools introduced previously in this chapter. For the sake of simplicity and brevity, we drop the subscript $_{\Xi}$ from the multi-target pdf, i.e.

$$p(\mathbf{X}) = p_{\Xi}(\mathbf{X}). \tag{6.17}$$

At time step k, assume that we have a posterior distribution for the multi-target set,

$$p\left(\mathbf{X}_k \left| \mathbf{Z}^k \right.\right). \tag{6.18}$$

The posterior can be predicted (time updated) to the next time step using the set equivalent of (4.17),

$$p\left(\mathbf{X}_{k+1} \left| \mathbf{Z}^k \right.\right) = \int p\left(\mathbf{X}_{k+1} \left| \mathbf{X}_k \right.\right) p\left(\mathbf{X}_k \left| \mathbf{Z}^k \right.\right) \delta \mathbf{X}_k, \tag{6.19}$$

where $p\left(\mathbf{X}_{k+1} \left| \mathbf{X}_k \right.\right)$ is the multi-target transition density from time k to time $k+1$.

Let $p\left(\mathbf{Z}_{k+1} \left| \mathbf{X}_{k+1} \right.\right)$ be the multi-target measurement likelihood. Then the set equivalent of (4.18) is

$$p\left(\mathbf{X}_{k+1} \left| \mathbf{Z}^{k+1} \right.\right) = \frac{p\left(\mathbf{Z}_{k+1} \left| \mathbf{X}_{k+1} \right.\right) p\left(\mathbf{X}_{k+1} \left| \mathbf{Z}^k \right.\right)}{\int p\left(\mathbf{Z}_{k+1} \left| \mathbf{X}_{k+1} \right.\right) p\left(\mathbf{X}_{k+1} \left| \mathbf{Z}^k \right.\right) \delta \mathbf{X}_{k+1}}. \tag{6.20}$$

The prior at the initial time step $p\left(\mathbf{X}_0\right)$, and the prediction and correction equations, (6.19) and (6.20) respectively, is sufficient to describe the time evolution of the distribution of the state $\mathbf{X}$ given measurements $\mathbf{Z}$,

$$\begin{aligned} p\left(\mathbf{X}_0\right) &\xrightarrow{\text{c}} p\left(\mathbf{X}_0 \left| \mathbf{Z}^0 \right.\right) \xrightarrow{\text{p}} p\left(\mathbf{X}_1 \left| \mathbf{Z}^0 \right.\right) \xrightarrow{\text{c}} p\left(\mathbf{X}_1 \left| \mathbf{Z}^1 \right.\right) \xrightarrow{\text{p}} \ldots \\ \ldots &\xrightarrow{\text{c}} p\left(\mathbf{X}_k \left| \mathbf{Z}^k \right.\right) \xrightarrow{\text{p}} p\left(\mathbf{X}_{k+1} \left| \mathbf{Z}^k \right.\right) \xrightarrow{\text{c}} p\left(\mathbf{X}_{k+1} \left| \mathbf{Z}^{k+1} \right.\right) \xrightarrow{\text{p}} \ldots \end{aligned} \tag{6.21}$$

From the above we see that there are many conceptual similarities between single target Bayes filtering, presented in Section 4.3, and multi-target Bayes filtering. However, while the single target Bayes filter is computationally tractable, its multi-target generalization is typically not, due to the need to compute set integrals. Because of this approximations are needed, in the next section we will introduce a first order approximation of the full multi-target Bayes filter.

6.2.1 The probability hypothesis density filter

For a random vector $\mathbf{x} \in \mathbb{R}^{n_x}$ with conditional pdf $p_{\mathbf{x}}(\mathbf{x}|\mathbf{z})$, the first order moment, also called the expected value, is defined as

$$\mathrm{E}_{p_{\mathbf{x}}}\left[\mathbf{x}|\mathbf{z}\right] \triangleq \int_{\mathbb{R}^{n_x}} \mathbf{x}\, p_{\mathbf{x}}\left(\mathbf{x}|\mathbf{z}\right) \mathrm{d}\mathbf{x} \tag{6.22a}$$

$$= \hat{\mathbf{x}}. \tag{6.22b}$$

For a random vector with Gaussian pdf $p_{\mathbf{x}}\left(\mathbf{x}_k|\mathbf{z}^k\right) = \mathcal{N}\left(\mathbf{x}_k\,;\, m_{k|k}, P_{k|k}\right)$, the expected value is equal to the mean vector, $\hat{\mathbf{x}}_{k|k} = m_{k|k}$. The α-β-filter recursion (4.27) corresponds to propagating the expected value,

$$\ldots \xrightarrow{\text{c}} \hat{\mathbf{x}}_{k|k} \xrightarrow{\text{p}} \hat{\mathbf{x}}_{k+1|k} \xrightarrow{\text{c}} \hat{\mathbf{x}}_{k+1|k+1} \xrightarrow{\text{p}} \ldots \tag{6.23}$$

The first-order moment of a multi-target pdf is a density function defined on single target states $\mathbf{x} \in \mathcal{X}_0$ (Mahler, 2007b),

$$D_{k|k}(\mathbf{x}). \tag{6.24}$$

In point process theory $D_{k|k}(\mathbf{x})$ is called first-moment density or intensity density, see e.g. Mahler (2007b), however for target tracking the name probability hypothesis density (PHD) was adopted by Mahler (2007b). Despite its name, the PHD should not be confused with a probability density function. It is uniquely determined by the property that, given any region S in single target state space $\mathcal{X}_0$, i.e. $S \subseteq \mathcal{X}_0$, the integral

$$\int_S D_{k|k}(\mathbf{x})\,\mathrm{d}\mathbf{x} \tag{6.25}$$

is the expected number of targets in S (Mahler, 2007b). Especially, if $S = \mathcal{X}_0$ is the entire state space then

$$N_{k|k} \triangleq \int D_{k|k}(\mathbf{x})\,\mathrm{d}\mathbf{x} \tag{6.26}$$

is the expected total number of targets (Mahler, 2007b).

The PHD filter propagates the first order multi-target moment (6.24) through time,

$$\begin{array}{ccccccccc} D_0(\mathbf{x}) & \overset{\mathrm{c}}{\to} & D_{0|0}(\mathbf{x}) & \overset{\mathrm{p}}{\to} & D_{1|0}(\mathbf{x}) & \overset{\mathrm{c}}{\to} & D_{1|1}(\mathbf{x}) & \overset{\mathrm{p}}{\to} & \ldots \\ \ldots & \overset{\mathrm{c}}{\to} & D_{k|k}(\mathbf{x}) & \overset{\mathrm{p}}{\to} & D_{k+1|k}(\mathbf{x}) & \overset{\mathrm{c}}{\to} & D_{k+1|k+1}(\mathbf{x}) & \overset{\mathrm{p}}{\to} & \ldots \end{array} \tag{6.27}$$

and can be interpreted as an RFS equivalent to the α-β-filter for state vector estimation, see Section 4.4.1. It has been noted that, in principle, it is possible to derive predictor and corrector equations for a second order multi-target filter, however such a filter is unlikely to be computationally tractable (Mahler, 2007b).

PHD filter initialization consists of choosing a prior PHD (Mahler, 2007b)

$$D_0(\mathbf{x}) = n_0 \times s_0(\mathbf{x}), \tag{6.28}$$

where n_0 is the initial expected number of targets, and $s_0(\mathbf{x})$ is a pdf with peaks that correspond to likely initial target locations.

Given a PHD $D_{k|k}(\mathbf{x})$, the predicted PHD is (Mahler, 2007b)

$$\begin{aligned} D_{k+1|k}(\mathbf{x}) = & \underbrace{D^{\mathrm{b}}_{k+1|k}(\mathbf{x})}_{\text{Birth of new targets}} + \underbrace{\int p_{\mathrm{S}}(\mathbf{x}')\, p_{k+1|k}(\mathbf{x}|\mathbf{x}')\, D_{k|k}(\mathbf{x}')\,\mathrm{d}\mathbf{x}'}_{\text{Prediction of existing targets}} \\ & + \underbrace{\int p^{\mathrm{s}}_{k+1|k}(\mathbf{x}|\mathbf{x}')\, D_{k|k}(\mathbf{x}')\,\mathrm{d}\mathbf{x}'}_{\text{Target spawning}} \end{aligned} \tag{6.29}$$

where

- $D^{\mathrm{b}}_{k+1|k}(\mathbf{x})$ is the likelihood that new targets will appear at time $k+1$,
- $p_{\mathrm{S}}(\mathbf{x}')$ is the probability that a target with state $\mathbf{x}'$ at time k will survive to time $k+1$,
- $p_{k+1|k}(\mathbf{x}|\mathbf{x}')$ is a single target Markov transition density, and
- $p^{\mathrm{s}}_{k+1|k}(\mathbf{x}|\mathbf{x}')$ is the likelihood that at target with state $\mathbf{x}'$ at time k will spawn a target with state $\mathbf{x}$ at time $k+1$.

