Linköping studies in science and technology. Dissertations.
No. 1723

Modeling of Magnetic Fields and Extended Objects for Localization Applications

Niklas Wahlström

LINKÖPING UNIVERSITY

Cover illustration: The cover shows a magnetic dipole field (3.6), which is used as the sensor model in Paper A. The dipole is placed at the height of the front cover with position $\mathbf{r}_0 = [104\,\text{mm},\ 90\,\text{mm},\ 0\,\text{mm}]$ relative to the bottom left corner of the front cover, with the magnetic dipole moment $\mathbf{m} = [0.85,\ 0.53,\ 0]$ being orthogonal to its surface. A 3D cutout of the scalar potential for the magnetic dipole $\varphi(\mathbf{r}) = \frac{(\mathbf{r}\cdot\mathbf{m})}{\|\mathbf{r}\|^3}$ is displayed on front, back and side cover, where $\mathbf{r}$ is the displacement relative to the position of the magnetic dipole. On the front cover, the field lines of the magnetic dipole field are overlaid.

Linköping studies in science and technology. Dissertations.
No. 1723

Modeling of Magnetic Fields and Extended Objects for Localization Applications

Niklas Wahlström

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

ISBN 978-91-7685-903-2 ISSN 0345-7524

Printed by LiU-Tryck, Linköping, Sweden 2015

Till min familj

Abstract

The level of automation in our society is ever increasing. Technologies like self-driving cars, virtual reality, and fully autonomous robots, which all were unimaginable a few decades ago, are realizable today, and will become standard consumer products in the future. These technologies depend upon autonomous localization and situation awareness where careful processing of sensory data is required. To increase efficiency, robustness and reliability, appropriate models for these data are needed. In this thesis, such models are analyzed within three different application areas, namely (1) magnetic localization, (2) extended target tracking, and (3) autonomous learning from raw pixel information.

Magnetic localization is based on one or more magnetometers measuring the induced magnetic field from magnetic objects. In this thesis we present a model for determining the position and the orientation of small magnets with an accuracy of a few millimeters. This enables three-dimensional interaction with computer programs that cannot be handled with other localization techniques. Further, an additional model is proposed for detecting wrong-way drivers on highways based on sensor data from magnetometers deployed in the vicinity of traffic lanes. Models for mapping complex magnetic environments are also analyzed. Such magnetic maps can be used for indoor localization where other systems, such as GPS, do not work.

In the second application area, models for tracking objects from laser range sensor data are analyzed. The target shape is modeled with a Gaussian process and is estimated jointly with target position and orientation. The resulting algorithm is capable of tracking various objects with different shapes within the same surveillance region.

In the third application area, autonomous learning based on high-dimensional sensor data is considered. In this thesis, we consider one instance of this challenge, the so-called pixels to torques problem, where an agent must learn a closed-loop control policy from pixel information only. To solve this problem, high-dimensional time series are described using a low-dimensional dynamical model. Techniques from machine learning together with standard tools from control theory are used to autonomously design a controller for the system without any prior knowledge.

System models used in the applications above are often provided in continuous time. However, a major part of the applied theory is developed for discrete-time systems. Discretization of continuous-time models is hence fundamental. Therefore, this thesis ends with a method for performing such discretization using Lyapunov equations together with analytical solutions, enabling efficient implementation in software.

Populärvetenskaplig sammanfattning

Hur kan man få en dator att följa pucken i bordshockey för att sammanställa match-statistik, en pensel att måla virtuella vattenfärger, en skalpell för att digitalisera patologi, eller ett multi-verktyg för att skulptera i 3D? Detta är fyra applikationer som bygger på den patentsökta algoritm som utvecklats i avhandlingen. Metoden bygger på att man gömmer en liten magnet i verktyget, och placerar ut ett antal tre-axliga magnetometrar - av samma slag som vi har i våra smarta telefoner - i ett nätverk kring vår arbetsyta. Magnetens magnetfält ger upphov till en unik signatur i sensorerna som gör att man kan beräkna magnetens position i tre frihetsgrader, samt två av dess vinklar. Avhandlingen tar fram ett komplett ramverk för dessa beräkningar och tillhörande analys.

En annan tillämpning som studerats baserat på denna princip är detektion och klassificering av fordon. I ett samarbete med Luleå tekniska högskola med projektpartners har en algoritm tagits fram för att klassificera i vilken riktning fordonen passerar enbart med hjälp av mätningar från en två-axlig magnetometer. Tester utanför Luleå visar på i princip 100% korrekt klassificering.

Att se ett fordon som en struktur av magnetiska dipoler i stället för en enda stor, är ett exempel på ett så kallat utsträckt mål. I klassisk teori för att följa flygplan, båtar mm, beskrivs målen som en punkt, men många av dagens allt noggrannare sensorer genererar flera mätningar från samma mål. Genom att ge målen en geometrisk utsträckning eller andra attribut (som dipols-strukturer) kan man inte enbart förbättra målföljnings-algoritmerna och använda sensordata effektivare, utan också klassificera målen effektivare. I avhandlingen föreslås en modell som beskriver den geometriska formen på ett mer flexibelt sätt och med en högre detaljnivå än tidigare modeller i litteraturen.

En helt annan tillämpning som studerats är att använda maskininlärning för att lära en dator att styra en plan pendel till önskad position enbart genom att analysera pixlarna i video-bilder. Metodiken går ut på att låta datorn få studera mängder av bilder på en pendel, i det här fallet 1000-tals, för att förstå dynamiken av hur en känd styrsignal påverkar pendeln, för att sedan kunna agera autonomt när inlärningsfasen är klar. Tekniken skulle i förlängningen kunna användas för att utveckla autonoma robotar.

Acknowledgments

Doing a PhD is like hiking in the Swedish mountains. The driving force is the eager to explore and discover what is behind the next crest. Each new panorama broadens the perspective and one starts to realize how different paths are connected. However, the hike is not always smooth sailing. Effort needs to be invested and the conditions can sometimes be harsh. But when you reach the peak of a mountain, you are rewarded! While hiking, you can choose to follow well marked paths. But when you deviate from the path, you need a map and the ability to navigate for not getting lost. I have been lucky to have two skillful supervisors who trained me how to navigate within the landscape of research.

First of all, I would like to thank my supervisor Prof. Fredrik Gustafsson for your guidance and encouragement. Your efficiency and source of ideas are really amazing. Not only is your scientific record impressive. Also your entrepreneurial mindset inspires! Thanks for all late emails, comments on manuscripts with short notice, and new perspectives on my research and beyond.

My co-supervisor Prof. Thomas Schön has been great support and source of inspiration for my work, especially during the second half of my PhD. Your feedback on my work has been really encouraging. Thank you for all the effort that you have invested! I also appreciate your genuine interest in teaching and research, which I know is an inspiration not only for me.

While hiking, you often have the pleasure to meet new people who can help you along the way. During (and after) my pre-doc at Imperial College, I had the pleasure to work with Dr. Marc Deisenroth. Your passion and dedication to the topic is amazing. I also like your German style to be being efficient, direct, and in the same time very encouraging. Thank you Marc! I also thank John Assael who recently joined the project with fresh energy and an incredible working spirit!

I am in deep gratitude to Dr. Gustaf Hendeby who has been a great support. I admire both your scientific and technical skills. Thank you for always helping out with various computer related issues. I also would like to thank Dr. Emre Özkan for rewarding supervision and teamwork. I appreciate your Turkish way of approaching things without losing the precision. I also would like to thank Dr. Roland Hostettler for the nice collaboration we had, mainly during the first half of my PhD. To share and develop thoughts with others is rewarding.

My hike started already in 2010 when I moved into the RT-corridor while writing my Master's thesis. This gave me the inspiration to continue. Therefore, I am very grateful that Prof. Fredrik Gustafsson invited me to be part of the Automatic Control group. Since then, the group has been skillfully headed by Prof. Svante Gunnarsson, the division coordinator Ninna Stensgård and her predecessor Åsa Karmelind. I also want to acknowledge the Swedish Foundation for Strategic Research (SSF) for the financial support under the project Cooperative Localization in the program on Software Intensive Systems.

One of the best things with the RT group is all my amazing colleagues. Without the enjoyable working atmosphere that you create, this experience would not have been close as good. I would like to thank Lic. Manon Kok for all joint efforts with measurement collection and paper writing. I thank you Lic. Michael

Roth and Lic. Tohid Adeshiri for your hospitality and friendship. Many thanks go to Dr. Patrik Axelsson and Lic. Jonas Linder for all nice badminton matches. Dr. Sina Khoshfetrat Pakazad does a tremendous job organizing cheerful evenings and Dr. André Carvalho Bittencourt and Dr. Zoran Sjanic make sure that the spirit always is high on our parties. I want to thank my roommates Gustav Lindmark, Lic. Marek Syldatk and Dr. Mehmet Burak Guldogan for pleasant company during these years. I enjoy the company of Lic. Ylva Jung, Hanna Nykvist, Clas Veibäck, Dr. Martin Skoglund and Martin Lindfors with whom I often have rewarding discussions during fika breaks.

The hike can also bring you to other locations around the world. I would therefore like to thank Christian Andersson Naesseth, Lic. Johan Dahlin, Dr. Fredrik Lindsten, Lic. Isak Nilsen and Lic. Daniel Simon for nice company during various conference travels around the world, and Hanna, Patrik, Jonas, Andre, Tohid, Isak, Emre and Ylva for terrific ski-trips to the Alps and Swedish mountains.

Special thanks also go to Prof. Fredrik Gustafsson, Prof. Thomas Schön, Dr. Emre Özkan, Dr. Gustaf Hendeby, and Dr. Bram Dil who have been proofreading various parts of this thesis. I also thank Gustaf and Dr. Henrik Tidefelt for their contributions to the LaTeX-template, which made the thesis writing much easier. I also acknowledge Dr. Zoran Sjanic for providing additional perspectives on the "back" cover of this thesis.

The purpose of a hike is not always to take the fastest path from A to B. Detours along the way can be as rewarding. During my hike, I have made many detours, among which I especially want to mention the Wand project. I would like to thank all people that in one or another why have contributed to the project for realizing all cool applications with magnetic tracking throughout the past four years. First and foremost, I thank Prof. Fredrik Gustafsson for all collaborations on this project from day one. Without your input I would probably have left this with just a conference paper. Tomas Ahlström was involved early in the project having a mind for commercialization already at an early stage. I would like to thank Jesper Svanfeldt who made the design of the hardware (which we still are using!), and Lucas Correia and David Jonsson who made the first successful demo (Figure 1.1a), all three of them as a part of their Master's thesis projects during spring 2011. I also thank Dr. Stefan Gustavson who invested a lot of work for that demo to come true. Lucas and David also made the first version of the API based on my MATLAB code, for which I am very thankful.

Dr. Stefan Lindholm made valuable contributions to the project, everything from pitching for presumptive partners, to brainstorming ideas for commercialization. Without your input, we would not have reached as far. Thomas Wilkinson did a nice job creating a ping-pong game using the sensors, which gave us valuable feedback on what applications to aim for. During summer 2013, Mattias Lundstedt and Simon Borgenvall initiated the table hockey demo, which was finalized by Dr. Gustaf Hendeby, Fredrik and me, and presented at Venture Arena 2013 (Figure 1.1d). I also thank "M-verkstan" for changing all steel bars to brass bars in that table hockey game.

Gustaf has made huge contributions on shaping up the API. I highly admire your coding skills. Jonas Nilsson has since 2013 also been a valuable partner in

the project adding competence that Gustaf, Fredrik, and I are lacking. During spring 2014, Martin Törnros and Thomas Rydell at Interactive Institute made a workstation for pathologists using our sensors and API (Figure 1.1c). That put our tracking solution to a new stage! In 2015, Nils Hallqvist and Anton Hemling started the process of building a new sensor solution. Starting this summer, Isabelle Forsman and Olle Grahn made the most impressive demo so far (Figure 1.1b). I thank all of you for the work that you have invested. Finally, I would like to thank everybody who financially supported this project, in particular Innovationsbron, Almi, SSF and Innovationskontoret.

In the mountains, the weather can shift rapidly and the motivation can be affected. To endure longer hikes, you need mental support. I would like to show my deepest gratitude to my family, my parents Karin and Tord, my siblings Helen and Johan with families. I apologize for all birthdays that I have forgotten. You should know that I feel lucky to have you.

Nicky! Thank you for all your love and patience! I almost cannot understand how you have endured. You are a terrific hiking partner and I hope there are more to explore!

Linköping, October 2015
Niklas Wahlström

Contents

Notation

Throughout this thesis, scalars or scalar-valued functions are denoted with non-bold lower-case symbols, e.g. θ. Vectors or vector-valued functions are denoted with bold lower-case symbols, e.g. $\mathbf{y}$. Electromagnetic vector fields are denoted with bold upper-case symbols, e.g. $\mathbf{B}$. This choice has been made to be consistent with most literature on electromagnetism although it mathematically can be considered as a vector-valued function. Finally, matrices are denoted with upper-case non-bold symbols, e.g. P. Furthermore, Cartesian coordinates are denoted using Sans-serif font, e.g., x and y, to distinguish them from other variables.

Electromagnetic Theory

Notation	Meaning
$\mathbf{E}$	Electric field, $[\mathrm{V\,m^{-1}} = \mathrm{kg\,m\,s^{-3}\,A^{-1}}]$
$\mathbf{D}$	Electric displacement field, $[\mathrm{C\,m^{-2}} = \mathrm{A\,s\,m^{-2}}]$
$\mathbf{B}$	Magnetic field, $[\mathrm{T} = \mathrm{kg\,A^{-1}\,s^{-2}}]$
$\mathbf{H}$	Magnetizing field, $[\mathrm{A\,m^{-1}}]$
μ_0	Permeability of free space, $[\mathrm{H\,m^{-1}} = \mathrm{kg\,m\,A^{-2}\,s^{-2}}]$
ε_0	Permittivity of free space, $[\mathrm{F\,m^{-1}} = \mathrm{s^4\,A^2\,kg^{-1}\,m^{-3}}]$
ρ	Charge density, $[\mathrm{C\,m^{-3}} = \mathrm{A\,s\,m^{-3}}]$
$\mathbf{J}$	Current density, $[\mathrm{A\,m^{-2}}]$
$\mathbf{J}_m$	Magnetization current density, $[\mathrm{A\,m^{-2}}]$
$\mathbf{J}_f$	Free current density, $[\mathrm{A\,m^{-2}}]$
$\mathbf{M}$	Magnetization, $[\mathrm{A\,m^{-1}}]$
$\mathbf{A}$	Magnetic vector potential, $[\mathrm{V\,s\,m^{-1}} = \mathrm{kg\,m\,s^{-2}\,A^{-1}}]$
φ	Magnetic scalar potential, [A]
$\rho_{\mathbf{M}}$	Effective magnetic-charge density, $[\mathrm{A\,m^{-2}}]$
$\mathbf{m}$	Magnetic dipole moment, $[\mathrm{A\,m^2}]$
$\nabla \cdot \mathbf{B}$	Divergence of vector field $\mathbf{B}$
$\nabla \times \mathbf{B}$	Curl of vector field $\mathbf{B}$
$\nabla\varphi$	Gradient of scalar field φ

Symbols and Operators

Notation	Meaning
A^T	Transpose of matrix A
trA	Trace of matrix A
$\mathbb{E}$	Expected value
Var	Variance
Cov	Covariance
$\frac{\partial y}{\partial x}$	Partial derivative of y with respect to x
$\times$	Cross product
$\odot$	Quaternion product
$\otimes$	Kronecker product
$\triangleq$	Defined as
$\sim$	is distributed according to
$\in$	belongs to
$\mathcal{O}$	Ordo
I_n	Identity matrix of size $n \times n$
0_n	Matrix with only zeros of size $n \times n$
$0_{m \times n}$	Matrix with only zeros of size $m \times n$
$0_{m \times n}$	Matrix with only zeros of size $m \times n$

Estimation

Notation	Meaning
$\mathbf{x}$	State
$\mathbf{y}$	Measurement
$\mathbf{u}$	Input/control input
$\mathbf{z}$	Feature
$\mathbf{w}$	Process noise
$\mathbf{e}$	Measurement noise
T	Sampling time
$\mathcal{N}(\cdot, \cdot)$	Gaussian distribution with mean and covariance
$\mathcal{GP}(\cdot, \cdot)$	Gaussian process with mean and covariance function
P	State covariance matrix
Q	Process noise covariance matrix
R	Measurement noise covariance matrix

Geometry and dynamics

Notation	Meaning
$\mathbf{r}$	Position
$\mathbf{v}$	Velocity
$\mathbf{q}$	Unit quaternion
R	Rotation matrix
ω	Angular velocity
x	Cartesian x-coordinate
y	Cartesian y-coordinate
z	Cartesian z-coordinate

Abbreviations

Abbreviation	Meaning
GNSS	Global navigation satellite system
GPS	Global positioning system
WLAN	Wireless local area networks
IMU	Inertial measurement unit
WSN	Wireless sensor network
SLAM	Simultaneous localization and mapping
SNR	Signal to noise ratio
GLRT	Generalized likelihood ratio test
PDF	Probability density function
CRLB	Cramér-Rao lower bound
FIM	Fischer information matrix
BFGS	Broyden-Fletcher-Goldfarb-Shanno
GP	Gaussian process
SE	Squared exponential
DDM	Deep dynamical model
NARX	Nonlinear auto-regressive exogenous model
DoF	Degrees of freedom
NLL	Negative log likelihood
RMSE	Root-mean-square error
EKF	Extended Kalman filter
IOU	Intersection-Over-Union
MPC	Model predictive control
PCA	Principal component analysis
PILCO	Probabilistic inference for learning control
RL	Reinforcement learning
SISO	Single input, single output

Part I

Background

1

Introduction

Many modern technologies are characterized by a high degree of autonomy. To accomplish this autonomy, appropriate sensors technologies are needed. There exist many well-established sensor technologies such as GPS, radar and vision, all which are well suited for certain applications. However, all technologies have their advantages and disadvantages, which can be quantified, for example, in terms of cost, accuracy, range, reliability, flexibility, weight, and size. This thesis considers the problem of localization, control, and self-awareness using different types of such sensor technologies.

An important ingredient in these sensor technologies is the ability to describe the relation between measurements from a sensor and some quantity that we can interpret, for example a position of an object. This is accomplished by using models. Therefore, this thesis has a specific focus on how to model the data from these sensors.

This introductory chapter provides overview of the contributions in this thesis from an application point of view. The presentation is organized based on four different sensor techniques, all of which the author of this thesis has worked with.

1.1 Magnetometers

Magnetometers are sensors that measure strength and mostly also the direction of magnetic fields. They have various applications ranging from finding sea mines (Clem, 2002) to monitoring "space weather" (Singer et al., 1996). In navigation, magnetic sensors are most commonly used as a compass that measures the bearing of an object. However, in this thesis we use them to sense other magnetic objects. This approach is used in magnetic anomaly detectors for detecting ferromagnetic objects, see Lenz and Edelstein (2006) for an overview of the problem and other applications.

In this thesis, we are not only interested in detecting magnetic objects, but

also in determining their position, direction of motion, magnetic signature and geometric shape. This is accomplished by using mathematical models relating these quantities to the measured magnetic field. In contrast to GPS, laser range sensor and computer vision, this localization technology is not dependent of unobstructed line-of-sight between the sensors and the object. In fact, magnetic fields propagate in all direction before reaching the sensor. Therefore, it is nearly impossible to eliminate the magnetic signature of magnetic objects. This makes the sensors insensitive to jamming, which is important in many applications. In addition, magnetic sensing is almost independent of weather conditions.

Recently, magnetometers have become smaller and cheaper, which makes an extensive usage of magnetometers more interesting, for example, in localization of magnetic objects. However, for many applications, the short sensor range is limiting. With commercial-grade magnetometers, a 1 cm long neodymium magnet can be sensed from a distance of approximately 1 m, and a car can be sensed from a distance of approximately 10 m. Magnetic sensors are *superpositional* sensors, meaning that they measure the sum of the magnetic signatures from all present magnetic objects. In contrast to many other types of sensors (for example radar and vision sensors), more objects do not create more measurements (or detections), which makes a multi-target tracking framework more challenging than for non-superpositional sensors.

As part of the PhD program for the author of this thesis, many applications have been analyzed and realized using magnetometers. Some of them are also considered in more depth in this thesis. These applications can be divided into three categories, (1) devices for human-computer interaction, (2) traffic surveillance and (3) indoor localization and mapping. These application areas are introduced below.

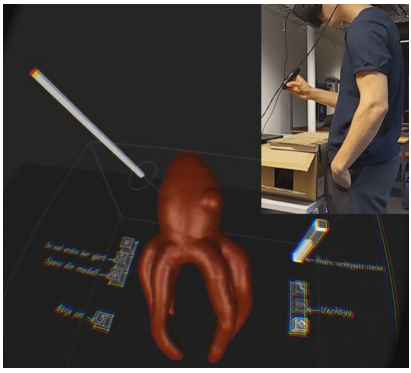
1.1.1 Devices for Human-Computer Interaction

A stationary sensor network of multiple magnetometers can be used to localize and track small magnets. Both position and orientation information for these magnets can be extracted. By mounting a magnet in a hand-held device, a wireless and cheap tool can be constructed. This tool can be used as a three-dimensional input device for computer programs, which increases the level of interaction in comparison to other input devices, e.g., touch screens, 2D mouse devices, and standard keyboards. In contrast to vision based solutions (for example Microsoft Kinect), the user does not have to operate relative to any special camera position. Further, that hand-held device does not need any batteries or external power supply.

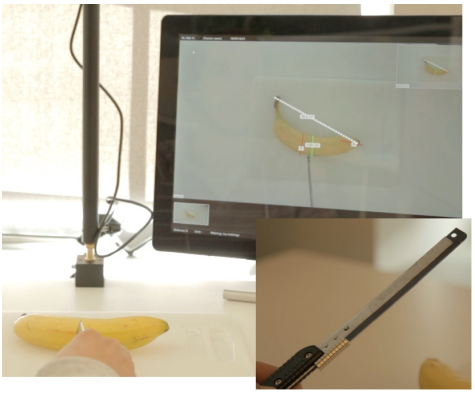
This magnetic localization technique is described in Paper A and has been used in multiple applications. Below four applications are described that all have been realized with this technique, see also Figure 1.1.

(a) **Digital watercolors:** Museums and science centers have a high need for technology enabling interactive exhibits that encourage visitors to experiment and explore. In exhibits where spatial information is important, a localiza-

*(a) **Digital watercolors:** Virtual watercolor painting application, where the painting is displayed on a screen. The user interacts using a regular painting brush equipped with a permanent magnet. A sensor network of four magnetometers is mounted under the screen that senses the position and orientation of the brush. The user can paint and splash color on the virtual canvas and select new colors from the palette and wet the brush in the (virtual) water glass.*



*(b) **Interactive modeling:** A computer program for interactive 3D modeling in a virtual reality. The hand-held device equipped with a magnet can be used to pull, push and smoothen textures of an object, as well as moving and turning it. Both the virtual object and the virtual device can be observed through a head-mounted display making the interaction intuitive and realistic.*



*(c) **Digital pathology:** Input device within digital pathology. A magnet is placed in a scalpel to be used by a pathologist. With the tracking system, they can directly measure distances and create a digital log of their work. This saves time and removes manual non-ergonomic activities.*

*(d) **Digital table hockey:** The sensor network is placed under a table hockey game and a magnet is mounted in the puck. Accurate position can be used to visualize the puck on a digital screen. The system can also count the number of goals.*

***Figure 1.1:** Four applications that have been realized with the technique described in Paper A. Photo: (a) Anders Ynnerman (2015), (b) Olle Grahn, Isabelle Forsman (2015), (c) Linkin AB (2014), (d) Martin Stenmarck (2015).*

tion system is required. These systems need to be intuitive for the visitors to control and interact with.

In this context, the magnetic localization technique was used in an exhibition case mimicking water color painting, see Figure 1.1a. The software for this exhibition case was a result of a Master's thesis project by Correia and Jonsson (2012) that was supervised by the author of this thesis. The resulting system was running at the science center *Visualiseringscenter C* in Norrköping during 2012 and 2013.

(b) **Interactive modeling in a virtual reality:** The hand-held device is also suitable for interaction and manipulation of three-dimensional virtual objects. During 2015 the technique was used in a computer program enabling interactive 3D modeling. With this program a virtual object can be crafted using the hand-held device, see Figure 1.1b. Together with 3D printers, this could increase the ease of prototyping and crafting real objects. The integration of the magnetic tracking solution was performed by Isabelle Forsman and Olle Grahn as a continuation of their Bachelor's project report (Forsman et al., 2015).

(c) **Digital pathology:** Within medicine and healthcare, technology has made a giant leap during the last decades. However, more can be accomplished to increase efficiency and quality even further. For example, in pathology the introduction of digital technologies could generate huge cost savings (Ho et al., 2014). This has motivated the VINNOVA financed project "Optimized flows and IT tools for digital pathology". As a part of that project, the magnetic localization technique has been used for improving the workstation that pathologists use when examining tissues. By mounting a magnet in a scalpel, a digital record can be constructed of the actions that have been performed, see Figure 1.1c.

(d) **Digital table hockey:** Automation has also increased in toys, games and other leisure activities. A common trend is to enhance classical analog toys and games with digital features. With magnetic localization, a similar enhancement can be made for a table hockey game. By mounting a magnet in a puck for a table hockey game, the puck can be localized in real time, and meta information can be extracted, e.g., number of goals, see Figure 1.1d.

1.1.2 Traffic Surveillance

Localization and tracking of vehicles is a primary concern in automated traffic surveillance systems. The information can be used for statistical purposes by road administrations, urban planners or traffic management centers to improve the road infrastructure. The information can also be used in safety systems, for example to detect wrong-way drivers on highways.

Vehicles have a high content of ferromagnetic material and they will therefore induce a magnetic field, which can be measured by magnetometers. By deploying

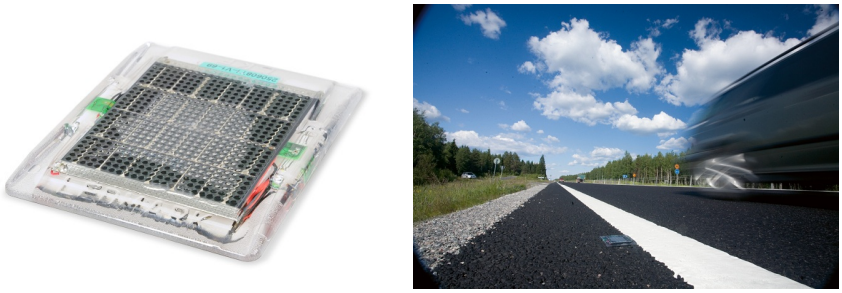


Figure 1.2: *Sensor unit including a 2-axis magnetometer and an accelerometer powered with solar energy. The unit is glued onto the road surface to sustain harsh weather conditions. Measurements from this unit are used in Paper B. By courtesy of GEVEKO ITS. (`www.gevekoits.dk`)*

one or more of these sensors in the vicinity of the traffic lane, the vehicle can be localized. For this application, the magnetometers have the advantage of being less sensitive to weather conditions in comparison to other technologies in automated surveillance systems, e.g., cameras. Their energy efficiency makes it possible to integrate them in a wireless sensor node powered by solar energy. These nodes can easily be deployed at points of interest, which makes the technology flexible.

In Wahlström (2010); Wahlström et al. (2011); Wahlström and Gustafsson (2014), different models are investigated for localizing vehicles based on a sensor network of magnetometers. Both models for point targets and extended targets were proposed.

Parts of this work have also been accomplished in collaboration with Luleå University of Technology working with a sensor unit equipped with a magnetometer suited for standing the harsh weather conditions present in northern Sweden, see Figure 1.2. Within this cooperation a robust classifier for determining the driving direction of a vehicle has been implemented and analyzed. This work is presented in Paper B. This sensor unit also contains an accelerometer enabling detection and estimation using road surface vibration, which has been investigated by Hostettler et al. (2012).

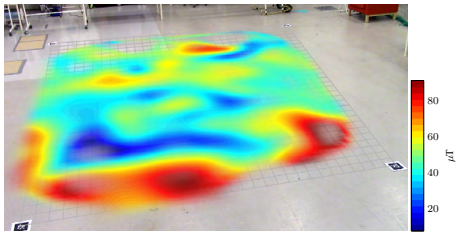
1.1.3 Indoor Localization and Mapping

Over the past decade, we have witnessed an increasing attention for localization in indoor environments. There are many applications, e.g., operation of emergency personnel, navigation in shopping malls, and positioning of autonomous vacuum cleaners. Because the GPS system does not work in indoor environments, many alternative localization techniques have been discussed and analyzed. Deak et al. (2012) give a survey of different indoor localization systems.

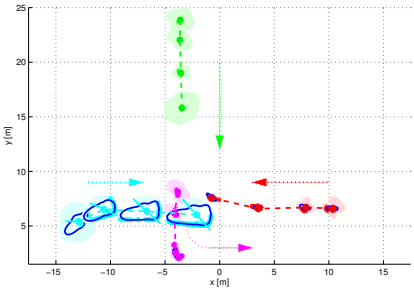
In recent years, the use of the magnetic disturbances present in indoor environments has been considered as a source for localization, (see, e.g., Vissière et al., 2007; Vallivaara et al., 2011; Zhang and Martin, 2011; Le Grand and Thrun,

2012). These disturbances are induced by metallic structures present in most buildings and carry enough information to be used for localization. The disturbances can be measured with a magnetometer and the localization can be aided using other sensors, e.g., accelerometers and gyroscopes.

The modeling of these magnetic environments is challenging. Unlike the two previous application areas, the magnetic content is not limited to be contained in a small region, i.e., within a vehicle or within a permanent magnet. In Paper C, the modeling of these complex magnetic environments is addressed. Based on that work, the setting has been extended in Solin et al. (2015) to handle more complex scenarios , e.g., larger buildings and environments changing over time, enabled by a more computational efficient algorithm. Figure 1.3a illustrates an estimated magnetic map constructed based on that work.



(a) *A map of the magnitude of the magnetic field in an indoor environment. The magnetic field has been measured by a robot equipped with a magnetometer. The position was determined by an optical reference system. This figure is from our work in Solin et al. (2015), which is a continuation of Paper C.*



(b) *Estimated position, orientation and shape of four different targets, here encoded with four different colors, are illustrated at four different time instances. The figure is from Paper D.*

Figure 1.3: *Two different applications considered in this thesis using Gaussian processes. Gaussian processes are explained in Section 2.3.*

1.2 Laser Range Sensors

A *laser range sensor* measures the distance from the sensor location to the nearest object using a laser beam. By sweeping over different angles, it provides a map of contours for the surrounding environment. If a certain object enters the scene, that object will be visible to the sensor in case it is within the range of the sensor, is not obstructed by other objects, and has a favorable reflectance property. Due to these properties, laser range sensors are among the most popular sensors in robotics (Thrun et al., 2005).

In many aspects, this type of sensor is different from magnetometers. In contrast to magnetometers, it does require line-of-sight to the target to be able to detect it, which is not required by magnetometers. This sensor also can be considered to be non-superpositional. If multiple objects enter the scene, more measurements will be generated. Also, if the object has a large extent, more measurements will be generated along the contour of that object, than if the target would have been smaller.

This last property is exploited in Paper D, in which a model is proposed for jointly estimating the position, orientation and extent of objects moving within line-of-sight of the sensor. This is accomplished by modeling the extent with *Gaussian processes*. Some of the results are illustrated in Figure 1.3b. This technique could for example be used in traffic surveillance applications monitoring cars, bicycles and pedestrians in a crossing or for autonomously localizing robots in unknown environments.

1.3 Image Sensors

An *image sensor* is a sensor whose measurement constitutes an image of some kind. In this thesis, we consider digital image sensors. They consist of an array of pixel sensors, each of them containing a photo detector. An image sensor can be considered as a high-dimensional (2D-array) sensor, in which each pixel corresponds to one dimension in the measurement vector.

Neighboring pixels are usually highly correlated with each other and in most cases only a small fraction of the measurements is related to the quantity that is of interest in the application. A common procedure is to reduce this high-dimensional measurement into a collection of lower-dimensional features. For tracking and localization purposes, this is usually performed with algorithms for extracting hand-crafted features that detect edges and corners similar to the measurements from the laser range sensor described in the previous section.

In Paper E and F, a fundamentally different path is followed. In these papers, a low-dimensional representation of the high-dimensional measurement is still extracted. However, this is not performed in a separate pre-processing step with hand-crafted features. Instead we employ data-driven dimensionality reduction methods, which do not explicitly take geometrical properties into account. Through this low-dimensional representation, predictions of future image frames can be generated. This allows a robot to plan and control for accomplishing a certain task without any prior knowledge of neither the environment it is operating in, nor its own dynamics, see Figure 1.4 for an illustration of this concept. In Figure 1.5, prediction results for a double planar pendulum is shown using the model described in Paper E and F. The figure is taken from (Assael et al., 2015), which is an extension of the work in Paper F.

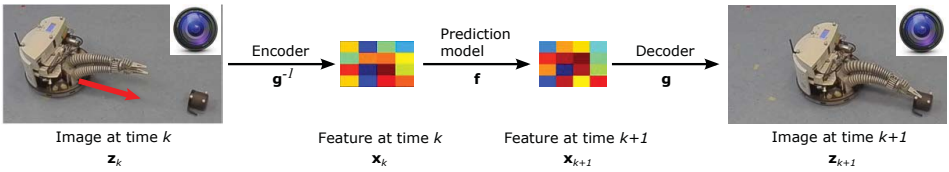


Figure 1.4: *A camera observes a robot approaching an object. A good low-dimensional feature representation of an image is important for learning a predictive model if the camera is the only sensor available. This is a sketch of the model architecture considered in Paper E and F.*

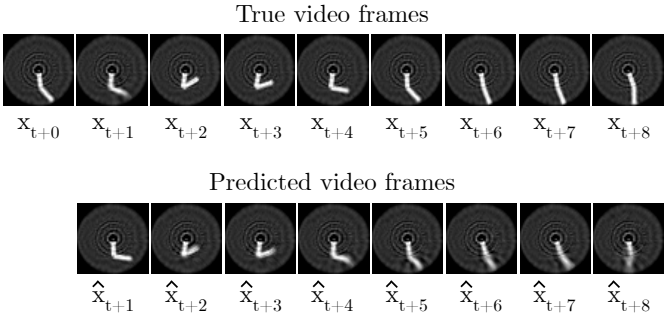
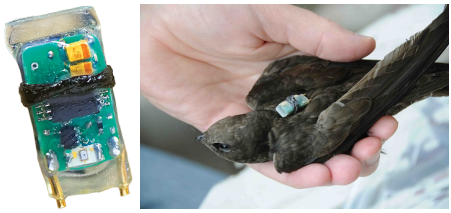


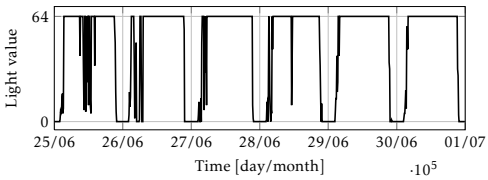
Figure 1.5: *True and predicted frames based on the model sketched in Figure 1.4. The figure is taken from Assael et al. (2015), which is a continuation of the work in Paper F.*

1.4 Light Sensors

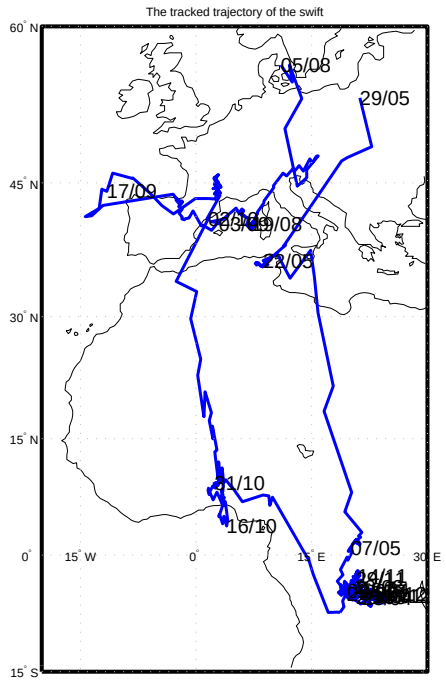
The position of celestial bodies, such as the sun, the moon, a planet or a star, has for hundreds of years been used by sailors in order to navigate. The angle between the celestial body and the horizon (altitude) reveals a combination of the longitude and latitude of the observer. Partial information about the altitude of the sun can be captured using a *light sensor* measuring the light intensity, see Figure 1.6b. From these data, the events of sunrise and sunset can be detected. The events occur when the sun geometrically is a bit below the horizon, which in turn depends on the threshold of light intensity for detecting these events. Localization using light sensors is for example discussed by Hill (1994); Stutchbury et al. (2009); Ekstrom (2004). The big advantage with light intensity sensors is their low weight and low energy consumption in comparison to the GPS. It allows to construct devices under one gram (including sensor, memory, battery and clock) lasting for many years, see Figure 1.6a. The sensor is also more cost efficient and smaller than the GPS. Like the GPS, the technology can be used over the whole earth, except close to the North Pole and South Pole during the winter solstice and summer solstice, where the sun never rises. This technology is an attractive solution for applications, in which weight, cost and global coverage are important,



(a) A light logger consisting of a battery, memory clock and a light sensor with a weight of less than one gram, suited for bird localization. Photo: Anders Hedenström, Forskning & Framsteg 5/6 - 2012.



(b) Light intensity sampled from a sensor mounted on a Common Murre (sv. Sillgrissla) from Karlsöarna in the Baltic Sea during the summer of 2010. The light sensor saturates during the day time and the night time. The measurements are also corrupted due to shading. The sampling time is 10 minutes.



(c) The trajectory of the common swift during a period of 298 days. The positions are estimated at each sunrise and sunset.

Figure 1.6: *Illustration of setup and results for the bird localization described in Wahlström et al. (2013).*

for example, to localize small migrating animals.

On the other hand, the accuracy of approximately 150 km is much lower than other technologies. It also depends on weather conditions and proximity to equinox and equator. The technology is also challenged by shading of foliage and other vegetation, which might cause false detection of the sunrise and sunset events.

Localization of migrating birds is important for evaluating theories about their the genetics, migration patterns, and the evolution behind. For smaller birds, the weight of the localization equipment attached to the bird is crucial. As a rule of thumb, the sensor can weigh at most 5 % of a bird's weight. Therefore, the use of light sensors is an attractive localization technique, providing absolute position of small birds that other techniques cannot accomplish with these weight requirements.

With this technology, the migration pattern of the *common swift* has been revealed by researchers from Lund University using data from light loggers mounted on different swifts (Åkesson et al., 2012). The common swift is a medium sized bird with a weight of 40 g in average, limiting the maximum al-

lowed weight of the sensor equipment to 2 g.

In Wahlström et al. (2013), the estimation of migration path was formulated as a nonlinear filtering problem, in which the position is updated at each sunrise and sunset. That study was performed in collaboration with aforementioned biologists from Lund University. They also provided the real data, which were used in the work.

This work is not included or further studied in this thesis. The interested reader can refer to the author's Licentiate's thesis (Wahlström, 2013), which also contains an introductory chapter on astronomy needed to derive the appropriate sensor models.

1.5 Contribution

This thesis contains the following contributions:

- **Parametric magnetic models:** Models describing moving magnetic objects. In Paper A, a variety of different models are presented including a point target model, an extended target model and motion models related to these. Paper B describes a model for estimating the driving direction of the vehicle.
- **Nonparametric magnetic models:** Models describing complex magnetic environments suitable for indoor localization are described in Paper C.
- **Flexible models for extended target tracking:** Models for describing the contour of targets suitable for tracking and localization are proposed in Paper D.
- **Autonomous learning from raw pixel information:** A model and control strategy for autonomously learning a task from raw pixel information without any additional prior information about the system at hand is presented in Paper E and F.
- **Efficient discretization of stochastic dynamical systems:** A method for performing discretization of stochastic dynamical systems using Lyapunov equations is proposed in Paper G.

1.6 Thesis Outline

The thesis is divided into two parts, with edited versions of published and submitted papers in Part II.

Part I - Background

Part I introduces the background theory needed for the models presented in this thesis. In addition to this introductory chapter, the relevant background is introduced in Chapter 2 and Chapter 3. Chapter 2 introduces three different model

components used in the publications. Chapter 3 introduces the relevant electromagnetic theory that is needed to describe the relation between magnetic objects and their induced magnetic field. Part I ends with Chapter 4 that summarizes the conclusions and presents possible directions for future work.

Part II - Publications

The second part consists of edited versions of seven publications. Below is a summary of each paper together with a clarification of the background and the contribution of the author for each of the papers.

Paper A: Tracking Position and Orientation of Magnetic Objects Using Magnetometer Networks

> N. Wahlström and F. Gustafsson. Tracking position and orientation of magnetic objects using magnetometer networks. *IEEE Transactions on Signal Processing*, 2015. Submitted.

Summary: This paper presents a localization technique, where a sensor network of magnetometers is used to track both the position and the orientation of a permanent magnet. The system can track all three degrees-of-freedom (DoF) for the position and two DoF for the orientation. The model is further extended to objects including multiple permanent magnets inducing an asymmetric magnetic field enabling tracking of all three DoF for orientation. Both motion models and sensor models are presented in the paper. The models are validated on real data achieving 5 mm error for position and 2° error for orientation. The paper ends with four applications: (1) virtual water colors, (2) interactive 3D modeling, (3) digital pathology, and (4) digital table hockey game, which all were realized as part of the research.

Background and Contribution: This paper is to a great extent a journal version of the patent application (Gustafsson and Wahlström, 2012), which was invented by me in collaboration with Prof. Fredrik Gustafsson in 2011. Since then, many application oriented project were conducted involving M.Sc. thesis students, science museums and other companies. For this paper, I wrote all code and also the vast majority of the text in the paper, which later was revised by Fredrik.

Paper B: Classification of Driving Direction in Traffic Surveillance Using Magnetometers

> N. Wahlström, R. Hostettler, F. Gustafsson, and W. Birk. Classification of driving direction in traffic surveillance using magnetometers. *IEEE Transactions on Intelligent Transportation Systems*, 15(4):1405–1418, 2014.

Summary: This paper presents a robust method for determining the driving direction of vehicles based on measurements from one 2-axis magnetometer is

presented. In contrast to the setting in Paper A, these targets are close to the sensor (relative to the size of the target) and cannot always be approximated with one or multiple permanent magnets. In addition, the algorithm is supposed to be implemented on wireless sensor nodes powered by solar cells with low energy budget. Consequently, the algorithm needs to be computationally cheap. The proposed solution relies on a non-linear transformation of the measurement data comprising two inner products. The validity of this transform is derived from the point target model (dipole model used in Paper A). Experimental verification indicates that good performance is achieved, even when targets are close to the sensor.

Background and Contribution: The cooperation with Dr. Roland Hostettler was initiated at Reglermöte (Swedish control conference) 2010 and during the fall we collected data together. Later, the author of this thesis came up with the core idea used in this paper. An early version of this work was then published in Wahlström et al. (2012b). The work was accomplished jointly by Roland and me including data collection, theoretical analysis, coding and writing. Prof. Fredrik Gustafsson and Prof. Wolfgang Birk acted as supervisors and reviewed the manuscript.

Paper C: Modeling Magnetic Fields Using Gaussian Processes

> N. Wahlström, M. Kok, T. B. Schön, and F. Gustafsson. Modeling magnetic fields using Gaussian processes. In *Proceedings of the the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 3522–3526, Vancouver, Canada, May 2013.

Summary: This is the third and last paper in this thesis dealing with magnetic sensors. In this paper a different approach for modeling of magnetic objects is taken in comparison to the previous two papers. In contrast to Paper A, the magnetic field is not induced by moving magnets, but rather by extended metallic structures present in indoor environments. Due to the complexity of such environments, we have used a non-parametric model, more precisely a Gaussian process that exploits constraints imposed by physics. The model and the associated estimator are validated on both simulated and real experimental data producing Bayesian non-parametric maps of both the magnetic field and the magnetized objects. This is related to the work Kok et al. (2013), in which the positioning based on magnetic maps was considered.

Background and Contribution: I came up with the modeling idea presented in this paper after a course in Machine Learning 2011, where Gaussian processes were taught. After that, I did the implementation and wrote the vast majority of the text. The measurements used in the paper were collected together with Lic. Manon Kok. After this work, Gaussian processes for magnetic maps were further investigated by primarily Arno Solin and Manon Kok, to which I also contributed. That work is reported in Solin et al. (2015).

Paper D: Extended Target Tracking Using Gaussian Processes

N. Wahlström and E. Özkan. Extended target tracking using Gaussian processes. *IEEE Transactions on Signal Processing*, 63(16):4165–4178, 2015.

Summary: In this paper, we suggest using Gaussian processes for tracking extended targets. Instead of magnetic fields, the Gaussian processes are used to model the contour of an extended object or group of objects. The shape and the kinematics of the object are simultaneously estimated, and the shape is learned online. The proposed algorithm is capable of tracking different objects with different shapes within the same surveillance region. The shape of the object is expressed analytically, with well-defined confidence intervals, which can be used for gating and association. Furthermore, we use an efficient recursive implementation of the algorithm by deriving a state space model in which the Gaussian process regression problem is cast into a state estimation problem.

Background and Contribution: Together with Dr. Emre Özkan the idea of using Gaussian Processes in this context was initiated. I did the major work in implementing this idea and in writing the technical part of the text. The remaining part of the text has been written jointly together. The experimental data used in this paper was collected by Dr. Karl Granström and has previously been used in Granström and Orguner (2012); Granström et al. (2012).

Paper E: Learning Deep Dynamical Models from Image Pixels

N. Wahlström, T. B. Schön, and M. P. Deisenroth. Learning deep dynamical models from image pixels. In *Proceedings of the 17th IFAC Symposium on System Identification (SYSID)*, Bejing, China, October 2015a.

Summary: Modeling dynamical systems is important in many disciplines, such as control, robotics, or neurotechnology. Commonly, the state of these systems is not directly observed, but only available through noisy and potentially high-dimensional observations. In these cases, system identification, i.e., finding the measurement mapping and the transition mapping (system dynamics) in latent space can be challenging. For linear system dynamics and measurement mappings efficient solutions for system identification are available. However, in practical applications, the linearity assumption does not hold, requiring nonlinear system identification techniques. If additionally the observations are high-dimensional (e.g., images), nonlinear system identification is inherently hard. To address the problem of nonlinear system identification from high-dimensional observations, we propose a Deep Dynamical Model (DDM) combining recent advances in deep learning and system identification. This model uses deep auto-encoders to learn a low-dimensional embedding of images jointly with a predictive model in this low-dimensional feature space. Joint learning ensures that not only static, but also dynamic properties of the data are accounted for. This is crucial for long-term predictions, which are important for the developments in Paper F.

Background and Contribution: This work started during my pre-doctoral visit at Imperial College in London, where I spent three months collaborating with Dr. Marc Deisenroth. Starting from a small code base, I made the vast majority of the implementation. The text was written after that visit, mainly by me and Marc, and reviewed by Thomas.

Paper F: From Pixels to Torques: Policy Learning with Deep Dynamical Models

> N. Wahlström, T. B. Schön, and M. P. Deisenroth. From pixels to torques: Policy learning with deep dynamical models. In *Deep Learning Workshop at the International Conference on Machine Learning (ICML)*, Lille, France, July 2015b.

Summary: In this paper the pixels-to-torques problem is analyzed, where an agent must learn a closed-loop control policy from pixel information only. We use the DDM introduced in the previous paper together with an adaptive model predictive control strategy for getting a closed-loop control. Compared to state-of-the-art reinforcement learning methods for continuous states and actions, the proposed approach learns fast, scales to high-dimensional state spaces, and is an important step toward fully autonomous learning from pixels to torques. After this work, John Assael, PhD candidate at Oxford University, entered the project. With some more computationally efficient code and state-of-the-art learning for the neural networks, we also managed to control a two-link arm (Assael et al., 2015).

Background and Contribution: This work was done after my pre-doctoral visit but still in close collaboration with Marc Deisenroth. Also for this paper the vast majority of the implementation was done by me and the writing was done together with Marc and reviewed by Thomas.

Paper G: Discretizing Stochastic Dynamical Systems using Lyapunov Equations

> N. Wahlström, P. Axelsson, and F. Gustafsson. Discretizing stochastic dynamical systems using Lyapunov equations. In *Proceedings of the The 19th World Congress of the International Federation of Automatic Control (IFAC)*, pages 3726–3731, Cape Town, South Africa, August 2014.

Summary: Stochastic state space models are fundamental in state estimation, system identification and control. System models are often provided in continuous time, while a major part of the applied theory is developed for discrete-time systems. Discretization of continuous-time models is hence fundamental. In this paper, we present a novel algorithm using a combination of Lyapunov equations and analytical solutions, enabling efficient implementation in software. The proposed method circumvents numerical problems exhibited by standard algorithms in the literature. Both theoretical and simulation results are provided.

Background and Contribution: This work started from a perspective of using Kalman filters for doing Gaussian process regression (see for example work by

Hartikainen and Särkkä (2010)). For making the connection between Kalman filtering and Gaussian processes, discretization of continuous time state space models is necessary. After coming up with the core idea, discussions continued with Partrik Axelsson who also had analyzed numerical aspects of the time update in Kalman filter using Lyapunov equations (Axelsson and Gustafsson, 2015). All code and almost all text were written by me and final manuscript was reviewed by Patrik and Fredrik.

1.7 Other Publications

The following additional publications have been authored or co-authored by myself, but are not included in this thesis:

F. Ceragioli, G. Lindmark, C. Veibäck, N. Wahlström, M. Lindfors, and C. Altafini. A bounded confidence model that preserves the signs of the opinion. In *European Control Conference*, 2015. Submitted.

J.-A. M. Assael, N. Wahlström, T. B. Schön, and M. P. Deisenroth. Data-efficient learning of feedback policies from image pixels using deep dynamical models. In *Deep Reinforcement Learning Workshop at the Annual Conference on Neural Information Processing Systems (NIPS)*, Montréal Canada, December 2015. Accepted.

A. Solin, M. Kok, N. Wahlström, T. B. Schön, and S. Särkkä. Modeling and interpolation of the ambient magnetic field by Gaussian processes. *Pre-print arXiv:1509.04634*, September 2015.

G. Hendeby, F. Gustafsson, and N. Wahlström. Teaching Sensor Fusion and Kalman Filtering using a Smartphone. In *Proceedings of the The 19th World Congress of the International Federation of Automatic Control (IFAC)*, pages 10586–10591, Cape Town, South Africa, August 2014.

V. Deleskog, H. Habberstad, G. Hendeby, D. Lindgren, and N. Wahlström. Robust NLS sensor localization using MDS initialization. In *Proceedings of 17th International Conference on Information Fusion (FUSION)*, Madrid, Spain, July 2014.

N. Wahlström and F. Gustafsson. Magnetometer modeling and validation for tracking metallic targets. *IEEE Transactions on Signal Processing*, 62(3):545–556, 2014.

M. Kok, N. Wahlström, T. B. Schön, and F. Gustafsson. MEMS-based inertial navigation based on a magnetic field map. In *Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 6466–6470, Vancouver, Canada, May 2013.

N. Wahlström, F. Gustafsson, and S. Åkesson. A Voyage to Africa by Mr Swift. In *Proceedings of the 15th International Conference on Information Fusion (FUSION)*, pages 808–815, Singapore, July 2012a.

N. Wahlström, R. Hostettler, F. Gustafsson, and W. Birk. Rapid classification of vehicle heading direction with two-axis magnetometer. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3385–3388, Kyoto, Japan, March 2012b.

F. Gustafsson and N. Wahlström. Method and device for pose tracking using vector magnetometers, 2012. Patent. Under revision.

N. Wahlström, J. Callmer, and F. Gustafsson. Single target tracking using vector magnetometers. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4332–4335, Prague, Czech Republic, May 2011.

N. Wahlström, J. Callmer, and F. Gustafsson. Magnetometers for tracking metallic targets. In *Proceedings of 13th International Conference on Information Fusion (FUSION)*, Edinburgh, Scotland, July 2010.

E. Almqvist, D. Eriksson, A. Lundberg, E. Nilsson, N. Wahlström, E. Frisk, and M. Krysander. Solving the ADAPT benchmark problem - A student project study. In *21st International Workshop on Principles of Diagnosis (DX-10)*, Portland, Oregon, USA, October 2010.

2

Mathematical Modeling

In many scientific disciplines models are used to explain the reality. In psychology, mental models are used to explain how humans perceive the real world and learn from previous experiences. In atom physics, the well-known Bohr model is used to explain emission patterns and chemical reactions by describing atoms as positively charged nucleus surrounded by negatively charged orbiting electrons. In environmental science, global climate models are used to describe complex interconnections between atmosphere and oceans to predict weather and climate.

The purpose of a model is to explain, generalize and predict phenomena in the real world. Therefore, the model should not encode all aspects of the reality it is trying to describe. On the contrary, a model is by construction a simplification of the real world. This simplification is a necessity in order to have a tool that can be used to predict, generalize and explain the real world. For example, we know that the Bohr model is not quite correct.[1] Nevertheless, the Bohr model can be used to explain the emission line of atomic hydrogen. To take this argument even further, George Box made the famous statement "All models are wrong but some are useful" (Box, 1979). The most useful model might not even be the one that fits data the best, but rather the one that provides the best result in some performance measure, which depends on the application.

As an abstraction of the reality, a model can be described in terms of symbols, flow charts, or a computer program. In this thesis we consider mathematical models. Such models are described using a mathematical language. This could be a mathematical function, set of differential equations, or probabilistic description of the data.

Mathematical models can, according to Ljung (1999), be built either by

[1] In more modern quantum mechanics, the electron is rather considered to be a cloud of probability.

(a) *modeling*, i.e., combining previously known subsystems, for example using known physical law, into bigger models, or by

(b) *system identification*, i.e., inferring a model directly based on experimental data.