Under the assumption that the multi-target distribution corresponding to the predicted PHD is approximately Poisson, the corrected PHD is (Mahler, 2007b)

$$\begin{aligned} D_{k+1|k+1}(\mathbf{x}) \approx & \underbrace{(1-p_{\mathrm{D}}(\mathbf{x}))\,D_{k+1|k}(\mathbf{x})}_{\text{Not detected targets}} \\ & + \underbrace{\sum_{\mathbf{z}\in\mathbf{Z}} \frac{p_{\mathrm{D}}(\mathbf{x})\,p_{k+1}(\mathbf{z}|\mathbf{x})}{\lambda c(\mathbf{z}) + \int p_{\mathrm{D}}(\mathbf{x}')\,p_{k+1}(\mathbf{z}|\mathbf{x}')\,D_{k+1|k}(\mathbf{x}')\,\mathrm{d}\mathbf{x}'} D_{k+1|k}(\mathbf{x})}_{\text{Detected targets}} \end{aligned} \tag{6.30}$$

where

- $p_{\mathrm{D}}(\mathbf{x})$ is the probability that a measurement will be collected at time step $k+1$ from a target with state $\mathbf{x}$,
- $p_{k+1}(\mathbf{z}|\mathbf{x})$ is the sensor likelihood function,
- λ is the average of the Poisson distributed number of false alarms collected by the sensor, and
- $c(\mathbf{z})$ is the spatial distribution of the false alarms.

In (6.29) and (6.30) the theoretical equations for PHD prediction and correction are given, however these equations must be implemented to be useful in a practical setting. One alternative is to use particle filters, e.g. a sequential Monte Carlo filter; another alternative is to approximate the PHD with a weighted mixture of Gaussian distributions and use the Kalman filter, or one of its non-linear extensions.

A sequential Monte Carlo implementation of the PHD filter for point targets is presented by Vo et al. (2005), with a convergence analysis given by Vo et al. (2005); Clark and Bell (2006); Johansen et al. (2006). The Gaussian mixture PHD (GM-PHD) filter for point targets assumes that at time step k the PHD has the following Gaussian mixture representation (Vo and Ma, 2006),

$$D_{k|k}(\mathbf{x}) = \sum_{j=1}^{J_{k|k}} w^{(j)}_{k|k}\mathcal{N}\left(\mathbf{x}\,;\, m^{(j)}_{k|k}, P^{(j)}_{k|k}\right). \tag{6.31}$$

Thus, because $\int \mathcal{N}\left(\mathbf{x}\,;\, m_{k|k}^{(j)}, P_{k|k}^{(j)}\right) \mathrm{d}\mathbf{x} = 1$, the number of present targets is readily given as $N_{k|k} = \sum_{j=1}^{J_{k|k}} w_{k|k}^{(j)}$. To arrive at predictor and corrector equations, the following assumptions were made by Vo and Ma (2006), repeated below for clarity:

Assumption A1. Each target evolves and generates observations independently of one another.

Assumption A2. Clutter is Poisson and independent of target-originated measurements.

Assumption A3. The predicted multiple-target RFS is Poisson.

Assumption A4. Each target follows a linear Gaussian dynamical model and the sensor has a linear Gaussian measurement model.

Assumption A5. The survival and detection probabilities are state independent.

Assumption A6. The intensities of the birth and spawn RFS:s are Gaussian mixtures.

As mentioned by Vo and Ma (2006), Assumption A1, A2, A4 and A5 are standard in many target tracking applications, see e.g. Bar-Shalom and Fortmann (1987). The third assumption is reasonable in applications where target interactions are negligible. Extended, more complete, remarks on the assumptions are given in Vo and Ma (2006).

The full prediction and correction equations are not repeated here, instead we refer the reader to the paper by Vo and Ma (2006). Convergence analysis of the GM-PHD filer is given in Clark and Vo (2007).

6.2.2 The cardinalized probability hypothesis density filter

A known drawback of the PHD filter is that the cardinality is estimated using a single parameter (the mean), resulting in the cardinality distribution being approximated with a Poisson distribution. Because the Poisson mean and variance are equal, when the true cardinality is high the corresponding estimate has a high variance. In practice, this results in an oversensitive cardinality estimate (Erdinc et al., 2005), e.g. seen when there are missed detections. To improve upon this, the cardinalized probability hypothesis density (CPHD) filter was introduced (Mahler, 2007a). In addition to propagating the PHD $D_{k|k}(\mathbf{x})$ in time, the CPHD filter also propagates the full cardinality distribution $P_{k|k}(n)$,

$$\begin{cases} D_0(\mathbf{x}) \\ P_0(n) \end{cases} \xrightarrow{c} \begin{cases} D_{0|0}(\mathbf{x}) \\ P_{0|0}(n) \end{cases} \xrightarrow{p} \begin{cases} D_{1|0}(\mathbf{x}) \\ P_{1|0}(n) \end{cases} \xrightarrow{c} \begin{cases} D_{1|1}(\mathbf{x}) \\ P_{1|1}(n) \end{cases} \xrightarrow{p} \cdots$$
$$\cdots \xrightarrow{c} \begin{cases} D_{k|k}(\mathbf{x}) \\ P_{k|k}(n) \end{cases} \xrightarrow{p} \begin{cases} D_{k+1|k}(\mathbf{x}) \\ P_{k+1|k}(n) \end{cases} \xrightarrow{c} \begin{cases} D_{k+1|k+1}(\mathbf{x}) \\ P_{k+1|k+1}(n) \end{cases} \xrightarrow{p} \cdots \tag{6.32}$$

To initialize the CPHD filter an initial PHD

$$D_0(\mathbf{x}) = n_0 \times s_0(\mathbf{x}) \tag{6.33}$$

and an initial cardinality distribution $P_0(n)$ have to be chosen (Mahler, 2007b), where $s_0(\mathbf{x})$ is a pdf with peaks that correspond to likely initial target locations, and $P_0(n)$ is a pmf definded on $n \in \mathbb{N} = \{0, 1, 2, 3, \ldots\}$ such that the expected value is n_0,

$$n_0 = \sum_{n=0}^{\infty} n P_0(n). \tag{6.34}$$

The CPHD prediction and correction equations are more intricate than their PHD counterparts, and they are not repeated here. However, we note that the CPHD prediction and correction require the assumption that the propagated multi-target distribution is approximated with an i.i.d. cluster process, cf. (6.14). A GM implementation of the CPHD filter for point targets can be found in the paper by Vo et al. (2007).

6.3 A brief revisit to the SLAM problem

Random finite sets were introduced in this chapter in part because they are a suitable remedy to the intricacies of performance evaluation of multiple object estimation, as highlighted in Section 5.3. Similarly to multiple target tracking, in SLAM the map state $\mathbf{M}$ consists of an unknown number of landmarks, each with an unknown state. Thus, modeling the landmarks as an RFS

$$\mathbf{M} = \left\{\mathbf{m}^{(i)}\right\}_{i=1}^{m}, \tag{6.35}$$

rather than a vector, as in (4.34), can simplify performance evaluation of the SLAM map. A further benefit of the RFS model is that the measurement to landmark data association can become more robust against high clutter rate and measurement noise.

An RFS formulation for SLAM was first proposed by Mullane et al. (2008), who model the robot state and map as a joint finite set valued variable and give a Gaussian mixture implementation of the proposed PHD-SLAM filter. A Rao-Blackwellized implementation of PHD-SLAM was given by Mullane et al. (2011), with early results presented by Mullane et al. (2010). The implemented PHD-SLAM filter is based on using the GM-PHD filter for the map and a particle filter for the robot trajectory. A similar approach is taken by Lee et al. (2012).

The RFS approaches to SLAM share most similarities with FAST-SLAM (Montemerlo et al., 2002). Comparisons show that the RFS approach outperforms FAST-SLAM in scenarios with high levels of clutter measurements.

7 Extended target tracking

In this chapter we revisit the target tracking problem, this time with an emphasis on a type of target that is called extended. We give a definition of extended target in Section 7.1 and provide some extended target models in Section 7.2. Measurement set partitioning is presented in Section 7.3, and performance evaluation for extended target estimates is discussed in Section 7.4.

7.1 Introduction

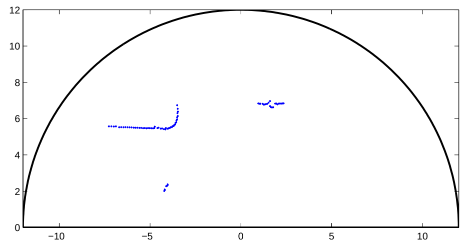
In many modern target tracking applications the point target assumption, see Definition 5.1 in Chapter 5, is not valid. Examples of such applications include vehicle tracking using automotive radars, people tracking using laser range sensors or object tracking using vision sensors, e.g. cameras. See Figure 7.1 for two examples. This prompts us to make the following definition:

Definition 7.1 (Extended target). A target that potentially gives rise to more than one measurement per time step.

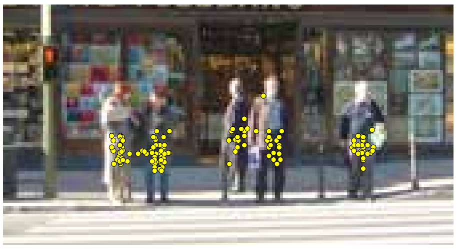
It is important to note here that the target tracking properties I to V listed in Section 5.1 apply to extended targets too. In addition to those properties, for extended target tracking it also holds that

VI the number of measurements generated by each target is unknown.

The multiple measurements per target raise interesting potentials for the estimation of target shape and size. While the single measurement setting is sufficient to track the targets' centers of mass, multiple measurements allow certain properties of the targets to be estimated too, e.g. shape, size and orientation. With this added knowledge, differentiation between different target types is possible.



(a)



(b)

Figure 7.1: *(a) Laser data with car, bicycle and pedestrian. (b) Camera image with pedestrian detections. Detections obtained using the classifier by Maji et al. (2008).*

7.2 Extended target modeling

Naturally there are more than one possible way to model extended targets. In this section we will briefly overview a few different alternatives.

7.2.1 Extended target measurements

Let ξ_k denote the extended target state at time t_k. Modeling the distribution of the set of target generated measurements means to model the distribution

$$p\left(\mathbf{Z}_k|\xi_k\right) = p\left(\mathbf{Z}_k|N_{z,k},\xi_k\right) p\left(N_{z,k}|\xi_k\right), \tag{7.1}$$

where the measurement set $\mathbf{Z}_k$ was defined in (5.1). The measurements $\mathbf{z}_k^{(j)}$ are often assumed to be i.i.d.,

$$p\left(\mathbf{Z}_k|N_{z,k},\xi_k\right) = \prod_{j=1}^{N_{z,k}} p\left(\mathbf{z}_k^{(j)}\middle|\,\xi_k\right), \tag{7.2}$$

where $p\left(\mathbf{z}_k|\,\xi_k\right)$ is a likelihood function for a single target generated measurement.

In the extended target model suggested by Gilholm and Salmond (2005) the number $N_{z,k}$ of target generated measurements is Poisson distributed with a rate parameter λ_k. The probability mass function for $N_{z,k}$ is

$$p\left(N_{z,k}|\lambda_k\right) = \mathcal{PS}\left(N_{z,k};\ \lambda_k\right) \tag{7.3a}$$

$$= \frac{e^{-\lambda_k}\lambda_k^{N_{z,k}}}{N_{z,k}!}. \tag{7.3b}$$

Each of the $N_{z,k}$ measurements are then distributed according to a spatial extent model. The analysis in Gilholm and Salmond (2005) is limited to the single target case, and a multiple hypothesis Kalman filter implementation and a particle filter implementation is given.

The model by Gilholm and Salmond (2005) has been extended to modeling the generation of target measurements as an inhomogeneous Poisson point process (Gilholm et al., 2005). The probability of n measurements falling in a region A of the surveillance space is

$$P\left(N_{z,k} = n\right) = \mathcal{PS}\left(n;\ \lambda_k(A)\right), \tag{7.4a}$$

where

$$\lambda_k(A) = \int_A \Lambda_k\left(\mathbf{z}\,|\xi_k\right) \mathrm{d}\mathbf{z} \tag{7.4b}$$

is the expected number of measurements falling in A, and $\Lambda_k\left(\mathbf{z}\,|\xi_k\right)$ is the spatially dependent intensity of the Poisson process. The likelihood function for a single measurement in the region A is

$$p\left(\mathbf{z}\,|\xi_k\right) = \frac{\Lambda_k\left(\mathbf{z}\,|\xi_k\right)}{\lambda_k(A)}, \tag{7.5}$$

i.e. given ξ_k, any measurement $\mathbf{z}$ is a random draw from this pdf. For this extended target model, Gilholm et al. (2005) gave a particle filter implementation for the multiple target case, and an extended target PHD filter for this model was developed by Mahler (2009).

A slightly different approach is taken by Swain and Clark (2010). They assume that at most one measurement is generated by a point target, however the point targets belong to groups, or clusters. The cluster centers are referred to as parent processes, and their point target processes are daughter processes. The resulting filter presented by Swain and Clark (2010) is similar to the extended target PHD filter (Mahler, 2009).

In all papers in the second part of this thesis, except Paper A and Paper F, the number of extended target generated measurements is modeled as Poisson distributed, with a target state dependent measurement rate $\lambda_k\left(\xi_k\right)$. In the next section, different models for the extended target state are given, together with corresponding measurement models.

7.2.2 Extended target state

For the extended target state, a common choice is to let the extended target state be a vector $\xi_k \in \mathbb{R}^{n_x}$ that is Gaussian distributed,

$$p\left(\xi_k\left|\mathbf{Z}^k\right.\right) = \mathcal{N}\left(\xi_k\,;\, m_{k|k}, P_{k|k}\right). \tag{7.6}$$

The measurement distribution is then typically modeled as

$$p\left(\mathbf{z}_k^{(j)}\middle|\,\xi_k\right) = \mathcal{N}\left(\mathbf{z}_k^{(j)}\,;\, h_k\left(\xi_k\right), R_k\right), \tag{7.7}$$

where $h_k(\cdot) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_z}$ is a non-linear measurement function. In this case, the extended state vector contains all states that are of interest, such as position, velocity, acceleration and the states that govern the shape and size of the extended target. A Gaussian state vector is used in Paper B and Paper C, and is also used by

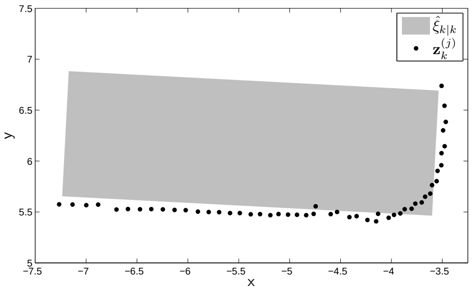


Figure 7.2: State estimate with rectangular shape for the extended target, and measurements from a car.

e.g. Salmond and Parr (2003); Gilholm and Salmond (2005); Baum et al. (2010b); Baum and Hanebeck (2011); Baum et al. (2011).

In Example 7.1 a Gaussian state vector is applied to the car data from Figure 7.1a, under the assumption that the target is shaped as a rectangle.

Example 7.1: Gaussian state

Consider the laser data in Figure 7.1a, which features measurements from a car, a bicycle and a pedestrian. For the car a rectangle appears to be a suitable shape model. Let the extended target state be a Gaussian distributed vector,

$$\xi_k = \left[\mathrm{x}_k\ ,\ \mathrm{y}_k\ ,\ \ell_k\ ,\ w_k\ ,\ \phi_k\right]^{\mathrm{T}} \tag{7.8}$$

where $(\mathrm{x}_k, \mathrm{y}_k)$ is the position, (ℓ_k, w_k) is the length and width of the rectangle, and ϕ_k is the orientation of the rectangle.

Using the knowledge that the laser range sensor measures along the edge of the shape, i.e. along the edges of the rectangle, Figure 7.2 shows the result of this extended target state applied to the car measurements from Figure 7.1a. The rectangle is not an exact fit to the measurements, however it is a reasonable approximation of the shape.

Note that for this type of example, modeling the measurement function $h_k(\,\cdot\,)$ can be quite complicated. Modeling of the measurement function for laser range data is considered in Paper C.

An alternative to the Gaussian model (7.6) was proposed by Koch (2008). This model is sometimes referred to as the random matrix model, or random matrix framework. The random matrix model defines the extended target state as the combination of a kinematical state vector $\mathbf{x}_k \in \mathbb{R}^{n_x}$, and an extension state matrix $X_k \in \mathbb{S}_{++}^{d}$ representing the size of the target. Modeling the extension as a random matrix means that the shape is modeled as elliptical.

The extended target state $\xi_k = (\mathbf{x}_k, X_k)$ is modeled as Gaussian inverse Wishart (GIW) distributed (Koch, 2008),

$$p\left(\xi_k \left| \mathbf{Z}^k\right.\right) = p\left(\mathbf{x}_k \left| X_k, \mathbf{Z}^k\right.\right) p\left(X_k \left| \mathbf{Z}^k\right.\right) \tag{7.9a}$$

$$= \mathcal{N}\left(\mathbf{x}_k \,;\, m_{k|k}, P_{k|k} \otimes X_k\right) \mathcal{IW}\left(X_k \,;\, v_{k|k}, V_{k|k}\right), \tag{7.9b}$$

and the measurement distribution can be modeled as

$$p\left(\mathbf{z}_k^{(j)} \middle| \xi_k\right) = \mathcal{N}\left(\mathbf{z}_k^{(j)} \,;\, h_k\left(\mathbf{x}_k\right), X_k\right). \tag{7.10}$$

Koch (2008) uses a linear function $h_k(\mathbf{x}_k) = H_k\mathbf{x}_k$, which results in linear correction for the extended target state estimate. The GIW model is used in Paper D, and is also used by e.g. Koch and Feldmann (2009); Wieneke and Koch (2010); Lan and Rong Li (2012).

In this model, the measurement covariance is given by the extension matrix, and the measurements are assumed to be spread across the target surface. The kinematic state vector contains states that are derivatives of the spatial state component, denoted $\mathbf{r}_k$ (Koch, 2008). For example, if the measurements are of the extended target's Cartesian $(\mathsf{x}_k, \mathsf{y}_k)$-position, then the spatial state component is $\mathbf{r}_k = [\mathsf{x}_k \,,\, \mathsf{y}_k]^{\mathrm{T}}$. In this case the kinematic state vector $\mathbf{x}_k$ contains the position $\mathbf{r}_k$, and possibly also higher derivatives of the spatial components such as velocity $\mathrm{d}\mathbf{r}_k/\mathrm{d}t_k$ and acceleration $\mathrm{d}^2\mathbf{r}_k/\mathrm{d}t_k^2$.

As mentioned above, in the random matrix model the extended target shape is assumed to be an ellipse (Koch, 2008). While this assumption is limiting, in many scenarios the ellipse is a sufficient approximation of the true extended target shape. Example 7.2 shows the GIW model applied to the laser data in Figure 7.1a.