In this thesis, examples will be provided using both of these model building techniques. In Paper A, B, C and D method (a) is applied and in Paper E and F method (b) is used.

In the remaining part of this chapter, three different model components are introduced, which are used throughout this thesis. These include state-space models (Section 2.1), neural networks (Section 2.2) and Gaussian processes (Section 2.3).

2.1 State-Space Models

A *state-space model* is a mathematical model of a dynamical system. It relates inputs $\mathbf{u}_k$ and outputs $\mathbf{y}_k$ of the system by introducing a latent state $\mathbf{x}_k$. These quantities are related to each other via a first order difference equation

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), \tag{2.1a}$$
$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k). \tag{2.1b}$$

Note that most dynamical models can be reformulated into a state-space model, for instance, the *nonlinear auto-regressive exogenous model* (NARX), which is used in Paper E and F.

Example 2.1: NARX model

Consider a nonlinear difference equation in which the next output $\mathbf{y}_{k+1}$ depends on the past n outputs and m inputs as

$$\mathbf{y}_{k+1} = \tilde{\mathbf{f}}(\mathbf{y}_k, \ldots, \mathbf{y}_{k-n+1}, \mathbf{u}_k, \mathbf{u}_{k-m+1}). \tag{2.2a}$$

This difference equation can be formulated on state-space form with the state

$$\mathbf{x}_k = [\mathbf{y}_k^\mathsf{T}, \ldots, \mathbf{y}_{k-n+1}^\mathsf{T}, \mathbf{u}_{k-1}^\mathsf{T}, \ldots, \mathbf{u}_{k-m+1}^\mathsf{T}]^\mathsf{T} \tag{2.2b}$$

and the motion and measurement model as

$$\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) = \tilde{\mathbf{f}}(\mathbf{y}_k, \ldots, \mathbf{y}_{k-n+1}, \mathbf{u}_k, \ldots, \mathbf{u}_{k-m+1}), \tag{2.2c}$$
$$\mathbf{h}(\mathbf{x}_k) = \begin{bmatrix} I & 0 & \ldots & 0 \end{bmatrix} \mathbf{x}_k. \tag{2.2d}$$

2.1.1 Stochastic State-Space Models

As already stated, no model is perfectly correct. To account for uncertainty, the state-space model is usually extended to include process noise $\mathbf{w}_k$ and measurements noise $\mathbf{e}_k$ (for ease of notation, we omit the inputs $\mathbf{u}_k$)

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k) + \mathbf{g}(\mathbf{x}_k)\mathbf{w}_k, \tag{2.3a}$$
$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{e}_k. \tag{2.3b}$$

Here, $\mathbf{w}_k$ and $\mathbf{e}_k$ are discrete-time stochastic processes. A useful way to model $\mathbf{w}_k$ and $\mathbf{e}_k$ is to consider them to be white, which means that they are mutually independently (Jazwinski, 1970). With this assumption, the model can also be described using conditional densities for the transition and the observation

$$p(\mathbf{x}_{k+1}|\mathbf{x}_k), \tag{2.4a}$$
$$p(\mathbf{y}_k|\mathbf{x}_k). \tag{2.4b}$$

This will make it easier to estimate the state sequence $\mathbf{x}_k$ based on the measurements $\mathbf{y}_k$ because a filter solution can be designed by alternating between computing the filter distribution $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ and the prediction distribution $p(\mathbf{x}_{k+1}|\mathbf{y}_{1:k})$.

Furthermore, the noise terms $\mathbf{w}_k$ and $\mathbf{e}_k$ are usually assumed to be Gaussian. This simplifies the filtering operation even further. For example, if the stochastic state-space model (2.3) is linear and $\mathbf{w}_k$ and $\mathbf{e}_k$ are white Gaussian random sequences, the filtering problem can be solved with the Kalman filter (Kailath et al., 2000).

A further reason to model $\mathbf{w}_k$ and $\mathbf{e}_k$ to be Gaussian is that many physical processes are approximately Gaussian (Jazwinski, 1970). The noise terms are a collection of many unmodeled random effect, all of which are independent. When these effects are added to each other, their total contribution is approximately Gaussian, regardless of their individual distributions. This is also the essence of the central limit theorem.

Stochastic state-space models can be identified directly from data (compare with item (b) on page 20) using different methods. A prediction error method (Ljung, 1999) can be used to estimate system parameters included in the model. Also subspace methods (Van Overschee and De Moor, 1996) can be used (mainly for linear models).

On the other hand, in many applications, the model is derived from physical laws (compare with item (a) on page 19). Most physical models are defined in continuous time. Therefore, if the model is derived using physical modeling, the continuous state-space model has to be discretized to reach a model in the form (2.3). Before reaching that point, we will first formalize the continuous-time stochastic state-space model.

2.1.2 Continuous-Time Stochastic State-Space Models

To find a continuous-time counterpart of (2.3a), we consider the stochastic differential equation

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{a}\Big(\mathbf{x}(t)\Big) + \mathbf{b}\Big(\mathbf{x}(t)\Big) \cdot \text{“noise”}, \tag{2.5}$$

where the term “noise” is a continuous-time process noise term that corresponds to the white Gaussian process noise term $\mathbf{w}_k$ in (2.3a). By following Öksendal (2007), the connection to (2.3a) can informally be made by first considering the case in which both the noise and the state are 1-dimensional

$$\frac{dx(t)}{dt} = a\Big(x(t)\Big) + b\Big(x(t)\Big) \cdot \text{“noise”}. \tag{2.6}$$

Now, the discrete version of (2.6) becomes

$$x_{k+1} - x_k = a(x_k)T_k + \underbrace{b(x_k) \cdot \text{“noise”}\, T_k}_{\Delta\beta_k}, \tag{2.7a}$$

in which

$$x_k = x(t_k), \qquad T_k = t_{k+1} - t_k, \qquad \Delta\beta_k = \beta(t_{k+1}) - \beta(t_k), \tag{2.7b}$$

and $\beta(t)$ is some suitable stochastic process. By comparing with the discrete-time model (2.3a), the increment $\Delta\beta_k$ corresponds to the white noise term w_k. We now want to select $\boldsymbol{\beta}(t)$ such that the increments $\Delta\beta_k$ are Gaussian and independent. This is fulfill by assuming $\boldsymbol{\beta}(t)$ to be a *Brownian motion*. A Brownian motion is a process with independent increments that are Gaussian distributed according to $\beta(t) - \beta(s) \sim \mathcal{N}(0,\ t - s)$ for $t \geq s$, which also can be written as $\mathbb{E}[d\beta(t)d\beta(t)] = dt$.

Further, from (2.7) we obtain

$$x_k = x_0 + \sum_{j=0}^{k-1} a(x_j)T_j + \sum_{j=0}^{k-1} b(x_j)\Delta\beta_j \tag{2.8}$$

and by letting the sample time $T_k \to 0$ we obtain

$$x(t) = x_0 + \int_0^t a\Big(x(s)\Big)ds + \int_0^t b\Big(x(s)\Big)d\beta(s), \tag{2.9}$$

which also can be written as

$$dx(t) = a\Big(x(t)\Big)dt + b\Big(x(t)\Big)d\beta(t). \tag{2.10}$$

Now (2.10), is only meaningful if the integrals in (2.9) are well defined. Following Jazwinski (1970), the first integral can be defined as a Riemann integral, whereas

the second integral is defined as an *Itô integral*. The interested reader can refer to Öksendal (2007); Jazwinski (1970) for more details.

The derivation can also be extended to a multi-dimensional setting

$$d\mathbf{x}(t) = \mathbf{a}\Big(\mathbf{x}(t)\Big)dt + \mathbf{b}\Big(\mathbf{x}(t)\Big)d\boldsymbol{\beta}(t), \qquad \text{where} \qquad \mathbb{E}[d\boldsymbol{\beta}(t)d\boldsymbol{\beta}(t)^{\mathsf{T}}] = S\,dt, \tag{2.11}$$

Here, $\boldsymbol{\beta}(t)$ is a multi-dimensional Brownian motion. By dividing both sides with dt, the model (2.11) can informally be written as a differential equation

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{a}\Big(\mathbf{x}(t)\Big) + \mathbf{b}\Big(\mathbf{x}(t)\Big)\mathbf{w}(t), \qquad \text{where} \qquad \mathbb{E}[\mathbf{w}(t)\mathbf{w}(s)^{\mathsf{T}}] = S\delta(t-s), \tag{2.12}$$

where $\mathbf{w}(t) = \frac{d\boldsymbol{\beta}(t)}{dt}$ is a white delta-correlated Gaussian process.

However, (2.12) is just an informal presentation of a continuous-time stochastic state-space model for two reasons. First, a white delta-correlated Gaussian process has a constant power spectral density (hence white), which requires infinite power and it is therefore not physically realizable. Secondly, $\mathbf{w}(t)$ is not mean square Riemann integrable, and (2.12) has therefore no mathematical meaning (Jazwinski, 1970).

The continuous-time model (2.11) will now be used to complete the connection to its discrete-time counterpart (2.3).

2.1.3 Discretization of Stochastic State-Space Models

Consider a linear version of the continuous-time model (2.11)

$$d\mathbf{x}(t) = A\mathbf{x}(t)dt + Bd\boldsymbol{\beta}(t), \qquad \text{where} \qquad \mathbb{E}[d\boldsymbol{\beta}(t)d\boldsymbol{\beta}(t)^{\mathsf{T}}] = S\,dt. \tag{2.13}$$

By integrating (2.13) over the time interval $[t_k, t_{k+1}]$, we can find its discrete-time equivalence as

$$\mathbf{x}(t_{k+1}) = \underbrace{e^{AT_k}}_{F}\,\underbrace{\mathbf{x}(t_k)}_{\mathbf{x}_k} + \underbrace{\int_{t_k}^{t_{k+1}} e^{A(t_{k+1}-\tau)}Bd\boldsymbol{\beta}(\tau)}_{\mathbf{w}_k}. \tag{2.14}$$

This can be stated as a discrete-time stochastic difference equation

$$\mathbf{x}_{k+1} = F\mathbf{x}_k + G\mathbf{w}_k, \qquad \text{where} \qquad \mathbb{E}[\mathbf{w}_k\mathbf{w}_l^{\mathsf{T}}] = Q_{T_k}\delta_{kl}, \tag{2.15}$$

where δ_{kl} is the Kronecker delta function. By following, for example Jazwinski (1970), which requires the aforementioned Itô calculus, the noise $\mathbf{w}_k$ will be zero-mean, white Gaussian and its covariance is given by

$$GQG^{\mathsf{T}} = \int_0^{T_k} e^{A\tau}BSB^{\mathsf{T}}e^{A^{\mathsf{T}}\tau}d\tau. \tag{2.16a}$$

Together with the discrete-time system matrix

$$F = e^{AT_k}, \tag{2.16b}$$

this completes the discretization procedure. The integral (2.16a) is analyzed further in Paper G, which proposes a solution based on Lyapunov equations for solving the integral.

In most physical models, the continuous-time process noise only affects a subset of the state (meaning that $\dim[\mathbf{w}(t)] < \dim[\mathbf{x}(t)] \Rightarrow \operatorname{rank}(BSB^\mathsf{T}) < \dim[\mathbf{x}(t)]$), which means that BSB^T does not have full rank. Nevertheless, GQG^T will as a result of the integration (2.16) in most cases have it. In these cases, a discrete-time process noise $\mathbf{w}_k$ needs to be of the same dimension as the state, i.e., $\dim(\mathbf{w}_k) = \dim(\mathbf{x}_k)$, to encode the same properties as the continuous-time model even though $\mathbf{w}(t)$ only affects a few state components.

We end this section with an example. This example generalizes each of the three discretizations presented in Section 5 in Paper A, with $M = I$ in Section 5.1, $M = C(\mathbf{m}(t_k))$ in Section 5.2, and $M = \bar{S}(\bar{\mathbf{q}}(t_k))$ in Section 5.3.

Example 2.2

Consider a system in the form

$$A = \begin{bmatrix} 0_{m\times m} & M \\ 0_{n\times m} & 0_{n\times n} \end{bmatrix}, \qquad B = \begin{bmatrix} 0_{m\times n} \\ I_n \end{bmatrix}, \tag{2.17}$$

where $M \in \mathbb{R}^{m\times n}$. Because A is nilpotent with $A^i = 0$ for $i \geq 2$, we obtain that

$$e^{A\tau} = I + A\tau = \begin{bmatrix} I_m & \tau M \\ 0_{m\times n} & I_n \end{bmatrix}. \tag{2.18}$$

By using this in (2.16), we obtain

$$F = \begin{bmatrix} I_m & TM \\ 0_{n\times m} & I_n \end{bmatrix}, \quad GQG^\mathsf{T} = \begin{bmatrix} \frac{T^3}{3}MSM^\mathsf{T} & \frac{T^2}{2}MS \\ \frac{T^2}{2}SM^\mathsf{T} & TS \end{bmatrix}, \tag{2.19a}$$

where the last term, for example, can be realized as

$$G = \begin{bmatrix} T^2M & 0_{m\times n} \\ 0_{n\times n} & TI_n \end{bmatrix}, \quad Q = \frac{1}{T}\begin{bmatrix} \frac{1}{3}S & \frac{1}{2}S \\ \frac{1}{2}S & S \end{bmatrix}. \tag{2.19b}$$

2.2 Neural Networks

A *neural network* is nonlinear function $\mathbf{y} = \mathbf{g}(\mathbf{u}; \boldsymbol{\theta})$ suitable for modeling complex relations between an input $\mathbf{u}$ (not to be confused with the control input in Section 2.1) and an output $\mathbf{y}$ using the parameters $\boldsymbol{\theta}$.

A one-layer neural network is a composition of a linear mapping $A\mathbf{u} + \mathbf{b}$ and a nonlinear function $\sigma(\mathbf{u})$ according to

$$\mathbf{y} = \mathbf{g}(\mathbf{u}) = \sigma(A\mathbf{u} + \mathbf{b}), \tag{2.20}$$

in which $\sigma(\cdot) = [\sigma(\cdot), \dots, \sigma(\cdot)]^{\mathsf{T}}$ operates element-wise on its input. The function is parametrized with the elements in the matrix A and the vector $\mathbf{b}$. To model complex relations, a multi-layer neural network is commonly used, which can be achieved by composing multiple layers as

$$\mathbf{z}^{(l+1)} = \sigma(A^{(l)}\mathbf{z}^{(l)} + \mathbf{b}^{(l)}), \tag{2.21}$$

where the first layer is the input $\mathbf{z}^{(1)} = \mathbf{u}$, the last layer is the output $\mathbf{y} = \mathbf{z}^{(N-1)}$ and the parameter vector $\boldsymbol{\theta}$ is a concatenation of the elements in all the matrices $A_1 \dots A_{N-1}$ and vectors $\mathbf{b}_1 \dots \mathbf{b}_{N-1}$

The scalar nonlinear function $\sigma(\cdot)$ is called the activation function and is the ingredient that makes the neural network nonlinear.

Various functional forms for the activation functions exist. Common choices are the logistic sigmoid (2.22a) and the hyperbolic tangent (2.22b), whereas the rectified linear unit (2.22c) has gained increasing popularity and is nowadays the most popular activation function in deep neural networks (LeCun et al., 2015).

$$\underbrace{\sigma(z) = \frac{1}{1 + e^{-1}}}_{\text{logistic sigmoid (2.22a)}}, \qquad \underbrace{\sigma(z) = \frac{e^{z} - e^{-1}}{e^{z} + e^{-1}}}_{\text{hyperbolic tangent (2.22b)}}, \qquad \underbrace{\sigma(z) = \max(0, z)}_{\text{rectfied linear unit (2.22c)}}. \tag{2.22}$$

By writing each layer (2.21) on indexed form

$$z_i^{(l+1)} = \sigma\left(\sum_{j=1}^{M^{(l)}} a_{ij}^{(l)} z_j^{(l)} + b_i\right), \tag{2.23}$$

it is clear that each dimension in one layer depends on all dimensions of the previous layer. This is commonly illustrated by a directed graph, in which each node corresponds to a hidden unit $z_j^{(l)}$ and each edge corresponds to the weight $a_{ij}^{(l)}$, see Figure 2.1.

The neural network can be trained by minimizing the squared error between the output of the model $\mathbf{g}(\mathbf{u}_k; \boldsymbol{\theta})$ and the measurement $\mathbf{y}_k$

$$\arg\min_{\boldsymbol{\theta}} V(\boldsymbol{\theta}), \qquad \text{where} \qquad V(\boldsymbol{\theta}) = \sum_{k=1}^{N} \|\mathbf{y}_k - \mathbf{g}(\mathbf{u}_k, \boldsymbol{\theta})\|^2. \tag{2.24}$$

The gradient of $V(\boldsymbol{\theta})$ with respect to θ can be calculated efficiently using the *back-propagation algorithm*, which essentially is a recursive use of the chain rule on each layer in the network. The details are covered in many textbooks, for example in Bishop (2006).

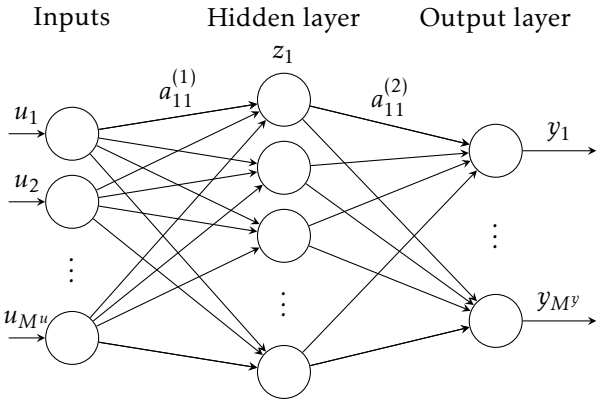
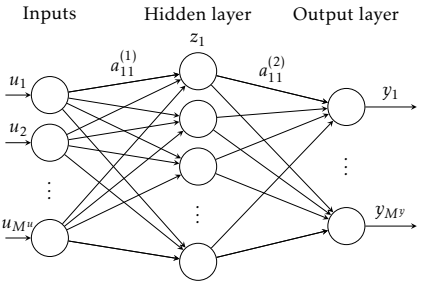


Figure 2.1: *A sketch of a two-layer neural network. The input units u_i, the hidden units z_i and output units y_i are represented with nodes and the weights a_{ij} are represented with edges.*

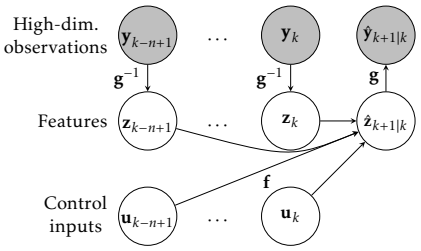
2.2.1 Deep Learning

Although the first conceptual ideas of neural networks date back to the 1940s (McCulloch and Pitts, 1943), they had their first main success stories in the late 1980s and early 1990s with the use of the back-propagation algorithm. At that stage, neural networks could, for example, be used to classify hand written digits from low-resolution images (LeCun et al., 1990). However, in the late 1990s neural networks were largely forsaken because it was widely thought that they could not be used to solve any challenging problems in computer vision and speech recognition (LeCun et al., 2015). In these areas, neural networks could not compete with hand-crafted solutions based on domain specific prior knowledge.

This picture has changed dramatically since the late 2000s. Since then, neural networks have had a new revival under the name *deep learning*. Progress in software, hardware and algorithm parallelization made it possible to address more complicated problems, which where unthinkable only a couple of decades ago. For example, in image recognition, these deep models are now the dominant methods of use and reach almost human performance on some specific tasks (LeCun et al., 2015). Recent advances based on deep neural networks have generated algorithms that can learn how to play computer games based on pixel information only (Mnih et al., 2015), and automatically understand the situation in images for automatic caption generation (Xu et al., 2015). As a part of this development, more attention has been directed to systems that can automatically adapt and learn actions from images based on little prior knowledge of the scenario. In Paper E and F, this problem has been addressed by proposing a deep dynamical model. This model is based on the autoencoder, which is introduced in the next section.



(a) *Autoencoder: The encoder maps the original data $\mathbf{y}_k \in \mathbb{R}^M$ onto its low-dimensional representation $\mathbf{z}_k = \mathbf{g}^{-1}(\mathbf{y}_k) \in \mathbb{R}^m$, where $m \ll M$; the decoder maps this feature back to a high-dimensional representation $\mathbf{y}_k = \mathbf{g}(\mathbf{z}_k)$. The gray color represents high-dimensional observations.*



(b) *Prediction model: Past features are encoded via $\mathbf{z}_k = \mathbf{g}^{-1}(\mathbf{y}_k)$. The transition model predicts the feature $\hat{\mathbf{z}}_{k+1|k}$ at the next time step based on the n-step history of n past features $\mathbf{z}_{k-n+1}, \ldots, \mathbf{z}_k$ and control inputs $\mathbf{u}_{k-n+1}, \ldots, \mathbf{u}_k$. A future high-dimensional observation $\hat{\mathbf{y}}_{k+1}$ can be predicted via the decoder $\hat{\mathbf{y}}_{k+1|k} = \mathbf{g}^{-1}(\mathbf{y}_k)$. The gray color represents high-dimensional observations.*

Figure 2.2: *Model components of the deep dynamical model.*

2.2.2 Autoencoder

Neural networks are normally used in a supervised fashion. The input $\mathbf{u}$ can, for example, represent the pixel values of an image, and the output $\mathbf{y}$ could represent the class depending on what the image is illustrating, e.g., a cat, a dog, or a car. By using a lot of labeled data $\{\mathbf{u}_k, \mathbf{y}_k\}_{k=1:N}$, the network can learn the function to correctly classify new unseen images. However, in this thesis, the neural network is used in an unsupervised fashion, where these labels do not exist. This is enabled via the autoencoder.

The *autoencoder* is a neural network suitable for learning compressed representations $\{\mathbf{z}_k\}_{1:N}$ of high-dimensional data $\{\mathbf{y}_k\}_{1:N}$, where $\dim(\mathbf{y}_k) \gg \dim(\mathbf{z}_k)$. The network architecture is based on an input layer and an output layer with equally many nodes. In addition, at least one of the hidden layers has fewer nodes than the input and output layers, see Figure 2.2a. By training this network with the same data as input and output, the narrower hidden layer needs to encode all information about the data from the input in order to reconstruct it on the output. The values of these hidden nodes $\mathbf{z}_k$ can therefore be seen as a low-dimensional representation of $\mathbf{y}_k$.

The first half of the autoencoder is called the encoder $\mathbf{z} = \mathbf{g}^{-1}(\mathbf{y})$ and the second half the decoder $\mathbf{y} = \mathbf{g}(\mathbf{z})$. After the training, the encoder and the decoder will be approximate inverses of each other, since $\mathbf{g}(\mathbf{g}^{-1}(\mathbf{y})) \approx \mathbf{y}$. We can then easily transform either $\mathbf{y}$ into $\mathbf{z}$ or $\mathbf{z}$ into $\mathbf{y}$ using either the encoder or the decoder. The access to both of these two mappings is advantageous for defining the deep

dynamical model that is explained in the next section.

2.2.3 Deep Dynamical Model

The encoder and the decoder are usually combined with each other to encode and decode the very same data point. In Paper E and Paper F, a dynamical version of this model is proposed, here referred to as a *deep dynamical model* (DDM). In this model, (1) past data points are encoded via the encoder to a set of past features, (2) a future feature is predicted using a NARX model (2.1), and (3) the predicted feature is decoded via the decoder to a predicted data point. This process is illustrated in Figure 2.2b. This model is used to autonomously learn models and design controllers purely based on high-dimensional pixel information from images. The model and the control design is further discussed and analyzed in Paper E and Paper F.

2.3 Gaussian Processes

Already in Section 2.1.2 we have encountered a white Gaussian process $\mathbf{w}(t)$. The concept can be extended by considering colored Gaussian processes and also to consider Gaussian processes with multi-dimensional inputs. A *Gaussian process* (GP) can be interpreted as a distribution over functions. The unknown function is denoted by $f(\mathbf{u})$ hereafter, where $\mathbf{u} \in \mathbb{R}^n$ and $f(\mathbf{u}) \in \mathbb{R}$. Rasmussen and Williams (2006) provide somewhat a formal definition of a GP:

Definition 2.1. "A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution."

A GP can consequently be seen as a generalization of the multivariate Gaussian probability distribution in the sense that the function values evaluated for a finite number of inputs $\mathbf{u}_1, \dots, \mathbf{u}_N$ are normally distributed

$$\begin{bmatrix} f(\mathbf{u}_1) \\ \vdots \\ f(\mathbf{u}_N) \end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}, K), \quad \text{where} \quad \boldsymbol{\mu} = \begin{bmatrix} \mu(\mathbf{u}_1) \\ \vdots \\ \mu(\mathbf{u}_N) \end{bmatrix}, \quad K = \begin{bmatrix} k(\mathbf{u}_1, \mathbf{u}_1) & \cdots & k(\mathbf{u}_1, \mathbf{u}_N) \\ \vdots & & \vdots \\ k(\mathbf{u}_N, \mathbf{u}_1) & \cdots & k(\mathbf{u}_N, \mathbf{u}_N) \end{bmatrix}.$$

Here, $\mu(\mathbf{u})$ is the *mean function*, and $k(\mathbf{u}, \mathbf{u}')$ is the *covariance function* defined as

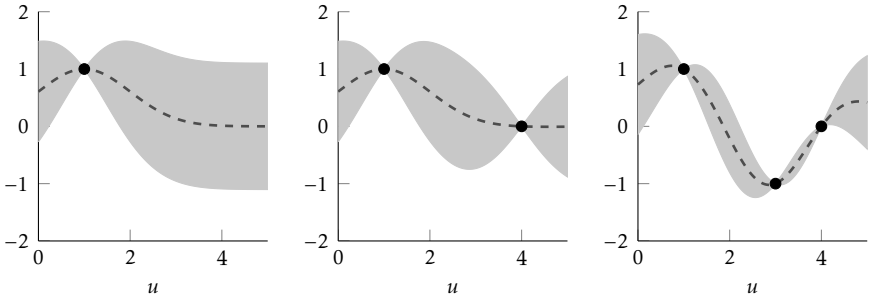
$$\mu(\mathbf{u}) = \mathbb{E}[f(\mathbf{u})], \tag{2.25a}$$

$$k(\mathbf{u}, \mathbf{u}') = \mathbb{E}[(f(\mathbf{u}) - \mu(\mathbf{u}))(f(\mathbf{u}') - \mu(\mathbf{u}'))]. \tag{2.25b}$$

The GP is then denoted as

$$f(\mathbf{u}) \sim \mathcal{GP}\Big(\mu(\mathbf{u}), k(\mathbf{u}, \mathbf{u}')\Big). \tag{2.26}$$

Although the GP is just a slight conceptual extension of the multi-variate Gaussian distribution, it can be used to model and encode fairly complex assumptions. A flavor of this will be provided in this section. Before digging into the modeling aspects of GPs, the GP regression will be outlined in the next section.