Example 7.2: GIW state

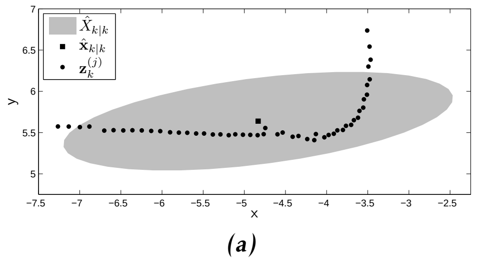
Consider again the laser data in Figure 7.1a. Let the extended target be GIW distributed, with kinematical state

$$\mathbf{x}_k = \left[\mathsf{x}_k \,,\, \mathsf{y}_k\right]^{\mathrm{T}}, \tag{7.11}$$

and extension state $X_k \in \mathbb{S}^2_{++}$. Using a linear measurement function $H_k\mathbf{x}_k = \mathbf{x}_k$, in Figure 7.3 this extended target state is applied to the car, bicycle and pedestrian measurements from Figure 7.1a, respectively.

For the bicycle and the pedestrian an ellipse appears to be a reasonable model for the shape, however it is a poor model for the car.

The measurement model (7.10) implicitly assumes that the sensor noise is negligible in comparison to the target extent (Feldmann and Fränken, 2008; Feldmann et al., 2011), a relation that does not hold in all scenarios.



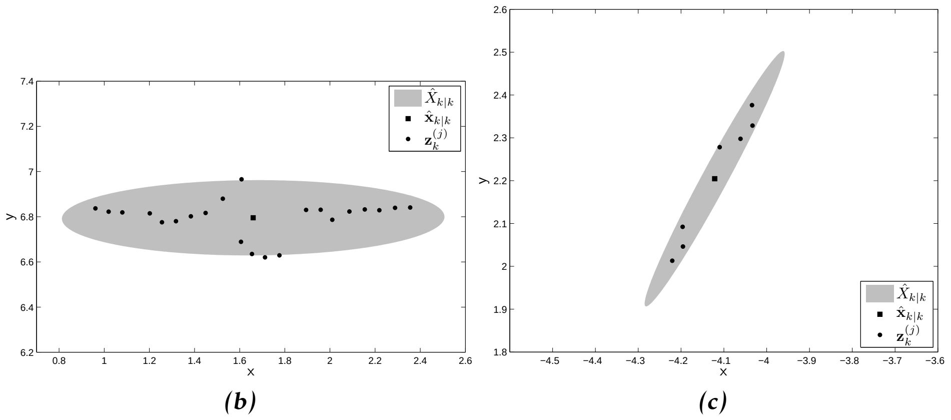


Figure 7.3: *The* GIW *extended target state model applied to the laser data from Figure 7.1a. (a), (b) and (c) Model applied to measurements from a car, a bicycle and a pedestrian, respectively.*

Feldmann and Fränken (2008), see also the work by Feldmann et al. (2011), proposed to approximate the kinematical and extension state as independent

$$p\left(\xi_k \,\middle|\mathbf{Z}^k\right) = p\left(\mathbf{x}_k \,\middle|X_k, \mathbf{Z}^k\right) p\left(X_k \,\middle|\mathbf{Z}^k\right) \tag{7.12a}$$

$$\approx p\left(\mathbf{x}_k \,\middle|\mathbf{Z}^k\right) p\left(X_k \,\middle|\mathbf{Z}^k\right) \tag{7.12b}$$

$$= \mathcal{N}\left(\mathbf{x}_k \,;\, m_{k|k}, P_{k|k}\right) \mathcal{IW}\left(X_k \,;\, v_{k|k}, V_{k|k}\right), \tag{7.12c}$$

with measurement distribution model

$$p\left(\mathbf{z}_k^{(j)}\middle|\, \xi_k\right) = \mathcal{N}\left(\mathbf{z}_k^{(j)} \,;\, H_k\mathbf{x}_k, zX_k + R_k\right), \tag{7.13}$$

where z is a scaling factor, and $R_k \in \mathbb{S}_+^d$. This measurement model can be interpreted as meaning that X_k is the true extension of the extended target, while R_k is the sensor error covariance matrix. Note that with this measurement distribution, the correction for the extended target state estimate is no longer linear, but has to be approximated (Feldmann and Fränken, 2008; Feldmann et al., 2011). For a comparison of the model (7.9), (7.10) and the model (7.12), (7.13), see the paper by Feldmann et al. (2011).

In comparison to (7.9), in the GIW model (7.12) the kinematic state vector can be defined independently of the spatial state component. This includes the possibility of states that represent, e.g., the heading and the turn-rate. This variant of the GIW model is used in Paper F and Paper G.

A third model for the measurement distribution is suggested by Lan and Rong Li (2012),

$$p\left(\mathbf{z}_k^{(j)}\middle|\,\xi_k\right) = \mathcal{N}\left(\mathbf{z}_k^{(j)}\,;\,H_k\mathbf{x}_k, B_k X_k B_k^{\mathrm{T}}\right), \tag{7.14}$$

where B_k is a $d \times d$ non-singular transformation matrix. The transformation that B_k represents could, e.g., be a rotation or a scaling of the extension matrix X_k. Setting the two models (7.13) and (7.14) equal, we have the following relation (Lan and Rong Li, 2012),

$$\begin{aligned} B_k X_k B_k^{\mathrm{T}} &= zX_k + R_k && (7.15\text{a}) \\ &= (zX_k + R_k)^{\frac{1}{2}}\, X_k^{-\frac{1}{2}} X_k X_k^{-\frac{\mathrm{T}}{2}}\, (zX_k + R_k)^{\frac{\mathrm{T}}{2}}\,, && (7.15\text{b}) \\ \Leftrightarrow B_k &= (zX_k + R_k)^{\frac{1}{2}}\, X_k^{-\frac{1}{2}}. && (7.15\text{c}) \end{aligned}$$

Under the assumption $X_k \approx \hat{X}_{k|k-1} = \mathrm{E}\left[X_k \,\middle|\mathbf{Z}^{k-1}\right]$, Lan and Rong Li (2012) approximate (7.15c) as

$$B_k \approx \left(z\hat{X}_{k|k-1} + R_k\right)^{\frac{1}{2}}\, \hat{X}_{k|k-1}^{-\frac{1}{2}}, \tag{7.16}$$

which gives a matrix B_k that is not a function of the extended target state.

Both the state vector representation (7.6) and the random matrix representation (7.9), (7.12) can be augmented with a state variable $\gamma_k > 0$, where γ_k is related to measurement generating Poisson rate as follows,

$$\lambda_k\left(\xi_k\right) = \gamma_k. \tag{7.17}$$

The conjugate prior for the Poisson rate is the gamma distribution, see Table 4.1 in Chapter 4. In this case it is natural to model $\xi_k = (\gamma_k, \mathbf{x}_k)$ as gamma Gaussian (GG) distributed,

$$\begin{aligned} p\left(\xi_k \,\middle|\mathbf{Z}^k\right) =& p\left(\gamma_k \,\middle|\mathbf{Z}^k\right) p\left(\mathbf{x}_k \,\middle|\mathbf{Z}^k\right) && (7.18\text{a}) \\ =& \mathcal{GAM}\left(\gamma_k\,;\,\alpha_{k|k}, \beta_{k|k}\right) \mathcal{N}\left(\mathbf{x}_k\,;\,m_{k|k}, P_{k|k}\right), && (7.18\text{b}) \end{aligned}$$

and to model $\xi_k = (\gamma_k, \mathbf{x}_k, X_k)$ as gamma Gaussian inverse Wishart (GGIW) distributed,

$$\begin{aligned} p\left(\xi_k \,\middle|\mathbf{Z}^k\right) =& p\left(\gamma_k \,\middle|\mathbf{Z}^k\right) p\left(\mathbf{x}_k \,\middle|X_k, \mathbf{Z}^k\right) p\left(X_k \,\middle|\mathbf{Z}^k\right) && (7.19\text{a}) \\ =& \mathcal{GAM}\left(\gamma_k\,;\,\alpha_{k|k}, \beta_{k|k}\right) \mathcal{N}\left(\mathbf{x}_k\,;\,m_{k|k}, P_{k|k} \otimes X_k\right) \mathcal{IW}\left(X_k\,;\,v_{k|k}, V_{k|k}\right), && (7.19\text{b}) \end{aligned}$$

or

$$p\left(\xi_k \left|\mathbf{Z}^k\right.\right) = p\left(\gamma_k \left|\mathbf{Z}^k\right.\right) p\left(\mathbf{x}_k \left|\mathbf{Z}^k\right.\right) p\left(X_k \left|\mathbf{Z}^k\right.\right) \tag{7.20a}$$

$$= \mathcal{GAM}\left(\gamma_k\, ;\, \alpha_{k|k}, \beta_{k|k}\right) \mathcal{N}\left(\mathbf{x}_k\, ;\, m_{k|k}, P_{k|k}\right) \mathcal{IW}\left(X_k\, ;\, v_{k|k}, V_{k|k}\right). \tag{7.20b}$$

A Bayesian recursion for estimation of the Poisson rate γ_k is given in Paper E. The GGIW model (7.19) is used in Paper H.

7.2.3 Extension shape models

Closely related to modeling the distribution of the extended target measurements is modeling the extended target shape. Dezert (1998) models the extension as a collection of points, which together with the target kinematics are jointly estimated from the measurements. In this case the target shape is given by the shape of the points. Gilholm and Salmond (2005) give an example in which the target extension is an infinitely thin stick of length ℓ, a similar example is used by Boers et al. (2006).

The GIW measurement models suggested by Koch (2008), Feldmann et al. (2011), and Lan and Rong Li (2012), assume that the extended target is shaped like an ellipse, and that the measurements are spread across the target surface. Inspired by Koch (2008), Degerman et al. (2011) decompose the extension matrix into principal components, and design a heuristic Kalman filter for tracking the extension. Zhu et al. (2011) model the extension as a combination of two ellipses, more specifically two Gaussian distributions. The random hypersurface model by Baum and Hanebeck (2009) models the measurements as random samples of measurement generating points on the target surface. A random hypersurface model is given for elliptic targets by Baum et al. (2010b), and a comparison between the elliptic random hypersurface model and the GIW model is given by Baum et al. (2010a).

Modeling the measurements as being spread across the target surface is appropriate, e.g., when airborne radars are used to track ground vehicles. For other sensors, e.g. the laser range sensor, the measurements are better modeled as being spread along the edge of the target surface. Rectangular and elliptical shape models are given for the laser range sensor in Paper C, Lundquist et al. (2011a) give a more general shape model for laser range type sensors, capable of estimating arbitrary shapes. Note that the methods presented by Lundquist et al. (2011a) and in Paper C are capable of estimating the entire extension, even when only parts of the extension are seen. For similar scenarios, Petrov et al. (2011) give a sampling based measurement model for extended targets whose extensions are measured across so called regions of interest. They illustrate their approach by considering circular objects. Lundquist et al. (2011b) model the extended targets using polynomials, an approach that is shown to be applicable to road-mapping using vehicle radars.

7.3 Measurement set partitioning

In Chapter 5 the data association problem was mentioned, and some data association methods for multiple point target tracking in clutter were given in Section 5.2. The data association problem must be solved also for multiple extended target tracking in clutter. Because more than one measurement may originate from the same target, one approach to solving the data association problem is to divide the set of measurements into non-empty subsets, where each subset is assumed to contain measurement that are all from the same source. The subsets can then be associated to the extended targets, similarly to how single measurements are associated to the point targets.

In this thesis, we refer to the division of the measurement set into non-empty subsets as partitioning the measurement set, a particular partitioning of the measurement set is called a partition, and the non-empty subsets are denoted cells. Note that in the literature, partitioning is sometimes called clustering or cluster analysis, and the cells are then typically called clusters. For a given partition, the cells can be interpreted as containing measurements that all stem from either a single extended target or a clutter source.

There are different ways to approach the measurement partitioning, e.g. the measurement-to-cell association could be either hard or soft. Hard association means that the measurement belongs to the cell or not, similarly to NN and GNN data association. Soft association means that the measurement belongs to the cell to a certain degree. This thesis will only consider hard measurement-to-cell associations.

When it comes to computing the partitions, one method is to assume that the partition should have K cells, and then assign the measurements to the K cells by minimizing a cost function. One such partitioning method is K-means clustering, see e.g. the textbooks by Bishop (2006); Hastie et al. (2009). K-means clustering is a type of combinatorial algorithm (Hastie et al., 2009), meaning that it works directly on the data and does not have an underlying probability model. Given a desired number of cells, the algorithm assigns the data to the cells by iteratively minimizing a cost function. One of K-means clustering's drawbacks is its tendency to get stuck in local optimas during the cost function minimization. An improved version, called K-means++ clustering, is reported to have an initialization that better avoids local optimas (Arthur and Vassilvitskii, 2007; Ostrovsky et al., 2006).

Alternatively, one may define a criterion by which it is determined whether, or not, two measurements belong to the same cell. One such criterion is the distance between the measurements, as given by a distance measure, e.g. the Euclidean metric or Mahalanobis distance. This is also known as hierarchical clustering (Hastie et al., 2009), and there are two types of hierarchical methods: agglomerative and divisive. Agglomerative is bottom-up, i.e. it starts with all measurements in one cell each, and builds larger cells (Hastie et al., 2009). Divisive is top-down, i.e. it starts with all measurements in the same cell and splits into smaller cells

(Hastie et al., 2009).

In terms of the distance used to determine cell membership, two alternatives are complete linkage and single linkage. Let W_1 and W_2 be two cells whose union is empty, i.e. the two cells do not have any measurements in common. For complete linkage, the distance between the cells is measured as

$$\max\left\{d\left(\mathbf{z}_k^{(i)}\,,\;\mathbf{z}_k^{(j)}\right):\mathbf{z}_k^{(i)}\in W_1\,,\;\mathbf{z}_k^{(j)}\in W_2\right\}, \tag{7.21}$$

where $d(\,\cdot\,,\,\cdot\,)$ is the distance measure used. This means that the distance between the two cells is the maximum distance between a pair of measurements. For single linkage, the distance between the cells is measured as

$$\min\left\{d\left(\mathbf{z}_k^{(i)}\,,\;\mathbf{z}_k^{(j)}\right):\mathbf{z}_k^{(i)}\in W_1\,,\;\mathbf{z}_k^{(j)}\in W_2\right\}. \tag{7.22}$$

In this case the distance between the two cells is the minimum distance between a pair of measurements.

In Paper B a type of agglomerative, single linkage, hierarchical partitioning method, called Distance Partitioning, is proposed for use in multiple extended target tracking. Distance Partitioning forms the basis for the measurement set partitioning that is used in this thesis, however additional methods are also proposed in Paper B and Paper D. Example 7.3 gives a small comparison of Distance Partitioning and K-means++ clustering.

Example 7.3: Measurement set partitioning

True target measurements were generated from three Gaussian distributions with the following means and covariances,

$$m^{(1)} = [0\ 0]^{\mathrm{T}}, \qquad P^{(1)} = \mathrm{diag}\left([1\ 1]\right), \tag{7.23a}$$
$$m^{(2)} = [15\ 0]^{\mathrm{T}}, \qquad P^{(2)} = \mathrm{diag}\left([0.25\ 1]\right), \tag{7.23b}$$
$$m^{(3)} = [0\ 15]^{\mathrm{T}}, \qquad P^{(3)} = \mathrm{diag}\left([1\ 0.25]\right). \tag{7.23c}$$

In total, 10 measurements were sampled from each distribution, let $\mathbf{Z}^T$ denote the set of 30 target measurements. Further, 10 clutter measurements were generated by uniform sampling in $[-5\,,\;20]\times[-5\,,\;20]$. Let $\mathbf{Z}^C$ denote the set of 10 clutter measurements, and let $\mathbf{Z}^{TC}$ denote the union of $\mathbf{Z}^T$ and $\mathbf{Z}^C$.

The measurements are shown in Figure 7.4a. Figure 7.4b and Figure 7.4c show $\mathbf{Z}^T$ and $\mathbf{Z}^{TC}$, respectively, after partitioning with Distance Partitioning with threshold 2, measurements with the same color belong to the same cell. We see that Distance Partitioning gives correct cells for the target generated measurements, and places the clutter measurements in individual cells in Figure 7.4c. Both these partitions are quite intuitive, and correspond well to our desire to have a partition in which the cells contain measurements that all stem from either a single extended target or a clutter source.

In Figure 7.4d $\mathbf{Z}^T$ is shown after partitioning with K-means++ clustering, we see that the result is the same partition as when Distance Partitioning is used, see

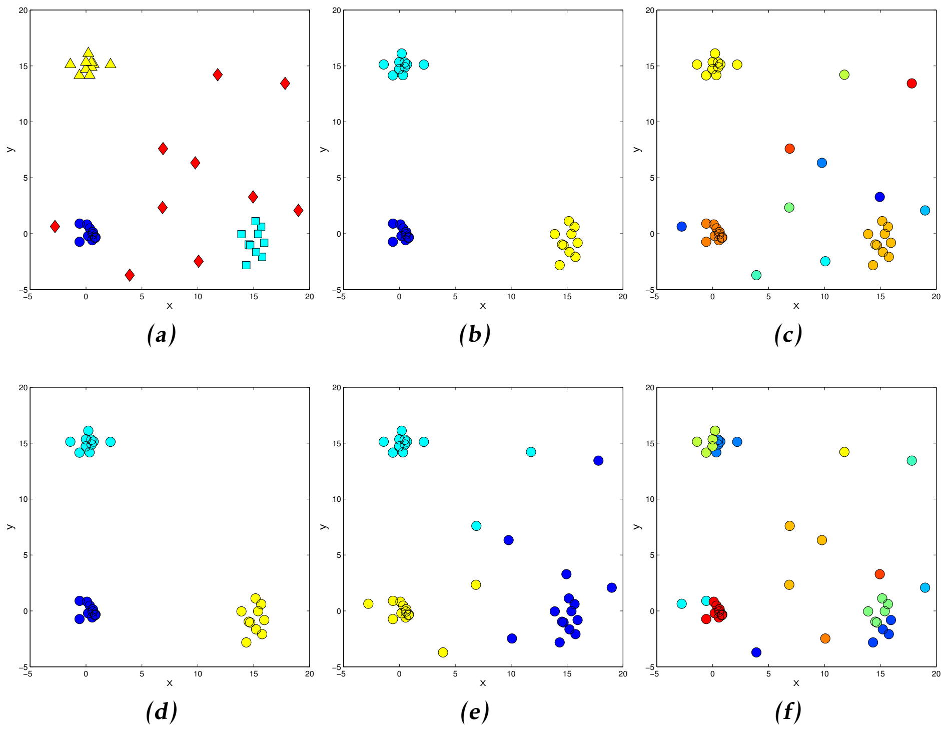


Figure 7.4: *Comparison of clustering methods. Measurements with the same color belong to the same cell. (a) True target measurements shown as blue circles, cyan squares and yellow triangles, respectively, and clutter measurements shown as red diamonds. (b) Partition of $\mathbf{Z}^T$ computed using Distance Partitioning, threshold 2. (c) Partition of $\mathbf{Z}^{TC}$ computed using Distance Partitioning, threshold 2. (d) Partition of $\mathbf{Z}^T$ computed using K-means++ clustering, $K = 3$. (e) Partition of $\mathbf{Z}^{TC}$ computed using K-means++ clustering, $K = 3$. (f) Partition of $\mathbf{Z}^{TC}$ computed using K-means++ clustering, $K = 13$.*

Figure 7.4b. However, when K-means++ clustering is used for partitioning of $\mathbf{Z}^{TC}$, the results are much less intuitive. The partitions, for $K = 3$ and $K = 13$ are shown in Figure 7.4e and Figure 7.4f. For this measurement set, K-means++ does not return the correct partition for any value of K between 3 and 13.