(a) One measurement *(b)* Two measurements *(c)* Three measurements

Figure 2.3: *Posterior of a GP after using one, two and three measurements. The posterior mean and the posterior variance (±3 standard deviations) are shown with a dashed line and gray region, respectively. The measurements are illustrated with black dots.*

2.3.1 The Covariance Function

The covariance function $k(\mathbf{u}, \mathbf{u}')$ determines how the function values are correlated. One commonly used covariance function is the exponential of a quadratic form, also called the squared exponential, with an additional constant term.

$$k(\mathbf{u}, \mathbf{u}') = \sigma_f^2 e^{-\frac{\|\mathbf{u}-\mathbf{u}'\|^2}{2l^2}} + \sigma_b^2, \tag{2.27}$$

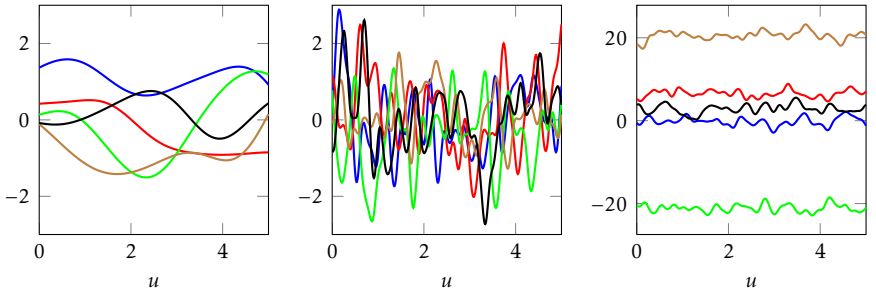
The parameters σ_f, l and σ_b (also called hyperparameters) determine the magnitude and length scale of the correlation. For this specific covariance function, σ_f determines the standard deviation of the function's fluctuations, l determines the characteristic length scale of these fluctuations, and σ_b determines the standard deviation of the constant level around which the function fluctuates. In Figure 2.4, a few samples from a GP using this covariance function are displayed with different sets of hyperparameters.

2.3.2 Gaussian Process Regression

The GP can be used to learn a function $y = f(\mathbf{u})$ by using a data set of inputs and outputs $\{\mathbf{u}_k, y_k\}_{1:N}$. We consider a measurement model in the form

$$y_k = f(\mathbf{u}_k) + e_k, \qquad \text{where} \qquad e_k \sim \mathcal{N}(0, \sigma_n^2). \tag{2.28}$$

The GP prior (2.26), the measurement model (2.28) and the data set $\{y_k, \mathbf{u}_k\}_{1:N}$ can be used to predict a yet unseen output f_* at any location $\mathbf{u}_*$. To do this, we first note that all measurements $y_1, \ldots, y_N$ together with the unseen output f_* are



(a) $\sigma_f = 1, l = 1, \sigma_b = 0$ *(b)* $\sigma_f = 1, l = 0.1, \sigma_b = 0$ *(c)* $\sigma_f = 1, l = 0.1, \sigma_b = 10$

Figure 2.4: *Samples from a GP prior using the covariance function* (2.27)*. The samples in each plot have been realized with different values of the hyperparameters* σ_f*,* l *and* σ_b*.*

jointly Gaussian

$$\left[\begin{array}{c} y_1 \\ \vdots \\ y_N \\ \hdashline f_* \end{array}\right] \sim \mathcal{N}\left(\left[\begin{array}{c} \mu(u_1) \\ \vdots \\ \mu(u_N) \\ \hdashline \mu(u_*) \end{array}\right], \left[\begin{array}{ccc:c} k(\mathbf{u}_1,\mathbf{u}_1)+\sigma_n^2 & \cdots & k(\mathbf{u}_1,\mathbf{u}_N) & k(\mathbf{u}_1,\mathbf{u}_*) \\ \vdots & \ddots & \vdots & \vdots \\ k(\mathbf{u}_N,\mathbf{u}_1) & \cdots & k(\mathbf{u}_N,\mathbf{u}_N)+\sigma_n^2 & k(\mathbf{u}_N,\mathbf{u}_*) \\ \hdashline k(\mathbf{u}_*,\mathbf{u}_1) & \cdots & k(\mathbf{u}_*,\mathbf{u}_N) & k(\mathbf{u}_*,\mathbf{u}_*) \end{array}\right]\right),$$

which in a more compact form can be written as

$$\left[\begin{array}{c} \mathbf{y} \\ \hdashline f_* \end{array}\right] \sim \mathcal{N}\left(\left[\begin{array}{c} \boldsymbol{\mu} \\ \hdashline \mu_* \end{array}\right], \left[\begin{array}{c:c} K+\sigma_n^2 I & \mathbf{k}_* \\ \hdashline \mathbf{k}_*^\mathsf{T} & k_{**} \end{array}\right]\right). \tag{2.29}$$

Based on (2.29), the conditional distribution $f_*|\mathbf{y}$ can be computed using standard formulas for the conditional Gaussian distribution

$$f_*|\mathbf{y} \sim \mathcal{N}(a,\ b) \quad \text{where} \quad a = \mu_* + \mathbf{k}_*^\mathsf{T}(K+\sigma_n^2 I)^{-1}(\mathbf{y}-\boldsymbol{\mu}), \tag{2.30a}$$

$$b = k_{**} - \mathbf{k}_*^\mathsf{T}(K+\sigma_n^2 I)^{-1}\mathbf{k}_*. \tag{2.30b}$$

The conditional distribution can be extended for multiple inputs $\mathbf{u}_1^*, \ldots, \mathbf{u}_{N_*}^*$ to obtain a posterior over a grid of function values $f_1^*, \ldots, f_{N_*}^*$. The posterior of these function values can be interpreted as a posterior over the whole function.

The GP regression is illustrated in Figure 2.3. The figure illustrates how the uncertainty decreases as more measurements are added. Also, the uncertainty is low for the points close to the measurements, and higher for the points distant from the measurements as expected. The interested reader might want to refer to Rasmussen and Williams (2006) for further details about GP regression.

Throughout the rest of this chapter, the focus will be on various modeling aspects of GPs; i.e., determining the functional form of the mean function $\mu(\mathbf{u})$ and the covariance function $k(\mathbf{u}, \mathbf{u}')$. In many situations the mean function can be set to zero without loss of generality. The main part of the modeling therefore

takes place in the design of the covariance function $k(\mathbf{u}, \mathbf{u}')$. We will discuss this function in greater detail with emphasis on the covariance functions that appear in Paper C and Paper D.

By comparing Figure 2.4a with Figure 2.4b, we can clearly see the effect of changing the characteristic length scale l. With a small l, the function samples change much faster than with a large l. Further, by comparing Figure 2.4b with Figure 2.4c, the effect of the constant term σ_b^2 can be observed. For $\sigma_b^2 = 0$, the function samples are centered around zero and for $\sigma_b^2 \gg 0$, the function samples are centered around a level $b \sim \mathcal{N}\left(0,\ \sigma_b^2\right)$.

2.3.3 Periodic Covariance Function

One way to construct new covariance functions is to combine and modify already existing ones. In Rasmussen and Williams (2006, Section 4.2.4) various methods for this purpose are presented. One method for constructing new covariance functions is to use the output of another static function $\mathbf{g}(\mathbf{u})$ as input to an already existing covariance function. The new covariance function is then given by $k(\mathbf{g}(\mathbf{u}), \mathbf{g}(\mathbf{u}'))$.

For example, a periodic covariance function can be constructed by using the following periodic nonlinear function

$$\mathbf{g}(u) = [\cos(u),\ \ \sin(u)]^\mathsf{T}. \tag{2.31}$$

Together with the squared exponential covariance function (2.27), we obtain

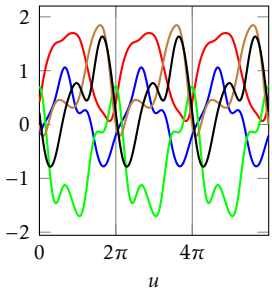


Figure 2.5: *Samples from a GP prior using the periodic covariance function* (2.32).

$$k\left(\mathbf{g}(u), \mathbf{g}(u)'\right) = \sigma_f^2 e^{-\frac{(\cos(u)-\cos(u'))^2+(\sin(u)-\sin(u'))^2}{2l^2}} = \sigma_f^2 e^{-\frac{2\sin^2\left(\frac{u-u'}{2}\right)}{l^2}}. \tag{2.32}$$

In Figure 2.5, a few samples based on this covariance function are displayed. Clearly, all samples do obey the periodic assumption encoded in the covariance function. Paper D uses this covariance function for modeling the contour of an extended target in polar coordinates, for which the angle is the input, and the radial distance from the center to the contour is the output.

2.3.4 Derivative and Integral Observations

It is well-known that Gaussian distributions are closed under linear transformation. In a similar manner, Gaussian processes are closed under linear operations (Papoulis and Pillai, 1991; Rasmussen and Williams, 2006; Hennig and Kiefel, 2013; Garnett, 2015). By following the notation of Garnett (2015), here we denote this linear operator $L_{\mathbf{w}}[f]$, where $\mathbf{w}$ is some argument of the operator. This

linear operator could for example be differentiation or integration, which are defined as

$$L_w[f] = \frac{\partial f(u)}{\partial u}\Big|_{u=w}, \qquad \text{and} \qquad L_{[a,b]}[f] = \int_a^b f(u)du. \tag{2.33}$$

By applying the functional $L_{\mathbf{w}}[\cdot]$ on both the mean function and the covariance function, the GP prior for $L_{\mathbf{w}}[f]$ is given by

$$L_{\mathbf{w}}[f] \sim \mathcal{GP}\Big(L_{\mathbf{w}}[\mu], L^2_{\mathbf{w},\mathbf{w}'}[k]\Big), \tag{2.34}$$

where $L^2_{\mathbf{w},\mathbf{w}'}[k]$ acts on both the first and second argument of $k(\cdot,\cdot)$ as

$$L^2_{\mathbf{w},\mathbf{w}'}[k] = L_{\mathbf{w}}\Big[L_{\mathbf{w}'}[k(\mathbf{u},\cdot)]\Big] = L_{\mathbf{w}'}\Big[L_{\mathbf{w}}[k(\cdot,\mathbf{u}')]\Big]. \tag{2.35}$$

For example, by considering the differentiation operator, the derivative of a GP is another GP with the following mean and covariance functions

$$\frac{\partial f(u)}{\partial u}\Big|_{u=w} \sim \mathcal{GP}\left(\frac{\partial \mu(u)}{\partial u}\Big|_{u=w} \quad , \quad \frac{\partial^2 k(u,u')}{\partial u \partial u'}\Big|_{u=w,u'=w'}\right). \tag{2.36}$$

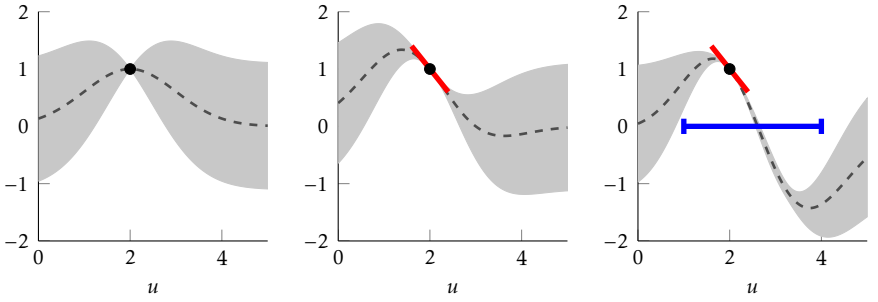
Moreover, not only $L_{\mathbf{w}}[f]$ is a GP; $L_{\mathbf{w}}[f]$ is also jointly Gaussian distributed with $f(\mathbf{u})$ (and also with any other linear operation of f). By considering two different points $\mathbf{u}_1$ and $\mathbf{w}_2$, this can be written as

$$\begin{bmatrix} f(\mathbf{u}_1) \\ L_{\mathbf{w}_2}[f] \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\mathbf{u}_1) \\ L_{\mathbf{w}_2}[\mu] \end{bmatrix}, \begin{bmatrix} k(\mathbf{u}_1,\mathbf{u}_1) & L_{\mathbf{w}_2}[k(\mathbf{u}_1,\cdot)] \\ L_{\mathbf{w}_2}[k(\cdot,\mathbf{u}_1)] & L^2_{\mathbf{w}_2,\mathbf{w}_2}[k] \end{bmatrix}\right), \tag{2.37}$$

where the cross-covariance between $f(\mathbf{u}_1)$ and $L_{\mathbf{w}_2}[f]$ is given by the off-diagonal terms in the covariance matrix. Consequently, observations of $L_{\mathbf{w}}[f]$ can be used to make predictions of $f(\mathbf{u})$, and vice versa.

Since both differentiation and integration are linear operators, see (2.33), we can use GPs to make predictions about a function $f(\mathbf{u})$ based on measurements of both its derivative and integral. Figure 2.6 shows the posterior of a GP in a 1D toy example using measurements of the function, its derivative, and its integral. The posterior clearly obeys the three constraints provided by the measurements and still provides the appropriate confidence interval based on the remaining uncertainty of the function.

Derivative observations have previously been used to for learning dynamical systems (Solak et al., 2003), to model light from supernovas (Holsclaw et al., 2013) and in Bayesian optimization (Osborne, 2010). However, integral observations have been less exploited in the literature (Hennig and Kiefel, 2013). Hennig and Kiefel (2013) used it to obtain a probabilistic interpretation of quasi-Newton methods. In Bayesian quadrature we want to do the opposite; given observations of the function $f(u)$, we want to obtain a posterior of the integral $L_p[f] = \int f(u)p(u)du$ under some distribution p. This can be used to derive a



(a) Function observation *(b) Derivative observation* *(c) Integral observation*

Figure 2.6: *Posterior of a GP after using (a) one measurement $f(2) = 1$, (b) one additional derivative measurement $f'(2) = -1$, and finally (c) one additional integral measurement $\int_1^4 f(u)du = 0$. The measurements are illustrated with a black dot, red line and blue line, respectively. As in Figure 2.3, the dashed line and the gray region represent the posterior mean and the posterior variance, respectively.*

Bayesian Mote Carlo method for evaluating such integrals. This method relies on an active sampling scheme that maximizes information gain by using the uncertainty $L_p[f]$, (see for example Osborne, 2010).

In this thesis, we use the derivative of a GP to derive two multi-variate covariance functions used in Paper C. These two covariance functions are explained in Section 2.3.6. Before this step, we first need to extend the scalar GP $f(\mathbf{u})$ to a multi-variate GP $\mathbf{f}(\mathbf{u})$.

2.3.5 Vector-Valued GPs

A GP can also be used as a prior over functions with multi-dimensional outputs $\mathbf{f}(\mathbf{u}) = [f_1(\mathbf{u}), \ldots, f_m(\mathbf{u})]^\mathsf{T}$. The most obvious extension is to consider each output dimension to be an independent one-dimensional GP

$$f_i(\mathbf{u}) \sim \mathcal{GP}\Big(\mu(\mathbf{u}), k(\mathbf{u}, \mathbf{u}')\Big). \tag{2.38}$$

We can also choose to introduce a correlation between the different output dimensions by considering a matrix-valued covariance function

$$\mathbf{f}(\mathbf{u}) \sim \mathcal{GP}\Big(\boldsymbol{\mu}(\mathbf{u}), K(\mathbf{u}, \mathbf{u}')\Big), \tag{2.39}$$

in which each component of $\boldsymbol{\mu}(\mathbf{u})$ and $K(\mathbf{u}, \mathbf{u}')$ are defined as

$$\mu_i(\mathbf{u}) = \mathbb{E}[f_i(\mathbf{u})], \tag{2.40a}$$

$$[K(\mathbf{u}, \mathbf{u}')]_{ij} = \mathbb{E}\Big[\Big(f_i(\mathbf{u}) - \mu_i(\mathbf{u})\Big)\Big(f_j(\mathbf{u}') - \mu_j(\mathbf{u}')\Big)\Big]. \tag{2.40b}$$

Álvarez et al. (2012) provide an excellent review of covariance functions for vector-valued functions. In the next section, two specific covariance function for vector-valued GPs will be introduced. Both of them are used in Paper C.

2.3.6 Divergence- and Curl-Free Covariance Functions

In Section 2.3.4, we explained how derivative observations could be incorporated in the GP regression. Since the derivative of a GP is another GP, we can also use it to construct new covariance functions. Two such functions are presented in this section. Both of them are used to model vector-valued GPs that can be derived by differentiation of *potentials*. These two covariance functions are used in Paper C to model magnetic fields obeying certain constraints from the physics. The motivation of these constraints is explained further in Section 3.6 in connection with the electromagnetic theory.

Curl-Free Covariance Function

Consider a vector field $\mathbf{g}(\mathbf{u}) = [g_1(\mathbf{u}),\ g_2(\mathbf{u}),\ g_3(\mathbf{u})]^\mathsf{T}$, where $\mathbf{u} = [u_1,\ u_2,\ u_3]^\mathsf{T}$. Such a vector field is known to be *conservative* if it can be written as the gradient of a scalar function $f(\mathbf{u})$ (scalar potential)

$$\mathbf{g}(\mathbf{u}) = \nabla f(\mathbf{u}) \quad \Leftrightarrow \quad g_i(\mathbf{u}) = \frac{\partial f(\mathbf{u})}{\partial u_i}. \tag{2.41}$$

These vector fields also obey the relation

$$\nabla \times \mathbf{g}(\mathbf{u}) = \nabla \times \Big(\nabla f(\mathbf{u})\Big) = \mathbf{0}, \tag{2.42}$$

which means that they are curl-free.

We model the scalar potential using a GP with a squared exponential covariance function

$$f(\mathbf{u}) \sim \mathcal{GP}\Big(\mathbf{0}, k(\mathbf{u}, \mathbf{u}')\Big), \qquad \text{where} \qquad k(\mathbf{u}, \mathbf{u}') = \sigma_f^2 e^{-\frac{\|\mathbf{u}-\mathbf{u}'\|^2}{2l^2}}. \tag{2.43}$$

Since differentiation is a linear operator (see Section 2.3.4), the vector field $\mathbf{g}(\mathbf{u}) \sim \mathcal{GP}\Big(\mathbf{0}, K(\mathbf{u}, \mathbf{u}')\Big)$ will also be a GP. By following the relations discussed in Section 2.3.4, the covariance function for $\mathbf{g}(\mathbf{u})$ is given by

$$K(\mathbf{u}, \mathbf{u}') = \frac{\sigma_f^2}{l^2}\left(I_3 - \left(\frac{\mathbf{u}-\mathbf{u}'}{l}\right)\left(\frac{\mathbf{u}-\mathbf{u}'}{l}\right)^\mathsf{T}\right) e^{-\frac{1}{2l^2}\|\mathbf{u}-\mathbf{u}'\|^2}. \tag{2.44}$$

The details of the derivation are outlined in Appendix A. Since $K(\mathbf{u}, \mathbf{u}')$ is not a diagonal matrix, the constraints imposed by (2.41) introduce cross-correlation between the different output-dimensions in $\mathbf{g}(\mathbf{u})$. Further, any sample from a GP prior with the covariance function (2.44) will obey the *curl-free* property (2.42). Therefore, this covariance function is suitable for modeling curl-free vector fields.

Divergence-Free Covariance Function

A vector field $\mathbf{g}(\mathbf{u})$ is known to be *solenoidal* if it can be written as the curl of another vector field $\mathbf{f}(\mathbf{u})$ (vector potential)

$$\mathbf{g}(\mathbf{u}) = \nabla \times \mathbf{f}(\mathbf{u}) \quad \Leftrightarrow \quad g_i(\mathbf{u}) = \sum_{j=1}^{3} \sum_{k=1}^{3} \epsilon_{ijk} \frac{\partial f_j(\mathbf{u})}{\partial u_k}, \tag{2.45}$$

where ϵ_{ijk} is the Levi-Civita symbol defined as

$$\epsilon_{ijk} = \begin{cases} +1 & \text{if } (i,j,k) \text{ is } (1,2,3), (2,3,1) \text{ or } (3,1,2), \\ -1 & \text{if } (i,j,k) \text{ is } (3,2,1), (1,3,2) \text{ or } (2,1,3), \\ 0 & \text{if } i = j \text{ or } j = k \text{ or } k = i. \end{cases} \tag{2.46}$$

The vector fields also obey the relation

$$\nabla \cdot \mathbf{g}(\mathbf{u}) = \nabla \cdot \Big(\nabla \times \mathbf{f}(\mathbf{u}) \Big) = 0, \tag{2.47}$$

which means that they are divergence-free.

Similar as for the scalar potential, we model the vector potential $\mathbf{f}(\mathbf{u})$ with a GP, in which each component $f_i(\mathbf{u})$ is a GP with a squared exponential covariance function as in (2.43). By following the details of the derivation outlined in Appendix A, the covariance function of $\mathbf{g}(\mathbf{u})$ is then given by

$$K(\mathbf{x}, \mathbf{x}') = \frac{\sigma_f^2}{l^2} \left(\left(2 - \left\| \frac{\mathbf{x} - \mathbf{x}'}{l} \right\|^2 \right) I_3 + \left(\frac{\mathbf{x} - \mathbf{x}'}{l} \right) \left(\frac{\mathbf{x} - \mathbf{x}'}{l} \right)^{\mathsf{T}} \right) e^{-\frac{1}{2l^2} \|\mathbf{x} - \mathbf{x}'\|^2}. \tag{2.48}$$

This covariance function is suitable for modeling divergence-free vector fields, since all function samples from a GP with this covariance function do obey the divergence-free property (2.45).

Both (2.44) and (2.48) are used in Paper C to model magnetic fields. Note, in that paper, the parametrization of the hyperparameters is slightly different, where σ_f^2/l^2 is replaced with σ_f^2.

2.4 Summary and Connections

In this chapter, an overview of three model components, all of which are used in this thesis, has been presented. These model components include

- **State-space models:** These models describe dynamical systems via a latent state. In particular, this thesis uses state-space models to track magnetic objects in Paper A, and extended targets in Paper D. In Paper E and F, the NARX model is used, which can be interpreted as a state-space model (see Example 2.1). Further, in Paper G, a procedure for discretizing these models is described.

- **Neural networks:** This model describes complex functional relationship between an input and an output. In Paper E and Paper F a deep dynamical model is studied, in which neural networks are one model component.
- **Gaussian processes:** Gaussian processes are, in contrast to a neural networks, a non-parametric model that describes functional relationship between inputs and outputs. Is it not as flexible as neural networks, at least not in its standard formulation. However, in contrast to the neural network, it allows you to encode certain constraints in the model. This model is used in Paper C to describe magnetic fields, and in Paper D to describe the spatial extent of targets.

In the next chapter, the electromagnetic theory is presented. This theory is needed to derive the functional form for some of the models above.

3

Electromagnetic Theory

This chapter introduces the electromagnetic theory. Special interest is given to the magnetic field and its properties when currents are steady. The material presented in this chapter is to a great extent based on the theory as presented by Jackson (1998) and Cheng (1989).

Although electromagnetic phenomena were known to the ancient Greeks, its development as a quantitative subject started first in the end of the 18th century with Cavendish' and Columbus' experiments and research. Fifty years later, Faraday was studying time-varying electromagnetic phenomena. By 1865 James Clerk Maxwell had published his famous paper (Maxwell, 1865) on a dynamical theory of electromagnetic fields. In that paper, the original set of four equations first appeared. These equations will be our theoretical starting point.

3.1 Maxwell's Equations

All electromagnetic phenomena are governed by *Maxwell's equations*,

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\varepsilon_0}, \tag{3.1a}$$

$$\nabla \times \mathbf{B} - \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} = \mu_0 \mathbf{J}, \tag{3.1b}$$

$$\nabla \times \mathbf{E} + \frac{\partial \mathbf{B}}{\partial t} = \mathbf{0}, \tag{3.1c}$$

$$\nabla \cdot \mathbf{B} = 0, \tag{3.1d}$$

in which $\mathbf{E}$ is the *electric field*, ρ is the *charge density*, $\mathbf{B}$ is the *magnetic field* and $\mathbf{J}$ is the *current density*. Apart from the fields $\mathbf{E}$ and $\mathbf{B}$ and their sources ρ and $\mathbf{J}$, the equations also include the constants ε_0 and μ_0, which are the *permittivity* and

permeability of free space, respectively. The permeability of free space μ_0 has a defined value

$$\mu_0 = 4\pi \times 10^{-7}\,\mathrm{F\,m^{-1}} = 1.257 \times 10^{-6}\,\mathrm{H\,m^{-1}}, \tag{3.2}$$

and the permittivity of free space is defined by

$$\varepsilon_0 = 1/(\mu_0 c_0^2) = 8.854 \times 10^{-12}\,\mathrm{F\,m^{-1}}, \tag{3.3}$$

where c_0 is the *speed of light* in vacuum. Table 3.1 summarizes the meaning of each symbol and the SI unit of measure.

Table 3.1: *Definitions and units.*

Symbol	Meaning	SI unit
$\mathbf{E}$	Electric field	Volt per meter [$\mathrm{V\,m^{-1}}$]
$\mathbf{B}$	Magnetic field	Tesla [T]
ρ	Charge density	Coulomb per cubic meter [$\mathrm{C\,m^{-3}}$]
$\mathbf{J}$	Current density	Ampere per square meter [$\mathrm{A\,m^{-2}}$]
ε_0	Permittivity of free space	Farad per meter [$\mathrm{F\,m^{-1}}$]
μ_0	Permeability of free space	Henry per meter [$\mathrm{H\,m^{-1}}$]
c_0	Speed of light in vacuum	Meter per second [$\mathrm{m\,s^{-1}}$]

3.2 Quasi-Static Approximation

In this thesis, the magnetic field $\mathbf{B}$ is of special interest, since that is the quantity that will be measured. In general, it is coupled with the electric field $\mathbf{E}$ through their time derivatives (3.1b) and (3.1c). However, in the stationary case, the terms $\partial\mathbf{E}/\partial t$ and $\partial\mathbf{B}/\partial t$ can be neglected, and Maxwell's equations become decoupled. The magnetic field will therefore obey the *magnetostatic equations*

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J}, \tag{3.4a}$$

$$\nabla \cdot \mathbf{B} = 0. \tag{3.4b}$$

From these equations it is clear that the current density $\mathbf{J}$ can be regarded as the source causing the magnetic field $\mathbf{B}$. If the current density is time-dependent, the static assumption does not hold and the full solution of Maxwell's equations has to be considered. However, if these changes are sufficiently small, a *quasi-static approximation* can be made, in which the magnetostatic equations (3.4) still hold. In this approximation, (3.4a) states that any variation in the current density $\mathbf{J}$ is instantaneously communicated to the magnetic field $\mathbf{B}$, implying that the velocity of propagation is infinite. Hence, this approximation is only valid if the time lag produced by the finite velocity of propagation is very small in comparison

with the time it takes for the currents to change value (Di Bartolo, 2004). To be more specific, assume that the currents undergo changes with a certain frequency. Following Di Bartolo (2004), the condition of quasi-stationarity can be expressed as

$$\text{size of the physical system} \times \text{frequency} \ll c_0.$$

This is illustrated with the following example borrowed from Di Bartolo (2004).