7.4 Performance evaluation

In terms of multiple extended target tracking, there is no conceptual difference to multiple point target tracking that prevents the use of the OSPA metric (see Section 5.3). However, the OSPA requires a metric $d(\cdot,\cdot)$ for comparison of a true extended target state and a state estimate. In this section we will discuss some alternatives for performance evaluation of a single extended target estimate.

In Section 4.5 the RMSE and NEES were introduced as performance metrics for state vector estimates, theses metrics are no less valid for an extended target state vector. However, depending on the particular modeling framework used, the extension states could benefit from being treated differently.

For extended targets whose extension parameters are included in the state vector, cf. (7.6), the RMSE and NEES can naturally be used also for the states related to the extension. In case the extended target state is decomposed as in (7.9) or (7.12), the extension estimation error can be evaluated using a matrix norm, e.g. the Frobenius norm,

$$\left\|X_k - \hat{X}_{k|k}\right\|_F = \sqrt{\sum_{i=1}^{d}\sum_{j=1}^{d}\left|X_k^{[ij]} - \hat{X}_{k|k}^{[ij]}\right|^2}, \tag{7.24}$$

where $d \times d$ is the dimension of the extension matrix, and the notation $A^{[ij]}$ is used to denote the i, jth element of the matrix A. The Frobenius norm for matrices is analogous to the Euclidean norm for vectors. It is used for performance evaluation in Paper G and Paper H.

Another performance metric, often used in computer vision, is a difference measure called Intersection-Over-Union (IOU). The IOU measures the volumes of the intersection and the union of the true extended target and the estimate, and then takes the ratio of the two volumes. Note that the IOU is not a metric, e.g. it does not satisfy the triangle inequality. Further, in comparison to the RMSE, NEES and Frobenius norm, who are all equal to zero when there is no error, the IOU is equal to one when there is no error. The IOU measure is used for performance evaluation in Paper C.

In case a measurement rate γ_k is estimated, a suitable performance metric is the absolute difference,

$$\left|\gamma_k - \hat{\gamma}_{k|k}\right|. \tag{7.25}$$

Example 7.4 compares performance metrics for GIW distributed extended targets.

Example 7.4: Extension estimation performance evaluation

Let the extended target state be GIW distributed, with true state

$$\xi_k = (\mathbf{x}_k\ ,\ X_k), \tag{7.26a}$$

$$\mathbf{x}_k = [0\ ,\ 0]^{\mathrm{T}}, \tag{7.26b}$$

$$X_k = R\left(\frac{45}{180}\pi\right)\operatorname{diag}\left(\left[5^2\ ,\ 2^2\right]\right)R^{\mathrm{T}}\left(\frac{45}{180}\pi\right), \tag{7.26c}$$

where $R(\,\cdot\,)$ is a 2D rotation matrix. There are two state estimates,

$$\hat{\xi}^{(1)}_{k|k} = \left(\hat{\mathbf{x}}^{(1)}_{k|k}\,,\ \hat{X}^{(1)}_{k|k}\right), \tag{7.27a}$$

$$\hat{\mathbf{x}}^{(1)}_{k|k} = [-0.25\,,\ 0.15]^{\mathrm{T}}, \tag{7.27b}$$

$$\hat{X}^{(1)}_{k|k} = R\left(\frac{35}{180}\pi\right)\operatorname{diag}\left(\left[6^2\,,\ 1.75^2\right]\right)R^{\mathrm{T}}\left(\frac{35}{180}\pi\right), \tag{7.27c}$$

and

$$\hat{\xi}^{(2)}_{k|k} = \left(\hat{\mathbf{x}}^{(2)}_{k|k}\,,\ \hat{X}^{(2)}_{k|k}\right), \tag{7.28a}$$

$$\hat{\mathbf{x}}^{(2)}_{k|k} = [-1\,,\ 1]^{\mathrm{T}}, \tag{7.28b}$$

$$\hat{X}^{(2)}_{k|k} = R\left(\frac{45}{180}\pi\right)\operatorname{diag}\left(\left[5.1^2\,,\ 2.1^2\right]\right)R^{\mathrm{T}}\left(\frac{45}{180}\pi\right). \tag{7.28c}$$

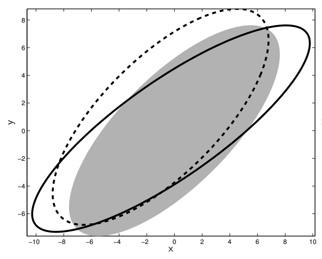
The true state and the estimates are shown in Figure 7.5a, the intersections and the unions of the 2σ-ellipses are shown in Figure 7.5b and Figure 7.5c. The Euclidean norms for the kinematical state vector differences, the Frobenius norms for the extension matrix differences, and the IOU values are

$$\left\|\mathbf{x}_k - \hat{\mathbf{x}}^{(1)}_{k|k}\right\|_2 = 0.29, \qquad \left\|\mathbf{x}_k - \hat{\mathbf{x}}^{(2)}_{k|k}\right\|_2 = 1.41, \tag{7.29a}$$

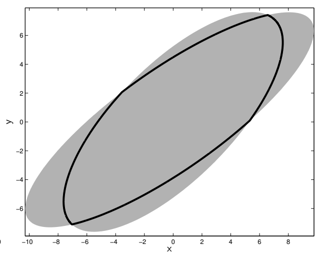
$$\left\|X_k - \hat{X}^{(1)}_{k|k}\right\|_F = 12.79, \qquad \left\|X_k - \hat{X}^{(2)}_{k|k}\right\|_F = 1.09, \tag{7.29b}$$

$$\text{IOU} = 0.70, \qquad \text{IOU} = 0.64. \tag{7.29c}$$

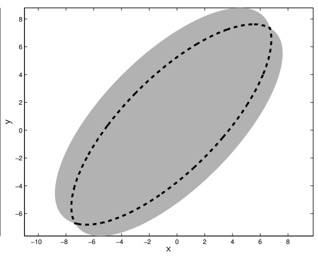
Determined by the IOU measure, $\hat{\xi}^{(1)}_{k|k}$ is the better estimate. To be able to determine by the Euclidean and Frobenius norms, the two norms would have to be weighed together. Depending on how this is performed, either estimate could be the better one.



(a)



(b)



(c)

Figure 7.5: *Illustration of the difference measure intersection over union. The 2σ-ellipses are plotted. (a) True extended target state ξ_k (gray area), state estimate $\hat{\xi}^{(1)}_{k|k}$ (black solid line), and state estimate $\hat{\xi}^{(2)}_{k|k}$ (black dashed line). (b) The intersection (black solid line) and the union (gray area) of ξ_k and $\hat{\xi}^{(1)}_{k|k}$. (c) The intersection (black dashed line) and the union (gray area) of ξ_k and $\hat{\xi}^{(2)}_{k|k}$.*

8 Concluding remarks

This chapter summarizes the thesis, with conclusions in Section 8.1 and recommendations for future work in Section 8.2.

8.1 Conclusions

A method for loop closure detection in SLAM using data from laser sensors was presented. A compact and efficient feature description of each point cloud is given, and AdaBoost is used to construct a classifier that uses the features to classify a point cloud pair as being either from the same location, or not. The classifier is able to detect loop closure from arbitrary direction, and in experiments it is shown to produce detection rates that compare well to related work at low false alarm rates. The real world SLAM experiments showed that the classifier can be used within the context for which it was constructed.

Two different implementations of the extended target probability hypothesis density filter was presented, one Gaussian mixture implementation and one Gaussian inverse Wishart implementation. The Gaussian mixture implementation was also extended to non-linear motion and measurement models. The ideal filter requires consideration of the full set of partitions, which is computationally unfeasible in all but the very simplest cases. It was shown that the full set of partitions can be approximated with a subset of partitions, without having to sacrifice tracking performance. Four different partitioning methods were suggested, and it was shown that they reduce the number of partitions considered by several orders of magnitude, and also that they outperform the well known partitioning method K-means clustering.

It was shown in simulations and experiments that the case of spatially close targets can be difficult to handle. Suitable remedies to improve performance under these circumstances were suggested, and results showed that performance was improved. Further, it was shown that the filter is sensitive to the number of measurements generated by each extended target. A framework for estimating the number of measurements generated was presented, and shown to be capable of estimating an individual measurement rate for each target. The complexity of the implementations increases exponentially with time, and approximations are necessary to ensure computational feasibility. To this end, merging of distribution mixtures was presented. Finally, extended target prediction, spawning and combination was also addressed.