Example 3.1

For a transformer with the characteristic size of 30 cm and a frequency of 50 Hz, we have

$$\text{size} \times \text{frequency} = 0.3\,\text{m} \times 50\,\text{s}^{-1} = 15\,\text{m}\,\text{s}^{-1} \ll c_0 = 300\,000\,000\,\text{m}\,\text{s}^{-1}$$

Thus, in this transformer, the quasi-static approximation is valid.

3.3 Magnetic Dipole Moment

Having complete knowledge of the current density $\mathbf{J}$ enables us to find the solution for the magnetic field $\mathbf{B}$ using the magnetostatic equations (3.4). However, its solution is nontrivial and further approximations have to be made. In many scenarios of interest the current density is zero except in one small region V, further on referred to as the *object*, see Figure 3.1. Instead of solving the full system

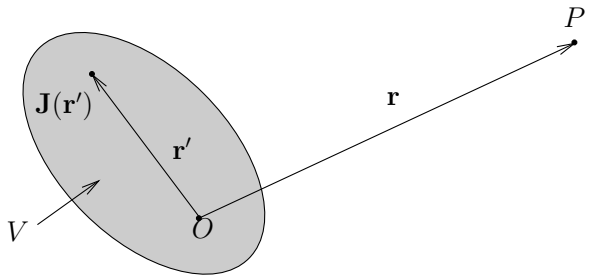


Figure 3.1: *Localized current density $\mathbf{J}(\mathbf{r}')$ in the region V resulting in a magnetic induction at the point P with coordinate $\mathbf{r}$. Outside the region V the current density is zero $\mathbf{J}(\mathbf{r}') = \mathbf{0}$.*

(3.4), we can represent region V with its *magnetic dipole moment* $\mathbf{m}$ at point O. This dipole moment is related to the current density as

$$\mathbf{m} \triangleq \frac{1}{2}\int_V \mathbf{r}' \times \mathbf{J}(\mathbf{r}')d^3r' \quad [\text{A}\,\text{m}^2], \tag{3.5}$$

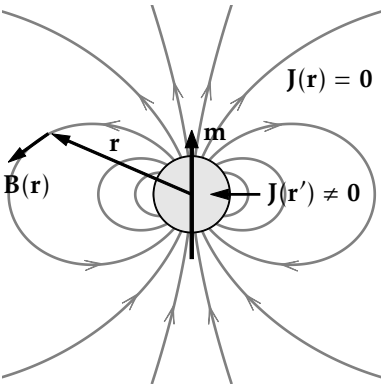


Figure 3.2: An object with a non-zero current density $\mathbf{J}(\mathbf{r}') \neq \mathbf{0}$ *induces a magnetic dipole field, here visualized with field lines. Via* (3.5)*, a magnetic dipole moment* $\mathbf{m}$ *can be associated with the object. The magnetic dipole field* $\mathbf{B}(\mathbf{r}')$ *can be computed by using* (3.6)*.*

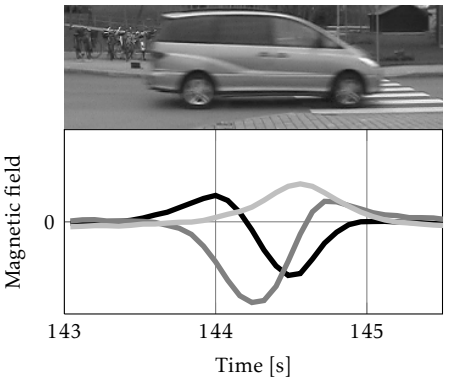


Figure 3.3: A metallic vehicle induces a magnetic field, which is measured with a magnetometer deployed at the road side. All three Cartesian components of this measurement are shown. The vehicle can be modeled as a moving magnetic dipole.

where $\mathbf{r}'$ is an integration variable, $d^3r' = dx'dy'dz'$ is a three-dimensional volume element at $\mathbf{r}'$, and $\mathbf{J}(\mathbf{r}')$ is the current density at position $\mathbf{r}'$. This formulation enables an analytical expression that relates the magnetic dipole moment to the magnetic field. This relation is called the magnetic dipole field

$$\mathbf{B}(\mathbf{r}) = \frac{\mu_0}{4\pi}\frac{3(\mathbf{r}\cdot\mathbf{m})\mathbf{r} - \|\mathbf{r}\|^2\mathbf{m}}{\|\mathbf{r}\|^5} = \underbrace{\frac{\mu_0}{4\pi\|\mathbf{r}\|^5}(3\mathbf{r}\mathbf{r}^\mathsf{T} - \|\mathbf{r}\|^2 I_3)}_{\triangleq L(\mathbf{r})}\mathbf{m} = L(\mathbf{r})\mathbf{m}, \tag{3.6}$$

in which $\mathbf{r}$ is a vector from the object O to the *observer* at point P. The magnetic dipole field is visualized in Figure 3.2. The derivation of this dipole field can be found in Appendix B and requires a Taylor expansion, which is only valid if the distance to the observer is large in comparison to the size of the object, i.e., if $\|\mathbf{r}\| \gg \|\mathbf{r}'\|$ in Figure 3.1. The magnetic dipole model can be used to describe the relation between measurement of a stationary magnetometer and a moving object that induces a magnetic field. This is illustrated in (3.3). This dipole model is the theoretical starting point for the models presented in Paper A and Paper B.

Furthermore, if there are multiple objects, each of them will induce its own magnetic dipole field

$$\mathbf{B}_i(\mathbf{r}) = L(\mathbf{r} - \mathbf{r}_i)\mathbf{m}_i, \tag{3.7}$$

where $\mathbf{r}_i$ is the position of object i and $\mathbf{m}_i$ its dipole moment. Due to the linearity

of the magnetostatic equations (3.4), all magnetic dipole fields will superimpose

$$\mathbf{B}(\mathbf{r}) = \sum_i \mathbf{B}_i(\mathbf{r}) = \sum_i L(\mathbf{r} - \mathbf{r}_i)\mathbf{m}_i. \tag{3.8}$$

This model is proposed in Paper A to represent objects including multiple magnetic dipoles. Multiple dipoles can also be used to represent larger objects by dividing them into smaller regions and representing each region with a magnetic dipole. This idea was advocated in Wahlström and Gustafsson (2014).

The idea can also be taken one step further. By dividing the object into even smaller regions, we reach, in the limit, a vector field in which each point in space has a magnetic dipole moment associated with it. This is called the *magnetization* and is explained in the next section.

3.4 Magnetization

All materials consist of atoms with orbiting electrons. The electrons generate circular currents that can be represented with magnetic dipole moments $\mathbf{m}_i$. All these dipole moments will contribute to the magnetization of the material. However, due to the very large number of atoms in a material, this description is not convenient. Therefore, we use its average over a larger volume and describe magnetization as the quantity of magnetic moment per unit volume. If there are n atoms per unit volume, we define the *magnetization* as

$$\mathbf{M}(\mathbf{r}) \triangleq \lim_{\Delta V \to 0} \frac{\sum_{i=1}^{n\Delta V} \mathbf{m}_i}{\Delta V} \quad [\mathrm{A\,m^{-1}}], \tag{3.9}$$

in which $\sum_{i=1}^{n\Delta V} \mathbf{m}_i$ denotes the sum of all magnetic dipole moments within the volume element ΔV centered at $\mathbf{r}$. According to this definition, the magnetization can be considered a density of magnetic dipole moments. This description is convenient because it allows us to describe complex magnetized regions without using a large number of magnetic dipole moments.

By generalizing (3.8), we can also find a relation between the magnetization and the magnetic field

$$\mathbf{B}(\mathbf{r}) = \int L(\mathbf{r} - \mathbf{r}')\mathbf{M}(\mathbf{r}')d^3r'. \tag{3.10}$$

In this formulation, the magnetic dipole can be considered a Dirac-distribution, illustrated in the example below.

Example 3.2

Consider a magnetization localized at position $\mathbf{r}_i$ and zero elsewhere $\mathbf{M} = \mathbf{m}\delta(\mathbf{r} - \mathbf{r}_i)$. Inserting this field into (3.10) we obtain

$$\mathbf{B}(\mathbf{r}) = \int L(\mathbf{r} - \mathbf{r}')\mathbf{m}\delta(\mathbf{r}' - \mathbf{r}_i)d^3r' = L(\mathbf{r} - \mathbf{r}_i)\mathbf{m}, \tag{3.11}$$

which is the magnetic dipole field of a magnetic dipole at $\mathbf{r}_i$ with magnetic dipole moment $\mathbf{m}$.

However, the description (3.10) is complicated and highly nonlinear. By introducing the *magnetizing field*, we can instead describe the relation between the magnetization $\mathbf{M}$ and the magnetic field $\mathbf{B}$ with a set of linear differential equations similar to (3.4). This is explained in the next section.

3.5 Magnetizing Field

The current density $\mathbf{J}$ can be divided into the *magnetization current* $\mathbf{J}_m$ and the *free current* $\mathbf{J}_f$ as

$$\mathbf{J} = \mathbf{J}_m + \mathbf{J}_f. \tag{3.12}$$

The magnetization current is bound to the material and can be seen as the current density due the electrons orbiting around the nucleus in the material. The magnetization current is defined as the curl of the magnetization

$$\mathbf{J}_m \triangleq \nabla \times \mathbf{M} \quad [\mathrm{A\,m^{-2}}]. \tag{3.13}$$

The free current density $\mathbf{J}_f$ is not bound to any material and its charges can move freely, e.g., currents in electric cables.

Using (3.12) in the magnetostatic equation (3.4a) results in

$$\frac{1}{\mu_0}\nabla \times \mathbf{B} = \mathbf{J}_f + \mathbf{J}_m = \mathbf{J}_f + \nabla \times \mathbf{M}, \tag{3.14}$$

or equivalently

$$\nabla \times \left(\frac{\mathbf{B}}{\mu_0} - \mathbf{M} \right) = \mathbf{J}_f. \tag{3.15}$$

For convenience, a new field $\mathbf{H}$ can be defined called the *magnetizing field*

$$\mathbf{H} \triangleq \frac{\mathbf{B}}{\mu_0} - \mathbf{M} \quad [\mathrm{A\,m^{-1}}]. \tag{3.16}$$

With this quantity, an alternative version of the magnetostatic equation, only including the free current $\mathbf{J}_f$, can be introduced

$$\nabla \times \mathbf{H} = \mathbf{J}_f, \tag{3.17a}$$

$$\nabla \cdot \mathbf{B} = \mathbf{0}. \tag{3.17b}$$

By this we have reformulated the magnetostatic equations (3.4) into (3.17), which only depend on the free current $\mathbf{J}_f$ and not on the magnetization current $\mathbf{J}_m$. Instead the effect of the magnetization current is described via the magnetization $\mathbf{M}$, which couples the two magnetic fields $\mathbf{B}$ and $\mathbf{H}$ via (3.16).

The equations (3.16) and (3.17) are used in Paper C for modeling complex magnetic structures in indoor environments using Bayesian nonparametric methods. In the paper, we assume there is no free current, i.e., $\mathbf{J}_f = \mathbf{0}$. With this assumption the $\mathbf{H}$- and $\mathbf{B}$-fields are curl- and divergence-free, respectively. These properties are exploited in the proposed model.

3.6 Magnetic Potentials

A useful strategy for solving problems in magnetostatics (and in electrostatics) is to work with potentials. Instead of studying the $\mathbf{B}$-field and $\mathbf{H}$-field directly, we can choose to study their corresponding potentials. The main motivation for introducing these potentials is to reformulate the magnetostatic equations (3.17) into a set of differential equations, which are easier to solve. However, our motivation for introducing these potentials is to derive probabilistic models for the $\mathbf{B}$- and the $\mathbf{H}$-field, which are used in Paper C.

There exist two different magnetic potentials potentials; (i) the scalar potential and the (ii) vector potential. Their existence relies on the following two important null-identities

$$\nabla \times (\nabla \varphi) \equiv 0, \tag{3.18a}$$

$$\nabla \cdot (\nabla \times \mathbf{A}) \equiv \mathbf{0}. \tag{3.18b}$$

These two identities can easily be proven in Cartesian coordinates. For general coordinate systems (see for example Cheng, 1989). We will discuss the two potentials corresponding to these two identities in the following subsections.

3.6.1 Magnetic Scalar Potential

In regions without any free current $\mathbf{J}_f = \mathbf{0}$, the $\mathbf{H}$-field will be curl-free

$$\nabla \times \mathbf{H} = \mathbf{0}. \tag{3.19}$$

By comparing (3.19) with the identity (3.18a), we can conclude that the $\mathbf{H}$-field can be written as the gradient of a scalar field

$$\mathbf{H} = -\nabla \varphi. \tag{3.20}$$

The scalar field φ is called the *magnetic scalar potential*. Note that the negative sign in the definition is of no importance but follows the standard conventions. The relation (3.20) can now be used to derive a probabilistic model for the $\mathbf{H}$-field by modeling φ as a Gaussian process. This was discussed in Section 2.3.6 with further details in Appendix A.

By using this potential, we can also derive a relation between the scalar potential and the magnetization causing it. If we use the divergence operator on (3.16), we obtain

$$\nabla \cdot \underbrace{\mathbf{H}}_{-\nabla\varphi} = \frac{1}{\mu_0} \underbrace{\nabla \cdot \mathbf{B}}_{=0} - \nabla \cdot \mathbf{M}. \tag{3.21}$$

This allows us to define the *effective magnetic-charge density*

$$\rho_{\mathbf{M}} = -\nabla \cdot \mathbf{M}, \tag{3.22}$$

and we obtain the following Poisson equation for φ

$$\nabla^2 \varphi = -\rho_{\mathbf{M}}. \tag{3.23}$$

Based on a certain distribution of the magnetic-charge density $\rho_{\mathbf{M}}$, this can be used to solve for the potential φ. See Jackson (1998) for further examples and techniques for solving magnetostatic problems using the magnetic scalar potential. The magnetic scalar potential is also used in Solin et al. (2015) for modeling magnetic fields in indoor environments.

3.6.2 Magnetic Vector Potential

Since the **B**-field is divergence-free (3.17b), the identity (3.18b) leads to that the **B**-field can be written as the curl of another vector field

$$\mathbf{B} = \nabla \times \mathbf{A}. \tag{3.24}$$

The vector field **A** is called the *magnetic vector potential*. In a similar manner as for the scalar potential, this relation can be used to model the **B**-field by using a Gaussian process prior for the vector potential **A**. This was discussed already in Section 2.3.6 with further details in Appendix A.

With this definition, the magnetic vector potential **A** has a built-in ambiguity; we can add any curl-free field to **A** without any effect on **B**. To resolve this ambiguity, we also require, without loss of generality, that $\nabla \cdot \mathbf{A} = 0$ (this is the Lorenz gauge condition). With this assumption, we have that

$$\nabla \times \mathbf{B} = \nabla \times (\nabla \times \mathbf{A}) = \nabla(\underbrace{\nabla \cdot \mathbf{A}}_{0}) - \nabla^2 \mathbf{A} = -\nabla^2 \mathbf{A}. \tag{3.25}$$

Together with Ampere's law (3.4a), we obtain one Poisson equation for each component in **A**

$$\nabla^2 \mathbf{A} = -\mu_0 \mathbf{J}. \tag{3.26}$$

In a similar manner as for the scalar potential, these equations can also be used to solve problems in magnetostatic. In Appendix B, the Poisson equation (3.26) and its solution is also used to derive the magnetic dipole model.

3.7 Magnetic Materials

As explained earlier, all materials consist of atoms, each of which can be described with a magnetic dipole. In most materials, these dipoles are not structured in any certain manner and the total magnetization will therefore be zero. However, in some materials, these dipoles will be affected by the presence of an external field, which in turn would cause a non-zero magnetization. In these materials, we distinguish between properties referred to as *soft iron* and *hard iron*. The soft and hard iron effects were used in Wahlström and Gustafsson (2014) to model the induced magnetic moment in vehicles.

3.7.1 Soft Iron

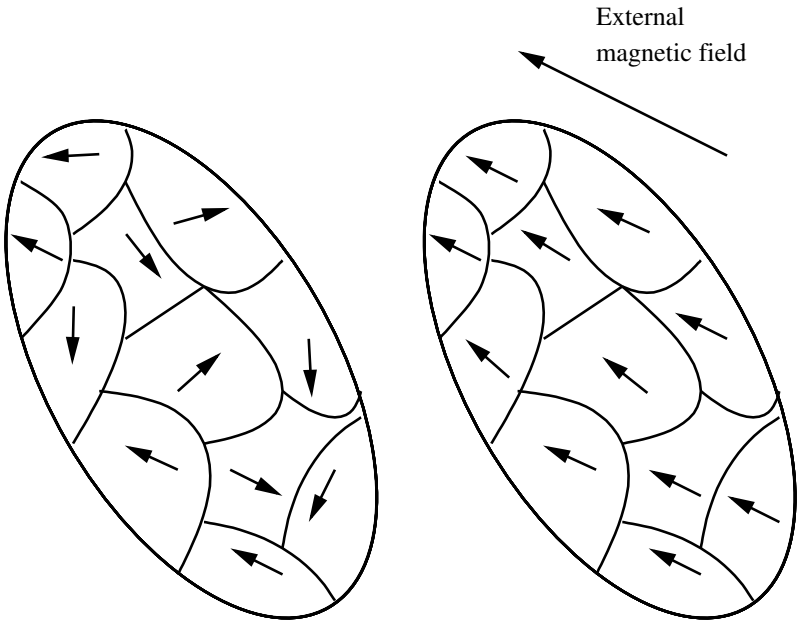
In soft iron, the magnetization is aligned with an applied external field, see Figure 3.4. If the material is linear and isotropic, this magnetization is directly proportional to the magnetizing field

$$\mathbf{M} = \chi_m \mathbf{H}, \tag{3.27}$$

where χ_m is the *magnetic susceptibility*, which is a dimensionless constant characteristic for the magnetic material. By using (3.16), this results in a relation between the magnetization $\mathbf{M}$ and the $\mathbf{B}$-field

$$\mathbf{M} = \frac{1}{\mu_0} \frac{\chi_m}{1 - \chi_m} \mathbf{B}. \tag{3.28}$$

Thus, the magnetization is always aligned with the external field regardless of the orientation of the object.



(a) *Non-magnetized material. All dipoles are independent of each other and in average we have* $\mathbf{M} \approx \mathbf{0}$.

(b) *Magnetized material. The dipoles are aligned in the direction of the applied external magnetic field and therefore* $\mathbf{M} \neq \mathbf{0}$.

Figure 3.4: *A material with* $\chi_m > 0$ *is magnetized when an external field is applied.*

3.7.2 Hard Iron

In soft iron materials, the magnetization vanishes when the external field is removed. However, for other materials, the magnetization is not reversible if strong magnetic fields are applied. This results in permanent magnets. Materials having this property are called ferromagnetic. Magnetized ferromagnetic substance also contributes to the total magnetization. Such permanent magnetization is referred to as hard iron. The hard iron magnetization is always be aligned with the reference frame of the magnetized object, whereas the soft iron magnetization is always aligned with the applied magnetic field.

3.8 Summary and Connections

In this chapter we present a short overview of the electromagnetic theory relevant for this thesis. Different concepts have been introduced, which are used in Paper A, B and C in the second part of this thesis.

The magnetic dipole model (3.6) has been introduced and proven to have properties suitable for modeling magnetic objects with a small geometrical extent. However, in the real world scenarios presented in each of these three papers, the extent of the magnetic objects is not always negligible, which affects the modeling. This issue is handled differently in each of the three papers.

- Paper A extends the dipole model by using multiple dipoles to model objects consisting of multiple magnets. Such multi-dipole objects break the symmetry in the magnetic field otherwise present in a single dipole vector field.
- The dipole model is also used in Paper B to derive a classifier that determines the driving direction of vehicles. This classifier is experimentally shown to give good classification results even though the target is not a point source, which the dipole model in fact assumes.
- In Paper C, the magnetization $\mathbf{M}$ (3.9) is used to model magnetic structures present in indoor environments. The magnetization can be seen as a continuum with infinitely many magnetic dipole moments. To handle these infinite-dimensional objects, Gaussian processes are used.

4

Concluding Remarks

In Part I, a background of the material in this thesis was provided. Chapter 1 introduced the research field and its applications, Chapter 2 described the concept of mathematical modeling by explaining three different models used throughout the thesis and Chapter 3 provided details about the electromagnetic theory needed to derive the models associated to magnetic fields.

The results were presented in the publications in Part II and the conclusions are summarized below in Section 4.1. Ideas for future directions of the research are provided in Section 4.2.

4.1 Conclusions

In Paper A, B, and C, the problem of localizing magnetic objects based on measurements of their induced magnetic field was investigated. The models introduced in these papers were all based on the electromagnetic theory presented in Chapter 3. However, to derive a model that works with real data is not straightforward. Especially, taking the object extent into consideration is challenging. This has been solved differently in each of the three papers.

In Paper A, a sensor description based on the magnetic dipole model was introduced. Based on this sensor model and a sensor network of magnetometers, both position and orientation of a small magnet can be tracked with high accuracy. The model was further developed to model extended magnetic objects that consist of multiple dipoles. The proposed model was validated using real data and compared with a high-precision optical reference system. Good tracking performance was achieved in the vicinity of the sensor network.

In Paper B, a method for classifying the driving direction of a vehicle was proposed. Similar to Paper A, the theoretical starting point was the magnetic dipole model. However, instead of using the full dipole model for estimating

position, orientation and velocity as in Paper A, a nonlinear transformation of the measurement was derived. Based on the dipole model, this transformed data has theoretically and experimentally been proven to be sensitive to the driving direction of the vehicle, but insensitive to other scenario dependent parameters, such as velocity, magnetic signature and target extent. Note that in contrast to an explicit use of the dipole model, this model works excellent on large objects, for example vehicles. The model was validated on a total number of 511 vehicles and outperformed a standard likelihood test.

The model presented in Paper C is not restricted to the parametric dipole model. It uses a non-parametric model called Gaussian processes to describe extended magnetic objects and their induced magnetic field. In contrast to the first two papers, this model is more suitable for stationary magnetic objects. The model has been validated on both simulated and experimental data showing ability to localize and determine the geometrical shape of magnetic structures.

Gaussian processes were also used in Paper D in which a new approach for tracking extended targets was proposed. The proposed method uses GPs to model the unknown target extent while simultaneously estimating the kinematic state of the target. An efficient algorithm was also proposed in which the filter updates are fully recursive and do not suffer from an increase in dimension with each available measurement, unlike the standard GP formulation. The performance and capabilities of the algorithm are demonstrated through simulations and real data experiments.

In Paper E and F, a data-efficient framework was proposed that learns controllers directly from pixel information. In contrast to Paper D, the model is not based on any geometrical assumption of the system. It only assumes that there exists a low-dimensional description of the data. Whereas the model was introduced in Paper E, it was used for control in Paper F. Compared to state-of-the-art controllers purely based on pixel information, the algorithm learns fairly quickly and scales to high-dimensional state spaces.

This thesis ended with Paper G in which an algorithm based on Lyapunov equations was used to discretize continuous-time stochastic dynamical models. In particular, an extension to systems with integrators was presented, where numerical evaluations showed that the proposed algorithm has advantageous numerical properties for slow sampling and fast dynamics in comparison with a standard method in the literature.

4.2 Future Work

Different directions are possible for future research related to the material presented in this thesis. These directions are divided into different application areas, which have been analyzed in this thesis.

Magnetic tracking for human-computer interaction The area of three-dimensional visualization and virtual reality has seen as tremendous growth the last decade. Such systems require input devices to enable intuitive and high qual-

ity computer-human interaction. Although existing techniques exist, magnetic tracking has benefits in terms of cost, robustness and flexibility. However, many research questions need to be solved to turn this into a high-precision localization technique. How should robust and reliable calibration routines be designed for this system? Could the range be increased by using an active instead of a passive system? In what other applications could the technique be used?

Traffic surveillance Future work should focus on finding solutions for cases that are hard to resolve, i.e., multiple vehicles, vehicles that are turning, possibly by fusing information from other sensor modalities. Further, so far the detection problem has not been analyzed. Here, standard methods from the literature can be used, for example the adaptive thresholding technique presented by Cheung and Varaiya (2007). In future work, the detection problem should be included, resulting in a complete system for localizing vehicles in a wireless sensor network. Finally, implementation in real sensor nodes should be targeted and the performance in a real time system analyzed.

Magnetic localization and mapping in indoor environments The initial work in this thesis has already been further developed in Solin et al. (2015), where larger scale problems were considered using more computational efficient algorithms for regression and using derivative measurements of the scalar potential instead of measurements from a curl-free vector field. However, in that contribution, the divergence-free property of the magnetic field was not exploited which we do exploit in Paper A. As a continuation of Paper C and Solin et al. (2015), it would be interesting to include that property as well. Further, an interesting extension is to implement a full SLAM (simultaneous localization and mapping) framework where the magnetic map and the position of the platform are estimated jointly.

Flexible models for extended target tracking The presented framework in Paper D only considers single target tracking. Future work should focus on developing a dedicated multi-target framework for this model. To increase robustness and avoid the approximations with the linearized filter (an Extended Kalman filter), more exact inference techniques including Sequential Monte Carlo and Variational Bayes should be considered. It would also be interesting to combine the proposed model with a Probability Hypothesis Density filter, which also treat the number of targets in a probabilistic manner. Another direction of research is to exploit the informative model of the target extent for high-accuracy gating and object classification.

Autonomous learning from raw pixel information Already today there exist systems that learn to autonomously control fairly complex system based on high-dimensional measurements. For example, Mnih et al. (2015) have used deep learning techniques to teach an agent to play different video games based on pixel information only. However, one limitation with existing techniques is that they do require a lot of data (millions of data points) to accomplish this task. The

approach in Paper F addresses this issue by using a low-dimensional dynamical model.This approach has potential to accomplish the same type of task, but with less data. Future work should include to test these ideas on more challenging problems. Although promising and impressive results already exist, the combination of deep learning and control is still in its infancy (LeCun et al., 2015).

Appendix

A

Derivation of Covariance Functions for Divergence-Free and Curl-Free Vector Fields

This appendix provides additional details to Section 2.3.6 in which two covariance functions for curl-free and divergence-free vector fields are derived.

A.1 Curl-Free Vector Fields

Based on identity (3.18a) we know that a curl-free vector field $\mathbf{H}$

$$\nabla \times \mathbf{H} = \mathbf{0}$$

can be written as the gradient of a scalar potential, such that

$$\mathbf{H} = -\nabla\varphi \qquad \Leftrightarrow \qquad H_i = -\frac{\partial\varphi}{\partial r_i}.$$

By modeling the scalar potential using a GP

$$\varphi(\mathbf{r}) \sim \mathcal{GP}(0, k(\mathbf{r}, \mathbf{r}')),$$

we have that also $\mathbf{H}(\mathbf{r})$ is a GP

$$\mathbf{H}(\mathbf{r}) \sim \mathcal{GP}(0, K_{\mathbf{H}}(\mathbf{r}, \mathbf{r}')),$$

for which each component of the covariance function is given by

$$\begin{aligned}
[K_{\mathbf{H}}]_{(ij)}(\mathbf{r}, \mathbf{r}') &= \operatorname{Cov}\left[H_i(\mathbf{r}),\ H_j(\mathbf{r}')\right] \\
&= \operatorname{Cov}\left[\frac{\partial}{\partial r_i}\varphi(\mathbf{r}),\ \frac{\partial}{\partial r_j}\varphi(\mathbf{r}')\right] \\
&= \frac{\partial^2}{\partial r_i \partial x_j'} k(\mathbf{r}, \mathbf{r}').
\end{aligned}$$

By using a squared exponential covariance function $k(\mathbf{r}, \mathbf{r}') = \sigma_f^2 e^{-\frac{1}{2l^2}\|\mathbf{r}-\mathbf{r}'\|^2}$ for the scalar potential, we obtain

$$\frac{\partial^2}{\partial r_i \partial x_j'} k(\mathbf{r}, \mathbf{r}') = l^{-2}(\delta_{ij} - l^{-2}(r_i - r_i')(r_j - r_j'))k(\mathbf{r}, \mathbf{r}'),$$

which in vector notation gives

$$K_{\mathbf{H}}(\mathbf{r}, \mathbf{r}') = \frac{1}{l^2}\left(I_3 - \left(\frac{\mathbf{r}-\mathbf{r}'}{l}\right)\left(\frac{\mathbf{r}-\mathbf{r}'}{l}\right)^{\mathsf{T}}\right)\sigma_f^2 e^{-\frac{1}{2l^2}\|\mathbf{r}-\mathbf{r}'\|^2}.$$

A.2 Magnetic Vector Potential

Based on identity (3.18b) we know that a divergence-free vector field $\mathbf{B}$

$$\nabla \cdot \mathbf{B} = 0$$

can be written as the curl of a vector potential, such that

$$\mathbf{B} = \nabla \times \mathbf{A} \qquad \Leftrightarrow \qquad B_i = \sum_{j,k} \epsilon_{ijk} \frac{\partial}{\partial r_j} \mathbf{A}_k(\mathbf{r}),$$

where ϵ_{ijk} is the Levi-Civita symbol defined in (2.46). By modeling the vector potential with a GP

$$\mathbf{A}(\mathbf{r}) \sim \mathcal{GP}(0, K(\mathbf{r}, \mathbf{r}')),$$

we have that also $\mathbf{B}(\mathbf{r})$ is a GP

$$\mathbf{B}(\mathbf{r}) \sim \mathcal{GP}(0, K_{\mathbf{B}}(\mathbf{r}, \mathbf{r}')),$$

for which each component of the corresponding covariance function is given by

$$\begin{aligned}
[K_{\mathbf{B}}]_{(ij)}(\mathbf{r}, \mathbf{r}') &= \operatorname{Cov}\left[B_i(\mathbf{r}),\ B_j(\mathbf{r}')\right] \\
&= \operatorname{Cov}\left[\sum_{k,l} \epsilon_{ikl} \frac{\partial}{\partial r_k} \mathbf{A}_l(\mathbf{r}),\ \sum_{m,n} \epsilon_{jmn} \frac{\partial}{\partial r_m} \mathbf{A}_n(\mathbf{r}')\right] \\
&= \sum_{k,l,m,n} \epsilon_{ikl}\epsilon_{jmn} \frac{\partial^2}{\partial r_k \partial x_m'} K_{ln}(\mathbf{r}, \mathbf{r}').
\end{aligned}$$

Further, by using diagonal covariance function $K_{ln}(\mathbf{r}, \mathbf{r}') = \delta_{ln} k(\mathbf{r}, \mathbf{r}')$, we have that

$$\begin{aligned}
(K_{\mathbf{B}})_{(ij)}(\mathbf{r}, \mathbf{r}') &= \sum_{k,l,m,n} \underbrace{\epsilon_{ikl}\epsilon_{jmn}\delta_{ln}}_{\delta_{ij}\delta_{km} - \delta_{im}\delta_{kj}} \frac{\partial^2}{\partial r_k \partial x_m'} k(\mathbf{r}, \mathbf{r}') \\
&= \sum_{k,m} (\delta_{ij}\delta_{km} - \delta_{im}\delta_{kj}) \frac{\partial^2}{\partial r_k \partial x_m'} k(\mathbf{r}, \mathbf{r}').
\end{aligned}$$

If we chose to model each diagonal component with a squared exponential $k(\mathbf{r}, \mathbf{r}') = \sigma_f^2 e^{-\frac{1}{2l^2}\|\mathbf{r}-\mathbf{r}'\|^2}$, we have

$$[K_{\mathbf{B}}]_{(ij)}(\mathbf{r}, \mathbf{r}') = \sum_{k,m} (\delta_{ij}\delta_{km} - \delta_{im}\delta_{kj}) l^{-2} (\delta_{km} - l^{-2}(r_k - r'_k)(r_m - r'_m)) k(\mathbf{r}, \mathbf{r}')$$

and with

$$\begin{aligned}
\sum_{k,m} \delta_{ij}\delta_{km}\delta_{km} &= 3\delta_{ij}, \\
\sum_{k,m} \delta_{im}\delta_{kj}\delta_{km} &= \delta_{ij}, \\
\sum_{k,m} \delta_{ij}\delta_{km}(r_k - r'_k)(r_m - r'_m) &= \sum_{k} \delta_{ij}(r_k - r'_k)^2, \\
\sum_{k,m} \delta_{im}\delta_{kj}(r_k - r'_k)(r_m - r'_m) &= (r_i - r'_i)(r_j - r'_j),
\end{aligned}$$

we can further simplify the expression to

$$(K_{\mathbf{B}})_{(ij)}(\mathbf{r}, \mathbf{r}') = l^{-2}\left(2\delta_{ij} - l^{-2}\delta_{ij}\left(\sum_k (r_k - r'_k)^2\right) + l^{-2}(r_i - r'_i)(r_j - r'_j)\right) k(\mathbf{r}, \mathbf{r}'),$$

which in vector notation gives

$$K_{\mathbf{B}}(\mathbf{r}, \mathbf{r}') = \frac{1}{l^2}\left(\left(2 - \left\|\frac{\mathbf{r}-\mathbf{r}'}{l}\right\|^2\right) I_3 + \left(\frac{\mathbf{r}-\mathbf{r}'}{l}\right)\left(\frac{\mathbf{r}-\mathbf{r}'}{l}\right)^{\mathsf{T}}\right) \sigma_f^2 e^{-\frac{1}{2l^2}\|\mathbf{r}-\mathbf{r}'\|^2}.$$

B

Derivation of the Magnetic Dipole Model

This appendix derives the equation for the induced magnetic fields caused by a magnetic dipole, which was introduced in Section 3.3.

Consider a region V with localized current density. That means, charged particles can move within the region, but neither leave it nor be added to it. According to (3.4), this current density gives rise to an induced magnetic field outside the region (see Figure 3.1).

From Section 3.6.2, we know that the the magnetic field $\mathbf{B}$ is the curl of its vector potential $\mathbf{A}(\mathbf{r})$

$$\mathbf{B}(\mathbf{r}) = \nabla \times \mathbf{A}(\mathbf{r}). \tag{B.1}$$

Following (3.26), this vector field obeys the following Poisson equation

$$\nabla^2 \mathbf{A} = -\mu_0 \mathbf{J}, \tag{B.2}$$

which has the solution

$$\mathbf{A}(\mathbf{r}) = \frac{\mu_0}{4\pi} \int \frac{\mathbf{J}(\mathbf{r}')}{\|\mathbf{r} - \mathbf{r}'\|} d^3 r'. \tag{B.3}$$

Since $\mathbf{J}(\mathbf{r}') \neq \mathbf{0}$ only in the region of the localized current density V, we have

$$\mathbf{A}(\mathbf{r}) = \frac{\mu_0}{4\pi} \int_V \frac{\mathbf{J}(\mathbf{r}')}{\|\mathbf{r} - \mathbf{r}'\|} d^3 r'. \tag{B.4}$$

Furthermore, the denominator in (B.4) can be expanded in powers of $\mathbf{r}'$. With $\|\mathbf{r}\| > \|\mathbf{r}\|'$ this will be

$$\frac{1}{\|\mathbf{r} - \mathbf{r}'\|} = \frac{1}{\|\mathbf{r}\|} + \nabla\left(\frac{1}{\|\mathbf{r}\|}\right) \cdot (-\mathbf{r}') + \cdots = \frac{1}{\|\mathbf{r}\|} + \frac{\mathbf{r} \cdot \mathbf{r}'}{\|\mathbf{r}\|^3} + \ldots, \tag{B.5}$$

where $r = \|\mathbf{r}\|$. If this Taylor expansion is used by (B.3), this results in

$$A_i(\mathbf{r}) = \frac{\mu_0}{4\pi}\left(\frac{1}{\|\mathbf{r}\|}\int_V J_i(\mathbf{r}')d^3r' + \frac{\mathbf{r}}{\|\mathbf{r}\|^3}\cdot\int_V J_i(\mathbf{r}')\mathbf{r}'d^3r' + \dots\right). \tag{B.6}$$

Due to the fact that the current density $\mathbf{J}(\mathbf{r})$ is localized and obeys the static continuity condition $\nabla\cdot\mathbf{J} = 0$, Gauss' theorem makes the first term in (B.6) zero. Furthermore, it can be shown that

$$\mathbf{r}\cdot\int_V \mathbf{r}'J_i(\mathbf{r})d^3r' = (\mathbf{m}\times\mathbf{r})_i, \tag{B.7}$$

where $\mathbf{m}$ is the *magnetic dipole moment*

$$\mathbf{m} = \frac{1}{2}\int_V \mathbf{r}'\times\mathbf{J}(\mathbf{r}')d^3r'. \tag{B.8}$$

The details of these steps are clearly outlined in Jackson (1998). By truncating (B.6) and using (B.8) in (B.6), we obtain

$$\mathbf{A}(\mathbf{r}) = \frac{\mu_0}{4\pi}\frac{\mathbf{m}\times\mathbf{r}}{\|\mathbf{r}\|^3}. \tag{B.9}$$

The induced magnetic field can be calculated directly by evaluating the curl of (B.9),

$$\mathbf{B}(\mathbf{r}) = \frac{\mu_0}{4\pi}\frac{3(\mathbf{r}\cdot\mathbf{m})\mathbf{r} - \|\mathbf{r}\|^2\mathbf{m}}{\|\mathbf{r}\|^5}. \tag{B.10}$$

As long as $\|\mathbf{r}\| \gg \|\mathbf{r}'\|$, the truncation of the Taylor expansion in (B.5) is a good approximation. In other words, if the characteristic size of region V is small in comparison to the distance from the region to the observer at point P, see Figure 3.1, the dipole model is a valid model for the magnetic target.

Bibliography

S. Åkesson, R. Klaassen, J. Holmgren, J. W. Fox, and A. Hedenström. Migration Route and Strategies in a Highly Aerial Migrant, the Common Swift *Apus apus*, Revealed by Light-Level Geolocators. *PloS ONE:e41195*, 7(7), 2012.

E. Almqvist, D. Eriksson, A. Lundberg, E. Nilsson, N. Wahlström, E. Frisk, and M. Krysander. Solving the ADAPT benchmark problem - A student project study. In *21st International Workshop on Principles of Diagnosis (DX-10)*, Portland, Oregon, USA, October 2010.

M. A. Álvarez, L. Rosasco, and N. D. Lawrence. Kernels for vector-valued functions: A review. *Foundations and Trends in Machine Learning*, 4(3):195–266, March 2012.

J.-A. M. Assael, N. Wahlström, T. B. Schön, and M. P. Deisenroth. Data-efficient learning of feedback policies from image pixels using deep dynamical models. In *Deep Reinforcement Learning Workshop at the Annual Conference on Neural Information Processing Systems (NIPS)*, Montréal Canada, December 2015. Accepted.

P. Axelsson and F. Gustafsson. Discrete-time solutions to the continuous-time differential Lyapunov equation with applications to Kalman filtering. *IEEE Transactions on Automatic Control*, 60(3):632–643, 2015.

C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

G. E. P. Box. Robustness in the strategy of scientific model building. In R. L. Launer and G. N. Wilkinson, editors, *Robustness in statistics*, pages 201–236. Academic Press, New York, 1979.

F. Ceragioli, G. Lindmark, C. Veibäck, N. Wahlström, M. Lindfors, and C. Altafini. A bounded confidence model that preserves the signs of the opinion. In *European Control Conference*, 2015. Submitted.

D. K. Cheng. *Field and wave electromagnetics*. Reading, Massachusetts : Addison Wesley, 2 edition, 1989.

S.-Y. Cheung and P. Varaiya. Traffic surveillance by wireless sensor networks: Final report. Technical report, Traffic surveillance, University of Califonia, Berkeley, 2007.

T. R. Clem. Sensor technologies for hunting buried sea mines. In *OCEANS '02 MTS/IEEE*, volume 1, pages 452–460, Biloxi, MI, USA, October 2002.

L. Correia and D. Jonsson. A Real-time Watercolor Simulation for a Novel Tracking Device. Master's thesis, Linköping University, Department of Science and Technology, 2012.

G. Deak, K. Curran, and J. Condell. A survey of active and passive indoor localisation systems. *Computer Communications*, 35(16):1939 – 1954, 2012.

V. Deleskog, H. Habberstad, G. Hendeby, D. Lindgren, and N. Wahlström. Robust NLS sensor localization using MDS initialization. In *Proceedings of 17th International Conference on Information Fusion (FUSION)*, Madrid, Spain, July 2014.

B. Di Bartolo. *Classical theory of electromagnetism*. World Scientific Publishing Company Incorporated, 2004.

P. A. Ekstrom. An advance in geolocation by light. *Memoirs of the National Institute of Polar Research (Special Issue)*, 58:210–226, 2004.

I. Forsman, J. Nordin, J. Bosson, and O. Grahn. DooVR - Interaktiv modellering i virtuell verklighet - Projektrapport, TNM094, 2015. Bachelor's project report.

R. Garnett. Lecture 11: Bayesian quadrature. University Lecture, 2015. URL `www.cse.wustl.edu/~garnett/cse515t/files/lecture_notes/11.pdf`. Accessed: 2015-09-30.

K. Granström and U. Orguner. A PHD filter for tracking multiple extended targets using random matrices. *IEEE Transactions on Signal Processing*, 60(11): 5657–5671, November 2012.

K. Granström, C. Lundquist, and U. Orguner. Extended target tracking using a Gaussian-mixture PHD filter. *IEEE Transactions on Aerospace and Electronic Systems*, 48(4):3268–3286, October 2012.

F. Gustafsson and N. Wahlström. Method and device for pose tracking using vector magnetometers, 2012. Patent. Under revision.

J. Hartikainen and S. Särkkä. Kalman filtering and smoothing solutions to temporal Gaussian process regression models. In *Proceedings of the International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 379–384, Kittila, Finland, August 2010.

G. Hendeby, F. Gustafsson, and N. Wahlström. Teaching Sensor Fusion and Kalman Filtering using a Smartphone. In *Proceedings of the The 19th World Congress of the International Federation of Automatic Control (IFAC)*, pages 10586–10591, Cape Town, South Africa, August 2014.

P. Hennig and M. Kiefel. Quasi-newton methods: A new direction. *The Journal of Machine Learning Research*, 14(1):843–865, 2013.

R. D. Hill. Theory of geolocation by light levels. In *Elephant seals: population ecology, behavior, and physiology. University of California Press, Berkeley*, pages 227–236. University of California Press, 1994.

J. Ho, S. M. Ahlers, C. Stratman, O. Aridor, L. Pantanowitz, J. L. Fine, J. A. Kuzmishin, M. C. Montalto, and A. V. Parwani. Can digital pathology result in cost savings? A financial projection for digital pathology implementation at a large integrated health care organization. *Journal of Pathology Informatics*, 5 (33), 2014.

T. Holsclaw, B. Sanso, H. K. H. Lee, K. Heitmann, S. Habib, D. Higdon, and U. Alam. Gaussian process modeling of derivative curves. *Technometrics*, 55 (1):57–67, 2013.

R. Hostettler, M. Lundberg, and W. Birk. A system identification approach to modeling of wave propagation in pavements. In *16th IFAC Symposium on System Identification*, Brussels, Belgium, July 2012.

J. D. Jackson. *Classical Electrodynamics*. John Wiley & Sons, Inc, 3rd edition, 1998.

A. H. Jazwinski. *Stochastic Processes and Filtering Theory*, volume 64 of *Mathematics in Science and Engineering*. Academic Press, New York, NY, USA, 1970.

T. Kailath, A. H. Sayed, and B. Hassibi. *Linear estimation*. Prentice Hall Upper Saddle River, NJ, 2000.

M. Kok, N. Wahlström, T. B. Schön, and F. Gustafsson. MEMS-based inertial navigation based on a magnetic field map. In *Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 6466–6470, Vancouver, Canada, May 2013.

E. Le Grand and S. Thrun. 3-axis magnetic field mapping and fusion for indoor localization. In *Proceedings of IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 358–364, Hamburg, Germany, 2012.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems*, pages 396–404. Morgan Kaufmann, 1990.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–444, 2015.

J. Lenz and S. Edelstein. Magnetic sensors and their applications. *Sensors Journal, IEEE*, 6(3):631–649, June 2006.

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

J. C. Maxwell. A dynamical theory of the electromagnetic field. *Philosophical Transactions of the Royal Society of London*, 155:459–513, 1865.

W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, and et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

B. Öksendal. *Stochastic differential equations*. Springer, 6th edition, 2007.

M. Osborne. *Bayesian Gaussian processes for sequential prediction, optimisation and quadrature*. PhD thesis, University of Oxford, UK, 2010.

A. Papoulis and S. U. Pillai. *Probability, random variables, and stochastic processes*. McGraw-Hill Education, New York, 1991.

C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.

H. Singer, L. Matheson, R. Grubb, A. Newman, and D. Bouwer. Monitoring space weather with the GOES magnetometers. *SPIE Proceesings*, 2812:299–308, 1996.

E. Solak, R. Murray-Smith, W. E. Leithead, D. J. Leith, and C. E. Rasmussen. Derivative observations in Gaussian process models of dynamic systems. In *Advances in Neural Information Processing Systems*, pages 1033–1040, 2003.

A. Solin, M. Kok, N. Wahlström, T. B. Schön, and S. Särkkä. Modeling and interpolation of the ambient magnetic field by Gaussian processes. *Pre-print arXiv:1509.04634*, September 2015.

B. J. M. Stutchbury, S. A. Tarof, T. Done, E. Gow, P. M. Kramer, J. Tautin, J. W. Fox, and V. Afanasyev. Tracking long-distance songbird migration by using geolocators. *Science*, 323(5916):896, 2009.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. The MIT Press, 2005.

I. Vallivaara, J. Haverinen, A. Kemppainen, and J. Roning. Magnetic field-based SLAM method for solving the localization problem in mobile robot floor-cleaning task. In *Proceedings of the 15th International Conference on Advanced Robotics (ICAR)*, pages 198–203, Tallin, Estonia, June 2011.

P. Van Overschee and B. De Moor. *Subspace identification for linear systems - theory, implementation, applications*. Kluwer Academic Publishers, 1996.

D. Vissière, A. P. Martin, and N. Petit. Using magnetic disturbances to improve IMU-based position estimation. In *Proceedings of the European Control Conference*, pages 2853–2858, Kos, Greece, July 2007.

N. Wahlström. Target tracking using Maxwell's equations. Master's thesis, Linköping University, Automatic Control, June 2010.

N. Wahlström. *Localization using Magnetometers and Light Sensors*. Licentiate's thesis, Linköping University, Automatic Control, March 2013.

N. Wahlström and F. Gustafsson. Magnetometer modeling and validation for tracking metallic targets. *IEEE Transactions on Signal Processing*, 62(3):545–556, 2014.

N. Wahlström and F. Gustafsson. Tracking position and orientation of magnetic objects using magnetometer networks. *IEEE Transactions on Signal Processing*, 2015. Submitted.

N. Wahlström and E. Özkan. Extended target tracking using Gaussian processes. *IEEE Transactions on Signal Processing*, 63(16):4165–4178, 2015.

N. Wahlström, J. Callmer, and F. Gustafsson. Magnetometers for tracking metallic targets. In *Proceedings of 13th International Conference on Information Fusion (FUSION)*, Edinburgh, Scotland, July 2010.

N. Wahlström, J. Callmer, and F. Gustafsson. Single target tracking using vector magnetometers. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4332–4335, Prague, Czech Republic, May 2011.

N. Wahlström, F. Gustafsson, and S. Åkesson. A Voyage to Africa by Mr Swift. In *Proceedings of the 15th International Conference on Information Fusion (FUSION)*, pages 808–815, Singapore, July 2012a.

N. Wahlström, R. Hostettler, F. Gustafsson, and W. Birk. Rapid classification of vehicle heading direction with two-axis magnetometer. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3385–3388, Kyoto, Japan, March 2012b.

N. Wahlström, M. Kok, T. B. Schön, and F. Gustafsson. Modeling magnetic fields using Gaussian processes. In *Proceedings of the the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 3522–3526, Vancouver, Canada, May 2013.

N. Wahlström, P. Axelsson, and F. Gustafsson. Discretizing stochastic dynamical systems using Lyapunov equations. In *Proceedings of the The 19th World Congress of the International Federation of Automatic Control (IFAC)*, pages 3726–3731, Cape Town, South Africa, August 2014.

N. Wahlström, R. Hostettler, F. Gustafsson, and W. Birk. Classification of driving direction in traffic surveillance using magnetometers. *IEEE Transactions on Intelligent Transportation Systems*, 15(4):1405–1418, 2014.

N. Wahlström, T. B. Schön, and M. P. Deisenroth. Learning deep dynamical models from image pixels. In *Proceedings of the 17th IFAC Symposium on System Identification (SYSID)*, Bejing, China, October 2015a.

N. Wahlström, T. B. Schön, and M. P. Deisenroth. From pixels to torques: Policy learning with deep dynamical models. In *Deep Learning Workshop at the International Conference on Machine Learning (ICML)*, Lille, France, July 2015b.

K. Xu, J. Ba, R. Kiros, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the International Conference on Learning representations*, 2015. URL `http://arxiv.org/abs/1502.03044`.

H. Zhang and F. Martin. Robotic mapping assisted by local magnetic field anomalies. In *Proceedings of the IEEE Conference on Technologies for Practical Robot Applications (TePRA)*, pages 25–30, Woburn, USA, April 2011.

Part II

Publications

Paper A

Tracking Position and Orientation of Magnetic Objects Using Magnetometer Networks

Authors: Niklas Wahlström, and Fredrik Gustafsson

Edited version of the paper:

N. Wahlström and F. Gustafsson. Tracking position and orientation of magnetic objects using magnetometer networks. *IEEE Transactions on Signal Processing*, 2015. Submitted.

Tracking Position and Orientation of Magnetic Objects Using Magnetometer Networks

Niklas Wahlström, and Fredrik Gustafsson

Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
{nikwa,fredrik}@isy.liu.se

Abstract

A framework for estimation and filtering of magnetic dipoles in a network of magnetometers is presented. The application in mind is tracking of objects consisting of permanent magnets for controlling computer applications, though the framework can also be applied to tracking larger objects such as vehicles. A general sensor model for the network is presented for tracking objects consisting of (i) a single dipole, (ii) a structure of dipoles and (iii) several freely moving (structures of) dipoles, respectively. A single dipole generates a magnetic field with rotation symmetry, so at best five degrees of freedom (5D) tracking can be achieved, where the SNR decays cubically with distance. One contribution is the use of structures of dipoles, which allows for full 6D tracking if the dipole structure is large enough. An observability analysis shows that the sixth degree of freedom is weakly observable, where the SNR decays to the power of four with distance, and that there is a 180 degree ambiguity around a specific symmetry axis. Experimental results are presented and compared to a reference tracking system, and four public demonstrators based on this framework are briefly described.

1 Introduction

Magnetic localization offers many advantages in comparison to other localization techniques in terms accuracy, robustness, and cost. The technique has for the last decade primarily been used in applications to track medical apparatus inside the human body (Stathopoulos et al., 2005; Guignet et al., 2006; Wang et al., 2006), in traffic surveillance to detect and localize ground and maritime vehicles (Birsan, 2005; Kozick and Sadler, 2007a; Wahlström and Gustafsson, 2014), and more lately in in computer gaming industry (Harrison and Hudson, 2009; Ashbrook et al., 2011; Chan et al., 2013; Liang et al., 2012). Magnetic localization

can provide accurate position and orientation estimates in applications for which occlusion disqualifies vision sensors, and lack of absolute position excludes pure IMU-based approaches. Further the magnetic field does not suffer from multi-path and scattering effects as in radio-based solutions.

This paper considers a sensor network of vector magnetometers that is used to determine the position and orientation of a permanently magnetized object, for example a magnet. We suggest a filtering framework including measurement and motion models that can be used to determine both the position and the orientation of the magnet based on the magnetometer measurements. We also present a range of applications in which this technique have been successfully implemented. This extends the palette of applications mentioned above. These applications have been realized with low-graded magnetometers (a few dollars each) that can be found in standard smartphones.

The use of stationary magnetometers to determine position and orientation is proposed in several publications (Haynor et al., 2001; Beck, 2001; Yabukami et al., 2000; Schlageter et al., 2001). They model the object as a magnetic dipole moment inducing a magnetic dipole field that can be computed analytically. This magnetic dipole moment includes both position and orientation of that object as well as its magnetic strength. This initial work has been further elaborated by Yabukami et al. (2002, 2003), Hu et al. (2005, 2006, 2008, 2010) and researchers at Institute of Physiology, University of Lausanne (Stathopoulos et al., 2005; Guignet et al., 2006). Their primary application has been to record images of the digestive track, so-called capsule endoscopy, where a capsule equipped with a small magnet can be localized during its path through the tract. All of these contributions have in common that they compute the position and orientation of the object at each time instant separately (typically using Levenberg-Marquardt algorithm). Sherman et al. (2007) demonstrate that these tracking systems manage to obtain a precision of 1 mm for position and 1° in orientation within a range of 100 mm, in that case using a network of 27 scalar magnetometers.

Birsan (2004, 2005); Kozick and Sadler (2007a,b, 2008) have used magnetometers to localize ground vehicles, the latter also fusing magnetometers with acoustic sensors.

This contribution presents a general framework for tracking one or several magnetic objects using a sensor network of magnetometers, extending the theory in the aforementioned references in several aspects:

1. A general sensor model that relates an object including at least one magnetic dipole and a general network of magnetometers, where the model also includes calibration parameters. Specific models for objects consisting of a single dipole (enabling 5D pose estimation), a structure of dipoles (for 6D pose), and several separate (structures of) dipoles (allowing multiple objects to be tracked). This is presented in Section 2. Together with these sensor models, different orientation representations are considered. They are described in Section 3.

2. The use of objects that consist of structures of dipoles is supposedly an original contribution. Therefore, an observability analysis is presented, based

on a Taylor expansion of the sensor model. The first term shows the intuitive result that if the object is far away from the sensor network, the structure will appear as one single dipole, and the orientation around the symmetry axis of a dipole equivalent will be lost. By a proper definition of the origin of the object, the symmetry axis can be defined easily. The second order Taylor term shows an interesting result in that that the sensor model is bimodal. A 180° flip of the object around the symmetry axis will give the same Taylor term. This analysis is presented in Section 4 with additional material in Appendix A.