8.2 Future work

A noted drawback of the presented loop detection classifier is that it, compared to related work, is more sensitive to translation, i.e. required a higher degree of point cloud overlap. While this is not problematic in environments with well defined pathways, such as road networks or office hallways, it would present a challenge in environments with less restrictions on motion. A topic for future work is to increase the classifiers ability to handle translation.

The underlying ideas behind the presented loop detection classifier could have extensions to the environment labeling problem. Environment labeling is a problem in which the parts of the sensor data is labeled according to which class it originated from, e.g. ground, vegetation, building walls, cars, humans, etc. Features similar to the ones used to describe the point clouds as a whole could be used as local descriptors of each point in the point clouds. Using a multi-class classifier, the data point could then be labeled with the most likely class label.

Two different models for estimation of the size and shape of extended targets were used, however they were both limited to simpler shapes. A comparison of different models for the shape and size of the extended targets would be interesting. Given a partition of the current measurement set, and prior extended target estimates, the cell to target association problem is similar to classic point target tracking. It would be interesting to see how the partitioning methods could be used together with classic target tracking approaches, like Multiple Hypothesis Tracking, to construct multiple extended target tracking algorithms. If such a tracking algorithm can be devised, a comparison to the extended target PHD filter could show the advantages and disadvantages of using random set theory.

Further work on measurement set partitioning can be undertaken, to ensure that the filtering framework is capable of handling multiple extended targets that maneuver close to each other in heavily cluttered measurement data. When targets are spatially close, the suggested partitioning methods sometimes fail in producing correct partitions. A method capable of detecting that a cell contains measurement that actually belong to multiple sources could improve performance.

Regarding estimation of the measurement rates, better models for the relation between the measurement rate and the target kinematics and target extension is needed. Reduction of distribution mixtures is addressed using merging, and a simple way to construct a pairwise criterion is suggested. The related literature on Gaussian mixture reduction contains different approaches to the problem of finding which components in the mixture should be merged, and which should not be merged. A comparison of these approaches applied to other distribution mixtures would be interesting.

The presented approach to prediction of an extended target modeled with random matrices can be tested further in multiple target scenarios. Only one step prediction is considered in the paper, however the case of multi-step prediction could further show the merits of the prediction. It would also be interesting to include the prediction in an interacting multiple model framework. The spawning and combination of extended targets only handles the two target case, an extension to the case of an arbitrary number of targets would be interesting.

Somewhere along the intersection of mapping and target tracking lies the problem of separating the sensor data into segments that correspond to either stationary or moving objects. This thesis has not handled the segmentation problem, although it is shown that the loop detection classifier is not sensitive to moving objects. An extended target tracking framework could possibly be used to determine which parts of the surrounding environment are stationary, and which parts are moving.

Bibliography

D. Arthur and S. Vassilvitskii. k-means++: The Advantages of Careful Seeding. In *Proceedings of the ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035, Philadelphi, PA, USA, January 2007.

T. Bailey and H. F. Durrant-Whyte. Simultaneous localization and mapping (SLAM): part II. *IEEE Robotics & Automation Magazine*, 13(3):108–117, September 2006.

T. Bailey, J. I. Nieto, and E. Nebot. Consistency of the FastSLAM algorithm. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 424–429, May 2006.

Y. Bar-Shalom. *Multitarget-multisensor tracking: applications and advances*, volume II of *Multitarget-multisensor Tracking: Applications and Advances*. Artech House, 1992.

Y. Bar-Shalom and T. E. Fortmann. *Tracking and data association*, volume 179 of *Mathematics in Science and Engineering*. Academic Press Professional, Inc., San Diego, CA, USA, 1987.

Y. Bar-Shalom and X. Rong Li. *Multitarget-Multisensor Tracking: Principles and Techniques*. YBS, 1995.

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

Y. Bar-Shalom, P. K. Willett, and X. Tian. *Tracking and data fusion, a handbook of algorithms*. YBS, 2011.

M. Baum and U. D. Hanebeck. Random hypersurface models for extended object tracking. In *IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pages 178–183, Ajman, United Arab Emirates, December 2009.

M. Baum and U. D. Hanebeck. Shape Tracking of Extended Objects and Group Targets with Star-Convex RHMs. In *Proceedings of the International Confer-*

ence on Information Fusion (FUSION), pages 338–345, Chicago, IL, USA, July 2011.

M. Baum, M. Feldmann, D. Fränken, U. D. Hanebeck, and J. W. Koch. Extended object and group tracking: A comparison of random matrices and random hypersurface models. In *Proceedings of the IEEE ISIF Workshop on Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, Leipzig, Germany, October 2010a.

M. Baum, B. Noack, and U. D. Hanebeck. Extended Object and Group Tracking with Elliptic Random Hypersurface Models. In *Proceedings of the International Conference on Information Fusion (FUSION)*, Edinburgh, UK, July 2010b.

M. Baum, B. Noack, and U. D. Hanebeck. Random hypersurface mixture models for tracking multiple extended objects. In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, Orlando, Florida, USA, December 2011.

P. J. Besl and N. D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

P. Biber and W. Strasser. The normal distribution transform: A new approach to laser scan matching. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2743–2748, Las Vegas, USA, October 2003.

C. M. Bishop. *Pattern recognition and machine learning*. Springer, New York, NY, USA, 2006.

S. Blackman and R. Popoli. *Design and Analysis of Modern Tracking Systems*. Artech House, Norwood, MA, USA, 1999.

Y. Boers, H. Driessen, J. Torstensson, M. Trieb, R. Karlsson, and F. Gustafsson. A track before detect algorithm for tracking extended targets. *IEE Proceedings Radar, Sonar and Navigation*, 153(4):345–351, August 2006.

M. C. Bosse and R. Zlot. Map matching and data association for large-scale two-dimensional laser scan-based SLAM. *International Journal of Robotics Research*, 27(6):667–691, June 2008.

J. Callmer, K. Granström, J. I. Nieto, and F. T. Ramos. Tree of Words for Visual Loop Closure Detection in Urban SLAM. In *Proceedings of the Australian Conference on Robotics & Automation (ACRA)*, Canberra, Australia, December 2008.

M. Campbell, E. Garcia, D. Huttenlocher, I. Miller, P. Moran, A. Nathan, B. Schimpf, N. Zych, J. Catlin, F. Chelarescu, H. Fujishima, F.-R. Kline, S. Lupashin, M. Reitmann, A. Shapiro, and J. Wong. Team cornell: Technical review of the darpa urban challenge vehicle, 2007. URL `http://www.darpa.mil/grandchallenge/TechPapers/Team_Cornell.pdf`.

Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, April 1992.

D. E. Clark and J. Bell. Convergence results for the particle PHD filter. *IEEE Transactions on Signal Processing*, 54(7):2652–2661, July 2006.

D. E. Clark and B.-N. Vo. Convergence analysis of the Gaussian mixture PHD filter. *IEEE Transactions on Signal Processing*, 55(4):1204–1212, April 2007.

J. Degerman, J. Wintenby, and D. Svensson. Extended target tracking using principal components. In *Proceedings of the International Conference on Information Fusion (FUSION)*, Chicago, IL, USA, July 2011.

J. C. Dezert. Tracking maneuvering and bending extended target in cluttered environment. In *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets*, pages 283–294, Orlando, FL, USA, September 1998.

M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba. A Solution to the Simultaneous Localization and Map Building (SLAM) Problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, June 2001.

H. F. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping (SLAM): part I. *IEEE Robotics & Automation Magazine*, 13(2):99–110, June 2006.

O. Erdinc, P. Willett, and Y. Bar-Shalom. Probability hypothesis density filter for multitarget multisensor tracking. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 146–153, Philadelphia, CA, USA, july 2005.

R. M. Eustice, H. Singh, and J. J. Leonard. Exactly sparse delayed-state filters for view-based SLAM. *IEEE Transactions on Robotics*, 22(6):1100–1114, December 2006.

M. Feldmann and D. Fränken. Tracking of Extended Objects and Group Targets using Random Matrices - A New Approach. In *Proceedings of the International Conference on Information Fusion (FUSION)*, Cologne, Germany, July 2008.

M. Feldmann, D. Fränken, and J. W. Koch. Tracking of extended objects and group targets using random matrices. *IEEE Transactions on Signal Processing*, 59(4):1409–1420, April 2011.

Y. Freund and R. E. Shapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of European Conference on Computational Learning Theory (EuroCOLT)*, pages 23–37, Barcelona, Spain, March 1995.

J. H. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28(2):337–407, 2000.

B. Gates. A Robot in Every Home. *Scientific American*, 296(1):58–65, January 2007.

A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Texts in Statistical Science. Chapman & Hall/CRC, 2004.

K. Gilholm and D. Salmond. Spatial distribution model for tracking extended objects. *IEE Proceedings Radar, Sonar and Navigation*, 152(5):364–371, October 2005.

K. Gilholm, S. Godsill, S. Maskell, and D. Salmond. Poisson models for extended target and group tracking. In *Proceedings of Signal and Data Processing of Small Targets*, volume 5913, pages 230–241, San Diego, CA, USA, August 2005. SPIE.

N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings on Radar and Signal Processing*, 140(2):107–113, April 1993.

K. Granström and U. Orguner. A PHD filter for tracking multiple extended targets using random matrices. *IEEE Transactions on Signal Processing*, 2012a. doi: 10.1109/TSP.2012.2212888.

K. Granström and U. Orguner. A New Prediction Update for Extended Target Tracking with Random Matrices. *IEEE Transactions on Aerospace and Electronic Systems*, 2012b.

K. Granström and U. Orguner. Estimation and Maintenance of Measurement Rates for Multiple Extended Target Tracking. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 2170–2176, Singapore, July 2012c.

K. Granström and U. Orguner. On the Reduction of Gaussian inverse Wishart mixtures. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 2162–2169, Singapore, July 2012d.

K. Granström and U. Orguner. On Spawning and Combination of Extended/Group Targets Modeled with Random Matrices. *IEEE Transactions on Signal Processing*, 2012e.

K. Granström and T. B. Schön. Learning to Close the Loop from 3D Point Clouds. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2089–2095, Taipei, Taiwan, October 2010.

K. Granström, J. Callmer, F. T. Ramos, and J. I. Nieto. Learning to Detect Loop Closure from Range Data. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22, Kobe, Japan, May 2009.

K. Granström, C. Lundquist, and U. Orguner. A Gaussian Mixture PHD filter for Extended Target Tracking. In *Proceedings of the International Conference on Information Fusion (FUSION)*, Edinburgh, UK, July 2010.

K. Granström, C. Lundquist, and U. Orguner. Tracking Rectangular and Elliptical Extended Targets Using Laser Measurements. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 592–599, Chicago, IL, USA, July 2011.

K. Granström, C. Lundquist, and U. Orguner. Extended Target Tracking using a Gaussian Mixture PHD filter. *IEEE Transactions on Aerospace and Electronic Systems*, 2012.

K. Granström, T. B. Schön, J. I. Nieto, and F. T. Ramos. Learning to close loops from range data. *The International Journal of Robotics Research*, 30(14):1728–1754, December 2011.

K. Granström, C. Lundquist, F. Gustafsson, and U. Orguner. On extended target tracking using PHD filters. In *Workshop on Stochastic Geometry in SLAM at IEEE International Conference on Robotics and Automation (ICRA)*, St. Paul, Minnesota, USA, May 2012.

M. Guerriero, L. Svensson, D. Svensson, and P. Willett. Shooting two birds with two bullets: how to find Minimum Mean OSPA estimates. In *Proceedings of the International Conference on Information Fusion (FUSION)*, Edinburgh, UK, July 2010.

F. Gustafsson. *Statistical Sensor Fusion*. Studentlitteratur, 2010.

T. Hastie, R. Tibshirani, and J. H. Friedman. *The elements of statistical learning : Data mining, inference, and prediction*. Springer, New York, 2 edition, 2009.

A. H. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, 1970.

A.M. Johansen, S.S. Singh, A. Doucet, and B.-N. Vo. Convergence of the smc implementation of the phd filter. *Methodology and Computing in Applied Probability*, 8(2):265–291, 2006.

R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME - Journal of Basic Engineering*, 82(Series D):35–45, March 1960.

J. W. Koch. Bayesian approach to extended object and cluster tracking using random matrices. *IEEE Transactions on Aerospace and Electronic Systems*, 44 (3):1042–1059, July 2008.

J. W. Koch and M. Feldmann. Cluster tracking under kinematical constraints using random matrices. *Robotics and Autonomous Systems*, 57(3):296–309, March 2009.

J. Lan and X. Rong Li. Tracking of extended object or target group using random matrix – part I: New model and approach. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 2177–2184, Singapore, July 2012.

C. S. Lee, D.E. Clark, and J. Salvi. Slam with single cluster phd filters. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2096–2101, May 2012.

L. Ljung. *System Identification Theory for the User*. Prentice Hall, 2 edition, 1999.

F. Lu and E. Milios. Globally Consistent Range Scan Alignment for Environment Mapping. *Autonomous Robots*, 4(4):333–349, October 1997.

C. Lundquist, K. Granström, and U. Orguner. Estimating the shape of targets with a PHD filter. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 49–56, Chicago, IL, USA, July 2011a.

C. Lundquist, U. Orguner, and F. Gustafsson. Extended target tracking using polynomials with applications to road-map estimation. *IEEE Transactions on Signal Processing*, 59(1):15–26, January 2011b.

C. Lundquist, K. Granström, and U. Orguner. An extended target CPHD filter and a gamma Gaussian inverse Wishart implementation. *Journal of Selected Topics in Signal Processing*, 2012a.

C. Lundquist, M. A. Skoglund, K. Granström, and T. Glad. Insights from implementing a system for peer review. *IEEE Transactions on Education*, 2012b. doi: 10.1109/TE.2012.2211876.

M. Magnusson, T. Duckett, and A. J. Lilienthal. Scan registration for autonomous mining vehicles using 3D-NDT. *jfr*, 24(10):803–827, October 2007.

R. P. S. Mahler. PHD filters of higher order in target number. *IEEE Transactions on Aerospace and Electronic Systems*, 43(4):1523–1543, October 2007a.

R. P. S. Mahler. *Statistical Multisource-Multitarget Information Fusion*. Artech House, Norwood, MA, USA, 2007b.

R. P. S. Mahler. PHD filters for nonstandard targets, I: Extended targets. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 915–921, Seattle, WA, USA, July 2009.

S. Maji, A. C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Anchorage, Alaska, USA, June 2008.

J. Markoff. Virtual and artificial, but 58,000 want course, August 2011. URL `http://www.nytimes.com/2011/08/16/science/16stanford.html`.

M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, Edmonton, Canada, September 2002.

J. Mullane, B.-N. Vo, M.D. Adams, and W.S. Wijesoma. A random set formulation for Bayesian SLAM. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1043–1049, September 2008.

J. Mullane, B.-N. Vo, and M.D. Adams. Rao-Blackwellised PHD SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 5410–5416, May 2010.

J. Mullane, B.-N. Vo, M.D. Adams, and B.-T. Vo. A random-finite-set approach to bayesian slam. *IEEE Transactions on Robotics*, 27(2):268–282, April 2011.

A. Nüchter. *3D Robotic Mapping: The Simultaneous Localization and Mapping Problem with Six Degrees of Freedom*. Springer Publishing Company, Incorporated, 2009.

U. Orguner, C. Lundquist, and K. Granström. Extended Target Tracking with a Cardinalized Probability Hypothesis Density Filter. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 65–72, Chicago, IL, USA, July 2011.

R. Ostrovsky, Y. Rabani, L. J. Schulman, and C. Swamy. The Effectiveness of Lloyd-Type Methods for the k-Means Problem. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 165–174, Berkeley, CA, USA, October 2006.

N. Petrov, L. Mihaylova, A. Gning, and D. Angelova. A sequential Monte Carlo approach for extended object tracking in the presence of clutter. In *Proceedings of the Informatik 2011: 6th Workshop on Sensor Data Fusion: Trends, Solutions, Applications*, 2011.

F. T. Ramos, D. Fox, and H. F. Durrant-Whyte. CRF-matching: Conditional random fields for feature-based scan matching. In *Proceedings of Robotics: Science and Systems (RSS)*, Atlanta, USA, June 2007.

X. Rong Li and V.P. Jilkov. Survey of maneuvering target tracking: Part I. Dynamic models. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4): 1333–1364, October 2003.

D. J. Salmond and M. C. Parr. Track maintenance using measurements of target extent. *IEE Proceedings - Radar, Sonar and Navigation*, 150(6):389–395, December 2003.

T. Schön, F. Gustafsson, and P.-J. Nordlund. Marginalized particle filters for mixed linear/nonlinear state-space models. *IEEE Transactions on Signal Processing*, 53(7):2279–2289, July 2005.

D. Schuhmacher, B.-T. Vo, and B.-N. Vo. A consistent metric for performance evaluation of multi-object filters. *IEEE Transactions on Signal Processing*, 56 (8):3447–3457, August 2008.

D. Simon. *Optimal State Estimation: Kalman, H-infinity and Nonlinear Approaches*. John Wiley & Sons, 2006.

R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In *Autonomous Robot Vehicles*, pages 167–193. Springer, 1990.

A. Swain and D. Clark. Extended object filtering using spatial independent cluster processes. In *Proceedings of the International Conference on Information Fusion (FUSION)*, Edinburgh, UK, july 2010.

S. Thrun and J. J. Leonard. Simultaneous Localization and Mapping. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, pages 871–889. Springer, 2008.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

B.-N. Vo and W.-K. Ma. The Gaussian mixture probability hypothesis density filter. *IEEE Transactions on Signal Processing*, 54(11):4091–4104, November 2006.

B.-N. Vo, S. Singh, and A. Doucet. Sequential monte carlo methods for multi-target filtering with random finite sets. *IEEE Transactions on Aerospace and Electronic Systems*, 41(4):1224–1245, October 2005.

B.-T. Vo, B.-N. Vo, and A. Cantoni. Analytic implementations of the cardinalized probability hypothesis density filter. *IEEE Transactions on Signal Processing*, 55(7):3553–3567, July 2007.

W. Wieneke and J. W. Koch. Probabilistic tracking of multiple extended targets using random matrices. In *Proceedings of SPIE Signal and Data Processing of Small Targets*, Orlando, FL, USA, April 2010.

Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, 1994.

H. Zhu, C. Han, and C. Li. An extended target tracking method with random finite set observations. In *Proceedings of the International Conference on Information Fusion (FUSION)*, pages 73–78, Chicago, IL, USA, July 2011.