3. A couple of motion models for temporal filtering. These models explore different ways to parametrize the orientation, suitable for tracking pose in 5D and 6D, respectively. These motion models are described in Section 5.

4. Section 6 presents performance results when tracking a single and double dipole, respectively, compared to a reference tracking system. Section 7 briefly describes four applications based on the filtering framework presented in this paper.

2 Sensor Model

In a statistical signal processing framework, measurements can be described with a stochastic state-space model

$$\mathbf{x}_{k+1} = F_k \mathbf{x}_k + G_k \mathbf{w}_k, \qquad \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, Q), \tag{1a}$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{e}_k, \qquad \mathbf{e}_k \sim \mathcal{N}(\mathbf{0}, R), \tag{1b}$$

where $\mathbf{y}_k$ is the measurement, $\mathbf{x}_k$ is the state of the system, $\mathbf{w}_k$ is the process noise and $\mathbf{e}_k$ is the measurement noise, all at time instant kT, T being the sample period. The process and measurement noises are assumed to be white Gaussian with $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, Q)$ and $\mathbf{e}_k \sim \mathcal{N}(\mathbf{0}, R)$. The motion model (1a) is explained in Section 5 and the sensor model (1b) is described in this section.

The measurement from a stationary vector-magnetometer $\tilde{\mathbf{y}}_k$ can be decomposed into a constant bias term $\mathbf{B}_0$ and a time varying term $\mathbf{h}(\mathbf{x}_k)$, where

$$\tilde{\mathbf{y}}_k = \mathbf{B}_0 + \mathbf{h}(\mathbf{x}_k) + \mathbf{e}_k. \tag{2}$$

The constant bias term is in turn composed of the earth magnetic field, magnetic distortions due the stationary magnetic environment, and sensor biases. This bias term can be measured during a calibration phase before any magnetic object enters the scene to then be subtracted from the measurements. This approach is advocated in this paper and we therefore only consider the calibrated measurement $\mathbf{y}_k = \tilde{\mathbf{y}}_k - \mathbf{B}_0$ onwards. Optionally, the bias could also be estimated jointly with the state $\mathbf{x}_k$ as a part of the filtering.

In this work, the time varying term represents magnetic distortions from moving objects that consist of permanent magnets. These objects are modeled with

magnetic dipoles, where each dipole corresponds to a magnet. We present a single dipole model in Section 2.1, continue by extending this to a multi-dipole model in Section 2.2, and finally a multi-object model in Section 2.3. In the model description, two coordinates systems are considered; (i) the global coordinate system, which is the coordinate system of the stationary sensor network, and (ii) the local coordinate system, which is the body-fixed coordinate system of the moving object. For the quantities that are presented in this paper, $\mathbf{s}_l$, $\mathbf{b}_l$ and U are defined in local coordinates, whereas $\mathbf{y}_k$, $\mathbf{r}_k$, $\mathbf{m}_k$, $\mathbf{q}_k$, R_k, $\boldsymbol{\theta}_j$, $\mathbf{v}_k$, $\boldsymbol{\omega}_k$ and W_k are given in global coordinates.

2.1 Single Dipole Model

A magnetic dipole produces a magnetic dipole field that can be derived from Maxwell's equations. This field decays cubically with the distance to the dipole. With J magnetometers positioned at $\{\boldsymbol{\theta}_j\}_{j=1}^{J}$, we obtain the following sensor model for the jth sensor

$$\mathbf{h}_j(\mathbf{x}_k) = J(\mathbf{r}_k - \boldsymbol{\theta}_j)\mathbf{m}_k, \qquad \text{where} \qquad J(\mathbf{r}) = \frac{1}{\|\mathbf{r}\|^5}(3\mathbf{r}\mathbf{r}^\mathsf{T} - \|\mathbf{r}\|^2 I_3), \tag{3}$$

where $\mathbf{m}_k$ is the magnetic dipole moment of the magnet, $\mathbf{r}_k = [r_k^{(\mathrm{x})}, r_k^{(\mathrm{y})}, r_k^{(\mathrm{z})}]$ is the position in Cartesian coordinates, both given in global coordinates, and $\mathbf{x}_k$ is the state of the system.

The magnetic dipole moment $\mathbf{m}_k \in \mathbb{R}^3$ has both a magnitude and a direction. The magnetic field produced is proportional to the magnitude $m = \|\mathbf{m}_k\|$ and the orientation of the object is represented by the direction of $\mathbf{m}_k$. In the single dipole model, we use $\mathbf{m}_k$ as a part of the state vector $\mathbf{x}_k$ to encode both the orientation and the strength of the magnet.

Note that this orientation representation only encodes two degrees of freedom (DoF) for the orientation. In fact, only two DoF are observable in (3) because the dipole field is symmetric around $\mathbf{m}_k$. By applying an addition rotation around the dipole moment $\mathbf{m}_k$, its direction does not change.

2.2 Multi-Dipole Model

The symmetry present in the single dipole model can be broken by considering objects with multiple dipoles. Therefore, to obtain all three DoF for the orientation, a multi-dipole object is considered. It consists of L magnetic dipole moments with a predefined and constant geometry. Since magnetic fields from multiple sources superpose, the sensor model is constructed by adding multiple dipole fields.

Consider the lth dipole moment to be located at position $\mathbf{s}_l$ relative to the center of the object with the direction $\mathbf{b}_l$, where $\|\mathbf{b}_l\| = 1$. Both $\mathbf{s}_l$ and $\mathbf{b}_l$ are provided in local coordinates. See Figure 1 for an example of such a geometry including two dipoles.

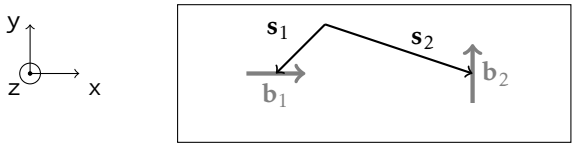


Figure 1: *The geometry of a multi-dipole object with two dipoles. The vectors* $\mathbf{s}_1$ *and* $\mathbf{s}_2$ *describe the position of the two dipole moments, and the vectors* $\mathbf{b}_1$ *and* $\mathbf{b}_2$ *their directions. The two magnets are in this example aligned perpendicular to each other with the first dipole pointing towards the second dipole. This geometry also illustrates the setup that was used in the experiment in Section 6.2.*

Each magnetic dipole moment $\mathbf{m}_{k,l}$ can then be expressed with its direction $\mathbf{b}_l$, its unknown magnitude m_l, and a rotation matrix R_k that relates the orientation of the object relative to the global coordinate frame

$$\mathbf{m}_{k,l} = m_l R_k \mathbf{b}_l. \tag{4}$$

Further, the position of dipole l is given by $\mathbf{r}_k + R_k \mathbf{s}_l$, where $R_k \mathbf{s}_l$ is the displacement of dipole l in global coordinates relative to the center of the object. With all combined, the sensor model is given by

$$\mathbf{h}_j(\mathbf{x}_k) = \sum_{l=1}^{L} J(\mathbf{r}_k + R_k \mathbf{s}_l - \boldsymbol{\theta}_j) m_l R_k \mathbf{b}_l, \tag{5}$$

where $\mathbf{r}_k$, R_k and $\{m_l\}_{l=1:L}$ are the unknown components. The parametrization of R_k is further discussed in Section 3.

2.3 Multi-Object Multi-Dipole Model

The multi-dipole model can be further extended to a model for multiple multi-dipole objects. As discussed above, the contributions from all objects superpose and the corresponding sensor model is given by

$$\mathbf{h}_j(\mathbf{x}_k) = \sum_{i=1}^{M} \sum_{l=1}^{L} J(\mathbf{r}_{k,i} + R_{k,i} \mathbf{s}_{l,i} - \boldsymbol{\theta}_j) m_{l,i} R_{k,i} \mathbf{b}_{l,i}, \tag{6}$$

where $\mathbf{r}_{k,i}$ and $R_{k,i}$ describe the position and the orientation of ith object, respectively.

Note that neither more objects nor extended objects do result in an increased number of measurements as otherwise common in multiple and extended target tracking (Koch, 2008). The number of measurements depend on the number of sensors deployed and not on the number of objects present in the scene. This makes it significantly more difficult to track multiple objects in comparison to a single object. For this reason, we only present experimental results for the single dipole model (3) and the multi-dipole model (5) in Section 6. The multi-object model is included to provide a general framework.

3 Orientation Representations

Whereas the position is parametrized with a three-dimensional Cartesian vector throughout this paper, different parametrizations for the orientation are considered. In this section, three different orientation representations are introduced and compared.

3.1 Magnetic Dipole Moment

In the single dipole model (3), the magnetic dipole moment $\mathbf{m}_k$ encodes both the orientation and the magnitude of the dipole. This can be described with a rotation matrix R_k giving

$$\mathbf{m}_k = mR_k\mathbf{b}, \tag{7}$$

where $\mathbf{b}$, with $\|\mathbf{b}\| = 1$, denotes the direction of the dipole in local coordinates.

As already discussed, this orientation representation only encodes two degrees of freedom (DoF) for the orientation. In the control literature, similar applications are referred to as pointing applications (Chaturvedi et al., 2011). In that context, a reduced representation of the rotation matrix can be expressed using the reduced-attitude vector $\boldsymbol{\gamma}_k = R_k\mathbf{b}$. In our context, $\boldsymbol{\gamma}_k$, with $\|\boldsymbol{\gamma}_k\| = 1$, describes the direction of the magnetic dipole vector in global coordinates and the magnetic dipole moment is expressed as

$$\mathbf{m}_k = m \cdot \boldsymbol{\gamma}_k. \tag{8}$$

Instead of using separate parameters for m and $\boldsymbol{\gamma}_k$, they can be combined into one "extended" reduced-attitude vector $\bar{\boldsymbol{\gamma}}_k = m \cdot \boldsymbol{\gamma}_k \in \mathbb{R}^3$. This parametrization is obviously equivalent of using the parameters in the magnetic dipole vector itself

$$\mathbf{m}_k = \bar{\boldsymbol{\gamma}}_k. \tag{9}$$

The assumption that $\|\mathbf{m}_k\|$ shall remain constant can be incorporated in the dynamical model, see Section 5.2.

3.2 Unit Quaternion

In (4), a rotation matrix was used to describe the full orientation state of the object. Since all three DoF are needed, the approach discussed in the Section 3.1 does not work. We instead use quaternions.

Unit quaternions $\mathbf{q} \in \mathbb{R}^4$, with $\|\mathbf{q}\| = 1$, are popular for parametrizing rotation matrices, see for example Kuipers (1999). This parametrization avoids singularities otherwise present while using Euler angles. The rotation matrix is given by $R_k = R(\mathbf{q}_k)$, where the elements of $R(\mathbf{q}_k)$ are quadratic expression in the elements of $\mathbf{q} = [q_0 \ q_1 \ q_2 \ q_3]^\mathsf{T}$, where

$$R(\mathbf{q}) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 + q_3^2 & 2q_1q_2 - 2q_0q_3 & 2q_1q_3 + 2q_0q_2 \\ 2q_1q_2 + 2q_0q_3 & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2q_2q_3 - 2q_0q_1 \\ 2q_1q_3 - 2q_0q_2 & 2q_2q_3 + 2q_0q_1 & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}. \tag{10}$$

The sensor model is then given by (5) with $R_k = R(\mathbf{q}_k)$, where $\mathbf{r}_k$, $\mathbf{q}_k$, and $\{m_1\}_{l=1:L}$ are contained in the state $\mathbf{x}_k$.

3.3 Extended Quaternion

By further assuming that all magnetic dipoles have the same unknown magnitude m, a compact representation of $\mathbf{q}_k$ and m can be achieved. This assumption can be relaxed by assuming that the relative magnitudes of the dipoles are known. These relative magnitudes can then be encoded in the magnitudes of $\mathbf{b}$. Note that this assumption is not too restrictive since magnets of use are normally of the same kind. Their relative magnitudes can be measured based on their relative size (or weight), or assumed to have the same magnitude if they are identical.

With this assumption, (4) can, in a similar manner as for the extended reduced-attitude vector described in Section 3.1, be rewritten as

$$\mathbf{m}_{k,l} = R(\bar{\mathbf{q}}_k)\mathbf{b}_l, \quad \text{where} \quad \bar{\mathbf{q}}_k = \sqrt{m}\mathbf{q}_k. \tag{11}$$

This defines an extended quaternion $\bar{\mathbf{q}}_k$ that not only encodes the orientation of the object, but also the magnitude of the magnetic dipoles. This combined with the sensor model (5) gives

$$\mathbf{h}_j(\mathbf{x}_k) = \sum_{l=1}^{L} J(\mathbf{r}_k + R_k\mathbf{s}_l - \boldsymbol{\theta}_j)\mathbf{m}_{k,l}, \quad \text{where} \quad R_k = R\left(\frac{\bar{\mathbf{q}}_k}{\|\bar{\mathbf{q}}_k\|}\right), \quad \mathbf{m}_{k,l} = R(\bar{\mathbf{q}}_k)\mathbf{b}_l, \tag{12}$$

where $\mathbf{r}_k$ and $\bar{\mathbf{q}}_k$ are components of the state $\mathbf{x}_k$.

Note that $\mathbf{s}_l$ needs to be rotated with a proper rotation matrix R_k because the dipole magnitude that is encoded in $\bar{\mathbf{q}}_k$ should not effect the term $R_k\mathbf{s}_l$. Therefore, a normalization of the extended quaternion $\bar{\mathbf{q}}_k$ is required in that term.

A similar representation can also be used for the multi-object multi-dipole model (6) where $\mathbf{r}_{k,i}$ and $\bar{\mathbf{q}}_{k,i}$ then describe the position and the orientation of the ith object.

3.4 Discussion and Comparison

Three different orientation representations were proposed in this section:

1. A magnetic dipole vector $\mathbf{m}_k \in \mathbb{R}^3$ that encodes (i) two DoF for orientation and (ii) the magnitude of the magnetic dipole moment. This parametrization only works for single dipole objects, see Section 3.1.

2. A unit quaternion $\mathbf{q}_k \in \mathbb{R}^4$ with $\|\mathbf{q}_k\| = 1$ that encodes three DoF for orientation. The magnitudes of the magnetic dipole moments are modeled with separate scalars $m_1, \dots m_L \in \mathbb{R}$. This parametrization works for arbitrary multi-dipole objects, see Section 3.2.

3. A (non-unit) quaternion $\bar{\mathbf{q}}_k \in \mathbb{R}^4$ that encodes (iii) three DoF for orientation and (iv) the common magnitude m of all magnetic dipole moments. This parametrization works for single dipole and multi-dipole objects where the relative magnitudes of the dipole moments are known, see Section 3.3.

For the single dipole model (3), parametrization 1 is preferred because it only encodes the two DoF that are observable in that model. That model is also linear in $\mathbf{m}_k$, which is not the case if parametrization 2 or 3 would have been used. For the multi-dipole model (5), parametrization 2 or 3 are needed since all three DoF must be encoded.

In filtering applications that use unit quaternions for orientation representation, the quaternion needs to be normalized after each measurement update (often in an ad-hoc manner). If we use the third orientation representation, that normalization step can be omitted. This is because the norm of the quaternion contains information (the magnitude of the dipoles) that should be preserved. Therefore, if the relative magnitudes of the dipole moments are known, parametrization 3 is preferred since (i) no normalization step is required and (ii) less states are needed.

Finally, note that all three parametrizations have in common that they do not contain any singularities. In addition, $\mathbf{m}_k$, $\mathbf{q}_k$, and $\bar{\mathbf{q}}_k$ all have in common that their magnitudes should remain constant. This information is encoded in the motion models presented in Section 5.

4 Analysis

The proposed possibility to use structures of dipoles opens up applications where full 6D tracking is possible. It is therefore important to understand the degree of observability of the sixth degree of freedom.

The observability of the single dipole model (3) has previously been discussed in Wahlström and Gustafsson (2014) with the conclusion that at least two three-axis magnetometers are required to obtain observability for both the position and the two-dimensional orientation.

To analyze the observability of the multi-dipole model (5), we do a Taylor series expansion of the multi-dipole model (5) where we obtain

$$\sum_{l=1}^{L} J(\mathbf{r}_k + R_k\mathbf{s}_l)R_k\mathbf{b}_l \approx \underbrace{J(\mathbf{r}_k)\mathbf{m}_k}_{\text{(13a)}} + \underbrace{\frac{3}{\|\mathbf{r}_k\|^5}\left(W_k - \left(\frac{5\mathbf{r}_k^\mathsf{T} W_k \mathbf{r}_k}{2\|\mathbf{r}_k\|^2} + \gamma\right) I\right)\mathbf{r}_k}_{\text{(13b)}}, \tag{13}$$

where

$$\mathbf{m}_k = R_k\mathbf{b}, \quad W_k = R_k U R_k^\mathsf{T}, \mathbf{b} = \sum_{l=1}^{L} \mathbf{b}_l, \quad U = \sum_{l=1}^{L} \mathbf{s}_l\mathbf{b}_l^\mathsf{T} + \mathbf{b}_l\mathbf{s}_l^\mathsf{T}, \quad \gamma = \sum_{l=1}^{L} (\mathbf{b}_l^\mathsf{T}\mathbf{s}_l). \tag{14}$$

The details are outlined in Appendix A. Based on this expansion, multiple properties of the multi-dipole model can be analyzed and explained.

The first term (13a) in the Taylor expansion is identical to the single dipole model (3), where the total dipole moment is the sum of all moments $\mathbf{m}_k = \sum_{l=1}^{L} \mathbf{m}_{k,l}$. On a distance far away from the object, this term dominates and the object appears as one single dipole. As for the single dipole model, only 2 DoF for the orientation are encoded in (13a). A rotation of the object around the magnetic dipole moment $\mathbf{m}_k$ is not observable.

The orientation around $\mathbf{m}_k$ is instead encoded in the higher order terms. Whereas the dipole term decays cubically $\|\mathbf{r}_k\|^{-3}$ with distance $\|\mathbf{r}_k\|$, the first higher order term (13b) decays to the power of four $\|\mathbf{r}_k\|^{-4}$. Therefore, the ability to estimate the orientation around $\mathbf{m}_k$ decays faster with the distance $\|\mathbf{r}_k\|$ than the ability to estimate the remaining part of the orientation state.

Further, according to Appendix A, the second term (13a) remains unchanged for any additional 180° rotation around $\mathbf{m}_k$. Consequently, the last orientation state is not only more difficult to estimate, it is even more difficult to distinguish its true orientation from an orientation rotated 180°. This results in a bimodality of the corresponding likelihood. Higher order terms in the Taylor expansion may resolve this ambiguity, which has been found to be the case in our experiments. However, the information of this ambiguity decays to the power of five with distance.

As one particular illustration, Figure 2 shows the negative log likelihood (NLL)

$$-\log p(\mathbf{y}_k|\mathbf{x}_k) \propto 1/2(\mathbf{y}_k - h(\mathbf{x}_k))^{\mathsf{T}} R^{-1}(\mathbf{y}_k - h(\mathbf{x}_k))$$

for a certain time instant k. The global minimum of the NLL is denoted with $\mathbf{x}_k^*$. Starting from that minimum, the angle β, corresponding to rotation around the vector $\mathbf{m}_k$, is adjusted and the NLL, $-\log p(\mathbf{y}_k|\mathbf{x}_k^*(\beta))$, is evaluated as a function of that angle. As a comparison, the same procedure is performed for the other two angles corresponding to rotations around the two axis orthogonal to $\mathbf{m}_k$.

As shown Figure 2, the likelihood exhibits a clear bimodality for the angle β, with two minima approximately 180° apart. This is in accordance with the theoretical analysis presented above and in Appendix A. In addition, the variations of the NLL is smaller when β is varied compared to the other two angles α_1 and α_2. This is also supported by the theoretical analysis in that rotation around β is harder to estimate than rotations around the other to axis orthogonal to $\mathbf{m}_k$.

5 Motion Model

The motion model (1a) describes the dynamics of the state $\mathbf{x}_k$. It is described with a linear state-space model

$$\mathbf{x}_{k+1} = F_k\mathbf{x}_k + G_k\mathbf{w}_k, \qquad \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, Q), \tag{15}$$

as commonly performed in target tracking applications. The state $\mathbf{x}_k$ is divided into position state $\mathbf{x}_k^{\text{pos}}$ and orientation state $\mathbf{x}_k^{\text{ori}}$. The position state includes position and velocity $\mathbf{x}_k^{\text{pos}} = \begin{bmatrix}\mathbf{r}_k^{\mathsf{T}}, & \mathbf{v}_k^{\mathsf{T}}\end{bmatrix}^{\mathsf{T}}$, whereas the orientation state includes orientation and angular velocity $\boldsymbol{\omega}_k$ denoted by $\mathbf{x}_k^{\text{ori}} = \begin{bmatrix}\mathbf{m}_k^{\mathsf{T}}, & \boldsymbol{\omega}_k^{\mathsf{T}}\end{bmatrix}^{\mathsf{T}}$ or $\mathbf{x}_k^{\text{ori}} = \begin{bmatrix}\bar{\mathbf{q}}_k^{\mathsf{T}}, & \boldsymbol{\omega}_k^{\mathsf{T}}\end{bmatrix}^{\mathsf{T}}$

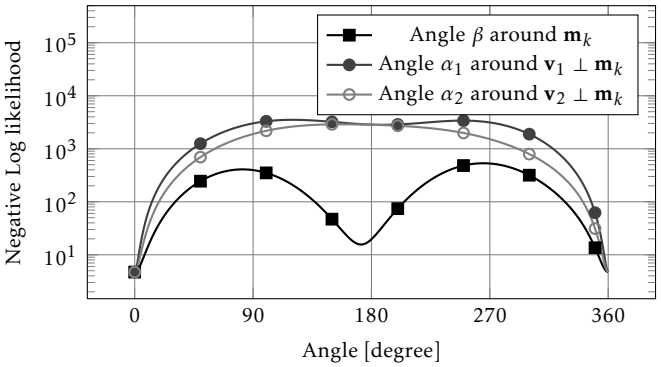


Figure 2: *Negative log likelihood (NLL) $-\log p(\mathbf{y}_k|\mathbf{x}_k^*)$ as a function of the three angles β, α_1 and α_2. The graph marked with squares is the NLL as a function of the angle β around the magnetic dipole moment $\mathbf{m}_k$. The two angles α_1 and α_2 are rotations around the other two axis $\mathbf{v}_1, \mathbf{v}_2$ orthogonal to $\mathbf{m}_k$, in the graph marked with circles. The angles $\beta = 0$, $\alpha_1 = 0$, and $\alpha_2 = 0$ correspond to the global minimum of the NLL. From this global minimum, one angle is adjusted at a time.*

depending on which of the two orientation representations that we use. The state-space is decomposed in a similar manner where

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_k^{\text{pos}} \\ \mathbf{x}_k^{\text{ori}} \end{bmatrix}, \; F_k = \begin{bmatrix} F^{\text{pos}} & 0_3 \\ 0_3 & F_k^{\text{ori}} \end{bmatrix}, \; G = \begin{bmatrix} G^{\text{pos}} & 0_3 \\ 0_3 & G_k^{\text{ori}} \end{bmatrix}, \; Q = \begin{bmatrix} Q^{\text{pos}} & 0_3 \\ 0_3 & Q^{\text{ori}} \end{bmatrix}, \; \mathbf{w}_k = \begin{bmatrix} \mathbf{w}_k^{\text{pos}} \\ \mathbf{w}_k^{\text{ori}} \end{bmatrix}.$$

We describe the motion model for $\mathbf{x}^{\text{pos}}$ in Section 5.1, and the motion model for the two representations of the orientation state in Section 5.2 and 5.3. We use a constant velocity model for both position and orientation that we derive from the continuous-time model.

5.1 Position State

To derive constant velocity motion model for the position, we model the velocity $\mathbf{v}$ to be a Brownian motion. This can formally be written as

$$\frac{d}{dt}\mathbf{v}(t) = \mathbf{w}^{\text{pos}}(t), \tag{16}$$

where $\mathbf{w}(t)$ is white Gaussian noise.[1] Together with $\frac{d}{dt}\mathbf{r}(t) = \mathbf{v}(t)$ this gives the joint model

$$\frac{d}{dt}\begin{bmatrix} \mathbf{r}(t) \\ \mathbf{v}(t) \end{bmatrix} = \begin{bmatrix} 0_3 & I_3 \\ 0_3 & 0_3 \end{bmatrix}\begin{bmatrix} \mathbf{r}(t) \\ \mathbf{v}(t) \end{bmatrix} + \begin{bmatrix} 0_3 \\ I_3 \end{bmatrix}\mathbf{w}(t), \qquad \mathbb{E}[\mathbf{w}(t)\mathbf{w}(\tau)^{\mathsf{T}}] = \sigma_{\text{pos}}^2 I_3 \delta(t-\tau), \tag{17}$$

[1] Note that white continuous-time Gaussian noise does not exist in practice, for example, because it has infinite energy. However, we can still think of it as a $\mathbf{w}$ as being driven by white noise. See Jazwinski (1970) for further details.

where $\mathbf{v}(t)$ is the velocity provided in global coordinates. This model can be derived from Newton's second law of motion, where the acceleration is interpreted as process noise, see for example Li and Jilkov (2003) for more details. By following Jazwinski (1970), we can derive a discrete-time constant velocity model for the position

$$\mathbf{x}_{k+1}^{\text{pos}} = F^{\text{pos}}\mathbf{x}_k^{\text{pos}} + G^{\text{pos}}\mathbf{w}_k^{\text{pos}}, \quad \mathbf{w}_k^{\text{pos}} \sim \mathcal{N}(\mathbf{0}, Q^{\text{pos}}), \tag{18}$$

where

$$F^{\text{pos}} = \begin{bmatrix} I_3 & TI_3 \\ 0_3 & I_3 \end{bmatrix}, \quad G^{\text{pos}} = \begin{bmatrix} TI_3 & 0_3 \\ 0_3 & I_3 \end{bmatrix}, \quad Q^{\text{pos}} = \frac{\sigma_{\text{pos}}^2}{T}\begin{bmatrix} \frac{1}{3}I_3 & \frac{1}{2}I_3 \\ \frac{1}{2}I_3 & I_3 \end{bmatrix}.$$

The derivation is also provided in, for example, Grewal and Andrews (2008).

5.2 Orientation State (Magnetic Dipole Moment)

In the first orientation representation (see Section 3.1), the orientation is encoded by the magnetic dipole moment $\mathbf{m}$. To derive the motion model for $\mathbf{m}$, we consider its angular velocity $\boldsymbol{\omega}$ provided in global coordinates. A vector $\mathbf{m}$ that rotates around an axis $\boldsymbol{\omega}/\|\boldsymbol{\omega}\|$ with the angular speed $\|\boldsymbol{\omega}\|$ obeys the following relation

$$\frac{d}{dt}\mathbf{m}(t) = \boldsymbol{\omega}(t) \times \mathbf{m}(t) \tag{19a}$$

$$= -C(\boldsymbol{\omega}(t))\mathbf{m}(t) = C(\mathbf{m}(t))\boldsymbol{\omega}(t), \tag{19b}$$

where $\times$ denotes the cross product and where

$$C(\mathbf{m}) = \begin{bmatrix} 0 & m_z & -m_y \\ -m_z & 0 & m_x \\ m_y & -m_x & 0 \end{bmatrix}. \tag{20}$$

In the same manner as in the previous section, we model the angular velocity $\boldsymbol{\omega}$ to be a Brownian motion

$$\frac{d}{dt}\boldsymbol{\omega}(t) = \mathbf{w}(t), \tag{21}$$

where $\mathbf{w}(t)$ is white Gaussian noise. This gives the joint model

$$\frac{d}{dt}\begin{bmatrix} \mathbf{m}(t) \\ \boldsymbol{\omega}(t) \end{bmatrix} = \begin{bmatrix} 0_3 & C(\mathbf{m}(t)) \\ 0_3 & 0_3 \end{bmatrix}\begin{bmatrix} \mathbf{m}(t) \\ \boldsymbol{\omega}(t) \end{bmatrix} + \begin{bmatrix} 0_3 \\ I_3 \end{bmatrix}\mathbf{w}(t), \quad \mathbb{E}[\mathbf{w}(t)\mathbf{w}(\tau)^\top] = \sigma_{\text{ori}}^2\delta(t-\tau). \tag{22}$$

We assume $C(\mathbf{m}(t))$ to be equal to $C(\mathbf{m}(t_k))$ in the interval $t \in [t_k, t_{k+1}]$. The discrete-time version of this constant velocity model for the orientation is then in the form

$$\mathbf{x}_{k+1}^{\text{ori}} = F_k^{\text{ori}}\mathbf{x}_k^{\text{ori}} + G_k^{\text{ori}}\mathbf{w}_k^{\text{ori}}, \quad \mathbf{w}_k^{\text{ori}} \sim \mathcal{N}(\mathbf{0}, Q^{\text{ori}}), \tag{23}$$

where

$$\mathbf{x}_k^{\text{ori}} = \begin{bmatrix} \mathbf{m}_k \\ \boldsymbol{\omega}_k \end{bmatrix}, \qquad F_k^{\text{ori}} = \begin{bmatrix} I_3 & TC(\mathbf{m}_k) \\ 0_3 & I_3 \end{bmatrix}, \tag{24}$$
$$G_k^{\text{ori}} = \begin{bmatrix} T^2 C(\mathbf{m}_k) & 0_3 \\ 0_3 & TI_3 \end{bmatrix}, \quad Q^{\text{ori}} = \frac{\sigma_{\text{ori}}^2}{T} \begin{bmatrix} \frac{1}{3} I_3 & \frac{1}{2} I_3 \\ \frac{1}{2} I_3 & I_3 \end{bmatrix}.$$

5.3 Orientation State (Quaternion)

In the second and third orientation representation (see Section 3.2 and 3.3), the orientation is encoded by the unit quaternion $\mathbf{q}_k$ and the extended quaternion $\bar{\mathbf{q}}_k$, respectively. The motion model for each of them is equivalent, so only $\bar{\mathbf{q}}_k$ is considered onwards. Its dynamics has a similar expression as in (19) with

$$\frac{d}{dt}\bar{\mathbf{q}}(t) = \frac{1}{2}\boldsymbol{\omega}(t) \odot \bar{\mathbf{q}}(t) \tag{25a}$$
$$= \frac{1}{2} S(\boldsymbol{\omega}(t))\bar{\mathbf{q}}(t) = \frac{1}{2}\bar{S}(\bar{\mathbf{q}}(t))\boldsymbol{\omega}(t), \tag{25b}$$

where the cross product in (19) is replaced with the quaternion product $\odot$, and where $S(\boldsymbol{\omega})$ and $\bar{S}(\bar{\mathbf{q}})$ are defined as

$$S(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & -\omega_z & \omega_y \\ \omega_y & \omega_z & 0 & -\omega_x \\ \omega_z & -\omega_y & \omega_x & 0 \end{bmatrix}, \qquad \bar{S}(\bar{\mathbf{q}}) = \begin{bmatrix} -\bar{q}_1 & -\bar{q}_2 & -\bar{q}_3 \\ \bar{q}_0 & \bar{q}_3 & -\bar{q}_2 \\ -\bar{q}_3 & \bar{q}_0 & \bar{q}_1 \\ \bar{q}_2 & -\bar{q}_1 & \bar{q}_0 \end{bmatrix}.$$

As in (21), the angular velocity $\boldsymbol{\omega}$ is modeled to be driven by white Gaussian noise. This gives a similar expression of the joint model as in (22), where

$$\frac{d}{dt}\begin{bmatrix} \mathbf{q}(t) \\ \boldsymbol{\omega}(t) \end{bmatrix} = \begin{bmatrix} 0_{4\times4} & \frac{1}{2}\bar{S}(\bar{\mathbf{q}}(t)) \\ 0_{3\times4} & 0_{3\times3} \end{bmatrix}\begin{bmatrix} \bar{\mathbf{q}}(t) \\ \boldsymbol{\omega}(t) \end{bmatrix} + \begin{bmatrix} 0_{4\times3} \\ I_3 \end{bmatrix}\mathbf{w}(t), \quad \mathbb{E}[\mathbf{w}(t)\mathbf{w}(\tau)^\mathsf{T}] = \sigma_{\text{ori}}^2 I_3 \delta(t-\tau).$$

By approximating $\bar{S}(\bar{\mathbf{q}}(t))$ to be equal to $\bar{S}(\bar{\mathbf{q}}(t_k))$ in the interval $t \in [t_k, t_{k+1}]$, we obtain a discrete-time model in the form (23), for which we have

$$\mathbf{x}_k^{\text{ori}} = \begin{bmatrix} \bar{\mathbf{q}}_k \\ \boldsymbol{\omega}_k \end{bmatrix}, \qquad F_k^{\text{ori}} = \begin{bmatrix} I_4 & \frac{T}{2}\bar{S}(\bar{\mathbf{q}}_k) \\ 0_{3\times4} & I_3 \end{bmatrix}, \tag{26a}$$
$$G_k^{\text{ori}} = \begin{bmatrix} \frac{T^2}{2}\bar{S}(\bar{\mathbf{q}}_k) & 0_{3\times4} \\ 0_{4\times3} & TI_3 \end{bmatrix}, \quad Q^{\text{ori}} = \frac{\sigma_{\text{ori}}^2}{T} \begin{bmatrix} \frac{1}{3} I_3 & \frac{1}{2} I_3 \\ \frac{1}{2} I_3 & I_3 \end{bmatrix}. \tag{26b}$$

5.4 Discussion

The motion models for the two orientation representations are in fact very similar. Essentially, the cross product in (19) is replaced with a quaternion product in (25). It should also be noted that the angular velocity $\boldsymbol{\omega}$ in this section is provided in global coordinates. To obtain the dynamics of the angular velocity $\boldsymbol{\omega}$ in local coordinates, the quaternion multiplication in (25) has to be swapped. However, a similar reformulation does not exist for the angular velocity in (19). This states an important difference between the two models.

5.5 Extended Kalman Filter

Based on the state-space model derived in the previous sections, the filtering problem can be solved using an extended Kalman filter (EKF). Given an initial state $\mathbf{x}_0$ and an initial state covariance P_0, the filtering can be performed following Algorithm 1.

Algorithm 1 Extended Kalman filter with addition projection step

Initialize the estimate $\hat{\mathbf{x}}_{0|-1} = \mathbf{x}_0$ and $P_{0|-1} = P_0$.
for $k = 0$ to $N - 1$ **do**

1. Perform measurement update by computing

$$\hat{\mathbf{y}}_{k|k-1} = \mathbf{h}(\hat{\mathbf{x}}_{k|k-1}), \tag{27a}$$
$$H_k = \frac{d}{d\mathbf{x}_k}\mathbf{h}(\hat{\mathbf{x}}_{k|k-1}), \tag{27b}$$
$$S_k = H_k P_{k|k-1} H_k^{\mathsf{T}} + R, \tag{27c}$$
$$K_k = P_{k|k-1} H_k^{\mathsf{T}} S_{k|k-1}^{-1}, \tag{27d}$$
$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + K_k(\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1}), \tag{27e}$$
$$P_{k|k} = (I - K_k H_k) P_{k|k-1}. \tag{27f}$$

2. Project the estimate onto the tracking volume

$$\hat{\mathbf{x}}_{k|k}^{\text{proj}} = \Pi_{\mathcal{C}}(\mathbf{x}_{k|k}). \tag{28}$$

   Further, if $\mathbf{x}_{k|k} \notin \mathcal{C}$ then set the velocities $\mathbf{v}_{k|k} = \mathbf{0}$ and $\omega_{k|k} = \mathbf{0}$ to zero

3. Perform time update by computing

$$\hat{\mathbf{x}}_{k+1|k} = F_k \hat{\mathbf{x}}_{k|k}^{\text{proj}}, \tag{29a}$$
$$P_{k+1|k} = F_k P_{k|k} F_k^{\mathsf{T}} + Q_k. \tag{29b}$$

end for

If the estimated position $\hat{\mathbf{r}}_{k|k}$ is far away from the sensor network, both $\mathbf{h}(\hat{\mathbf{x}}_{k|k})$ and it derivative $\frac{d}{d\mathbf{x}_k}\mathbf{h}(\hat{\mathbf{x}}_{k|k})$ will be small, which in turn results in an uninformative measurement update $\mathbf{x}_{k|k} = \mathbf{x}_{k|k-1}$. This eventually leads to divergence of the filter. To prevent this, an additional projection step is included, see Step 2. This step is not included in the standard EKF, but makes this specific application more robust. By projected the filtered state $\mathbf{x}_{k|k}$ onto a set of the state-space in which the state should be contained under normal operation, this divergence can be prevented.

In this case, the set $\mathcal{C}$ corresponds to a virtual box above the sensor network with the dimensions $1.2\,\text{m} \times 1.2\,\text{m} \times 0.6\,\text{m}$. Any estimated position $\hat{\mathbf{r}}_{k|k}$ that is

(a) The sensor platform *(b)* The object platform

Figure 3: *The sensor and object platform with Vicon markers. The sensor platform has four three-axis magnetometers mounted in the corners of an rectangle. The object platform consist of two magnets (encapsulated in the two Lego blocks) aligned in orthogonal directions. That object platform was only used in the multi-dipole experiment. Photo: Martin Stenmarck (2015).*

outside this region is projected back onto this box. In addition, if any projection is needed, the velocities $\mathbf{v}_{k|k}$ and $\boldsymbol{\omega}_{k|k}$ are set to zero. We also consider a maximum value $m_{\max}$ of the magnetic dipole moment. This gives the restriction $\|\hat{\mathbf{m}}_{k|k}\| \leq m_{\max}$ or $\|\hat{\tilde{\mathbf{q}}}_{k|k}\| \leq \sqrt{m_{\max}}$ depending on which of the two orientation representations that are used.

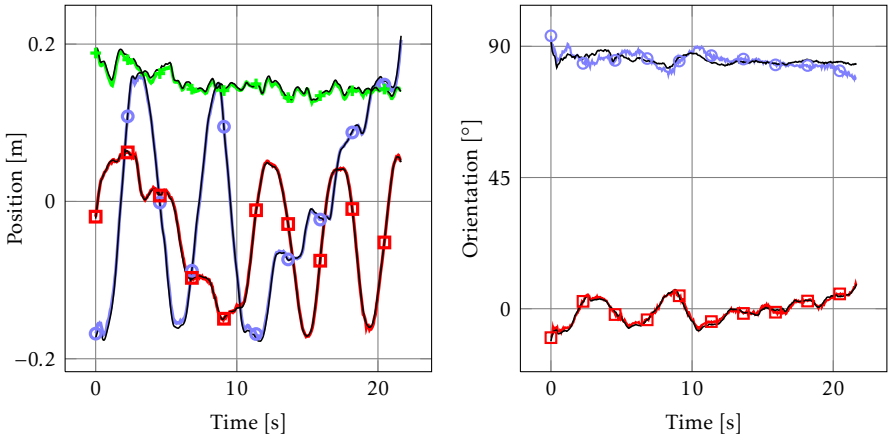
6 Real Data Experiments

To validate the proposed models, real data experiments were conducted. A network that consists of four magnetometers (3-Axis Digital Compass IC, HMC5983, Honeywell) was designed with a sampling frequency of 220 Hz. The sensors were deployed in the corners of a rectangle with the dimension 30 cm × 17.5 cm. The sensor platform is shown in Figure 3a. With this sensor platform, experiments were conducted to validate both the single dipole model (3), and the multi-dipole model (5). To obtain ground truth data, an optical motion capture system (Vicon)[2] was used.

6.1 Single Dipole Experiment

The single dipole object consisted of two neodymium magnets mounted placed upon each other, both with a diameter of 12 mm and a height of 6 mm. Markers to be detected by the Vicon-system were attached to the object platform on which the magnets were attached, as well as to the sensor network.

[2]High accuracy reference measurements are provided through the use of the Vicon real-time tracking system courtesy of the UAS Technologies Lab, Artificial Intelligence and Integrated Computer Systems Division (AIICS) at the Department of Computer and Information Science (IDA), Linköping University, Sweden. `http://www.ida.liu.se/divisions/aiics/aiicssite/index.en.shtml`



(a) Estimated and ground truth position *(b) Estimated and ground truth orientation*

Figure 4: *Experiment with a single dipole object for tracking 5 DoF pose (3 DoF for position and 2 DoF for orientation). The estimates are displayed in colored, marked, thick line and the ground truth in thin unmarked black lines. Blue circle: x-coordinate/angle, red square: y-coordinate/angle, green cross: z-coordinate.*

During the experiments, the object platform was moved approximately 20 cm above the magnetometer network in different directions close to the magnetometer network.

For processing the data, the single dipole model (3) was used together with the orientation representation described in Section 3.1. A constant velocity motion model was used for the position, see Section 5.1, as well as for the orientation, see Section 5.2. The process noise was set to $\sigma_{\text{pos}} = 0.1\,\text{m}\,\text{s}^{-2}$ for the acceleration and $\sigma_{\text{ori}} = 1\,\text{rad}\,\text{s}^{-2}$ for the angular acceleration. The covariance of the measurement noise was estimated using stationary data without any object.

In Figure 4a, all three Cartesian components of the estimated position are displayed together with the ground truth measured by the Vicon system. Note that the single dipole model is only able to estimate two DoF for orientation. These two DoF can be compared with the corresponding two DoF provided by the Vicon system. This is displayed in Figure 4b.

In all essence, both the full 3D position and 2D orientation are tracked with high accuracy. By comparing the estimated pose with the ground truth pose, the estimation performance for both the position and orientation can be computed.

In Table 1, the performance is presented in terms of root-mean-square error (RMSE), which has been divided into variance and bias contribution; see Appendix B for how these quantities have been computed. The orientation error was computed according to (53a). According to that table, the sensor network can estimate position with a five millimeter accuracy in position and two of de-

Table 1: *Single dipole experiment: RMSE, bias and variance for position and orientation*

Quantity	RMSE	Bias	$\sqrt{\text{Var}}$
Position	4.95 mm	4.90 mm	0.61 mm
Orientation	1.85°	1.82°	0.26°

grees accuracy in orientation. In these errors, the bias is clearly the dominating contribution.

6.2 Multi-Dipole Experiment

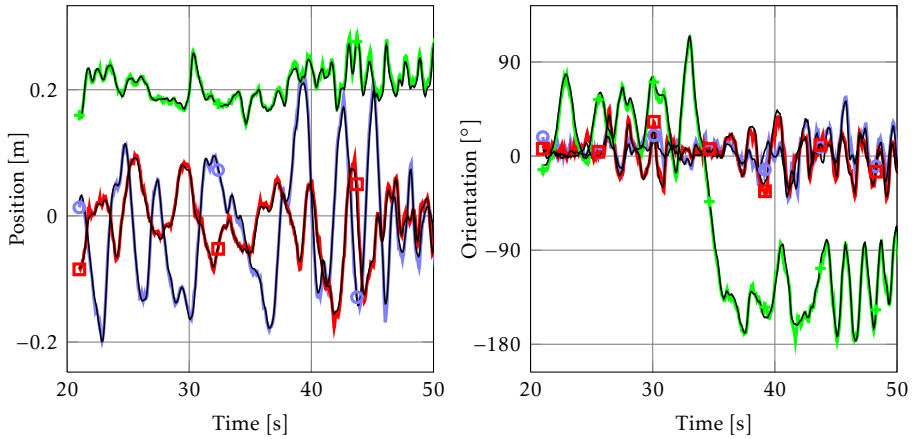
To acquire full position and orientation information, a multi-dipole object was constructed. Two neodymium magnets, each with a diameter of 12 mm and a height of 6 mm, were deployed perpendicularly on a distance 6.7 mm from each other, see Figure 1 for an illustration of this geometry. Figure 3b shows the object platform used in the experiment.

Three experiments were performed, each lasting approximately 100 s. For processing the data, the multi-dipole model was used, see Section 2.2. Since the two magnets were of the same size, their magnitudes were assumed to be equal. Therefore, the extended quaternion $\bar{\mathbf{q}}_k$, see Section 3.3, was used to model the orientation and the unknown magnitude together with the quaternion motion model presented in Section 5.3. The same tuning parameters as in the single dipole experiment was used. For the first experiment, the results for both position and orientation are presented in Figure 5. Only a fraction of the time span is presented to make the plots readable. According to these plots, good tracking performance was achieved for all dimensions in both position and orientation. In

Table 2: *Multiple dipole experiment: RMSE, bias and variance for position, full orientation and 2D orientation.*

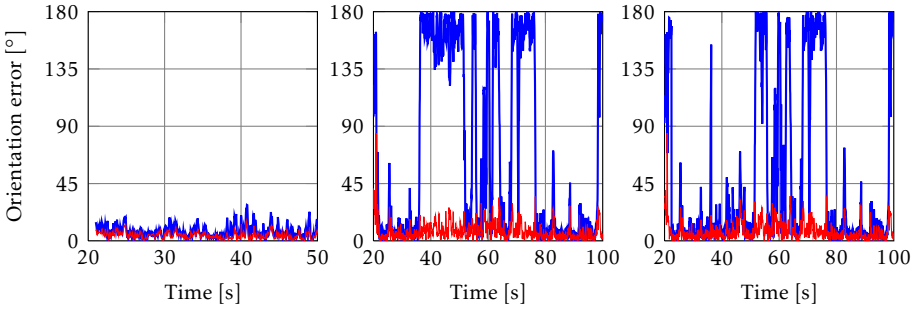
Quantity	RMSE	Bias	$\sqrt{\text{Var}}$
Position	11.89 mm	11.80 mm	1.14 mm
3D Orientation	8.80°	8.77°	0.58°
2D Orientation	4.97°	4.95°	0.37°

Table 2, the performance is computed in terms of RMSE, bias and variance in a similar manner as for Table 1. The orientation error was computed according to (53b) in Appendix B. In comparison with the single dipole model, the estimation performance was twice as bad for position, and four times as bad for orientation in comparison with the single dipole experiment in Table 1. The degraded performance is not surprising because an additional state is estimated in comparison to the previous model.



(a) Estimated and ground truth position *(b) Estimated and ground truth orientation*

Figure 5: *Experiment with a multi-dipole object for tracking the full position and orientation state (3 DoF for position and 3 DoF for orientation). The estimates is displayed in colored, marked, thick line and the ground truth in thin unmarked black lines. Blue circle: x-coordinate/angle, red square: y-coordinate/angle, green cross: z-coordinate/angle.*



(a) *(b)* *(c)*

Figure 6: *Orientation error for experiment 1 and 2 with and without filter banks. Red: full 3D orientation error* (53b)*, blue: 2D orientation error* (53c)*. (a): Experiment 1. (b): Experiment 2. (c): Experiment 2 using filter banks.*

As discussed in Section 4, the NLL $-\log p(\mathbf{y}_k|\mathbf{x}_k)$ is less sensitive to rotations around the dipole moment $\mathbf{m}_k$ than around any of the other two axis of rotation. This means that the additional DoF for orientation, acquired by extending the single dipole model to a multi-dipole model, is harder to estimate than the remaining two DoF. To analyze this, the orientation error for the renaming two DoF are computed according to (53c). Both the 2D and the 3D orientation errors are presented in Table 2 and in Figure 6a. According to Table 2, the 3D orientation

error is almost twice as bad in comparison to the 2D orientation error.

A second experiment with the same experimental setup was also conducted. The orientation errors for that data set is presented in Figure 6b. As shown in that figure, the 2D orientation error is estimated correctly with a fairly low error (red curve). However, starting at time 40 s, the full orientation error switches between the true and a false mode being approximately 180° apart. This can be addressed to the bimodal likelihood discussed in Section 4.

To circumvent this problem, a filter bank consisting of two Extended Kalman filters was considered where each of them was restricted to be within each of the two modes. See for example (Gustafsson, 2012, Chapater 10) for reference on Kalman filter banks. By evaluating the posterior of the two modes, the correct model could be selected. This strategy was tested on the same data set as displayed in Figure 6b and the result is presented in Figure 6c. According to that figure, the performance increases, however, the correct mode can still not always be resolved. A more reliable way to resolve this ambiguity is a subject for further research.

7 Applications

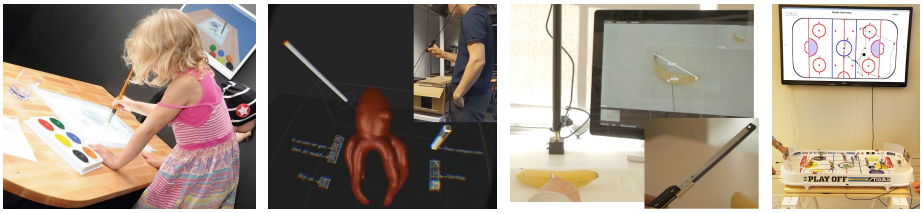
The magnetic localization technique described in this paper has multiple advantages in comparison to other localization techniques, for example, it (i) only includes low-cost components, (ii) does not require line-of-sight, and (ii) has a high accuracy in estimating both position and orientation. To show the applicability of the technique, four different demonstrators have been realized.

7.1 Virtual Watercolors

Museums and science centers have a high need for technology enabling interactive exhibits that encourage visitors to experiment and explore. In exhibits where spatial information is important, a localization system is required. These systems need to be intuitive for the visitors to control and interact with. In this context, the magnetic localization technique was used in an exhibition case mimicking water color painting, see Figure 7a. The user interacts using a regular painting brush equipped with a permanent magnet and the painting is displayed on a screen.

7.2 Interactive 3D Modeling

The hand-held device is also suitable for interaction and manipulation of three-dimensional virtual objects. The hand-held device equipped with a magnet can be used to pull, push and smoothen textures of an object, as well as moving and turning it, see Figure 7b. Both the virtual object and the virtual device can be observed through a head-mounted display making the interaction intuitive and realistic.



(a) *(b)* *(c)* *(d)*

Figure 7: *Applications realized with the proposed technology. (a) Digital watercolors. Photo: Anders Ynnerman (2012), (b) Digital watercolors. Photo: Olle Grahn and Isabelle Forsman (2015), (c) Digital pathology. Photo: Linkin AB (2014), (d) Digital table hockey game. Photo: Martin Stenmarck (2015).*

7.3 Digital Pathology

The magnetic localization technique has been used to improve the workstation that pathologists use when examining tissues. By mounting a magnet in a scalpel, see Figure 7c, a digital record can be constructed of the actions that have been performed. This saves time and also removes manual non-ergonomic activities.

7.4 Digital Table Hockey Game

By mounting a magnet in a puck for a table hockey game, the puck can be localized in real time, and meta information can be extracted, e.g., number of goals, see Figure 7d.

8 Conclusion and Future Work

In this paper, a framework for estimating the position and orientation of objects consisting of one or more magnetic dipoles was presented. The problem was cast into a statistical filtering problem, including both sensor models and motion models. The sensor models include (i) a point object model (one dipole), (ii) an extended object model (multi-dipole object) and (iii) multiple extended objects (multiple multi-dipole objects). Due to rotational symmetry, the point object model can only provide five degrees of freedom for position and orientation. Except for special geometries, this symmetry is not present in the extended object model, where all six degrees of freedom can be resolved. An analysis was provided showing that the sixth degree of freedom is weakly observable and bimodal. The models were validated on real data with a high-accuracy optical reference system. With a sensor network of four three-axis magnetometers, a tracking performance of 5 mm and 2° for model (i) and 12 mm and 9° for model (ii) was achieved. The vast majority of these errors consist of bias.

Future work should focus on dedicated calibration routines for magnetometer networks. These routines may calibrate for position, orientation, bias, gain, and skewness of the magnetometers, as well as the parameters for the mutli-dipole geometry. With a well calibrated sensor network, the bias contribution of the tracking performance could be significantly reduced. This would also facilitate resolving the ambiguity present in the multi-dipole object (ii).

Appendix

A Supplementary Details for Section 4

In this appendix, we investigate ambiguities for determining the orientation of the multi-dipole object. We consider the sensor model (5) in the form

$$\mathbf{f}(\varepsilon, R_k) = \sum_{l=1}^{L} J(\mathbf{r} + \varepsilon R_k \mathbf{s}_l) R_k \mathbf{b}_l, \tag{30}$$

where only one sensor (with position $\boldsymbol{\theta} = \mathbf{0}$) is considered to make the notation easier. In this appendix, the index $k = \{1, 2\}$ denotes two different instances of rotations that will compared. Further, without loss of generality we also assume $\|\bar{\mathbf{q}}_k\| = 1$ and denote $R_k = R(\bar{\mathbf{q}}_k / \|\bar{\mathbf{q}}_k\|)$. The parameter ε determines the size of the multi-dipole object.

We want to analyze the properties of (30) when the object is far away from the sensor, or equivalently, if the object size is small, this means when $\varepsilon \to 0$. We perform this analysis with a Taylor expansion of (30) around $\varepsilon = 0$. By keeping the first two terms, we obtain

$$\mathbf{f}(\varepsilon, R_k) = \sum_{n=0}^{\infty} \frac{1}{n!} \frac{\partial^n}{\partial \varepsilon^n} \mathbf{f}(\varepsilon, R_k)\Big|_{\varepsilon=0} \varepsilon^n = \mathbf{f}(0, R_k) + \frac{\partial}{\partial \varepsilon} \mathbf{f}(\varepsilon, R_k)\Big|_{\varepsilon=0} \varepsilon + \mathcal{O}(\varepsilon^2) \tag{31a}$$

$$= \underbrace{A(R_k)}_{\text{1st term}} + \underbrace{B(R_k)}_{\text{2nd term}} \varepsilon + \mathcal{O}(\varepsilon^2). \tag{31b}$$

The term $A(R_k)$ decays as $\|\mathbf{r}\|^{-3}$ and $B(R)$ as $\|\mathbf{r}\|^{-4}$ with distance $\|\mathbf{r}\|$. Higher order terms decay as $\|\mathbf{r}\|^{-5}$ or more. Therefore, if the object is far away from the sensors, these two terms will dominate.

We introduce two rotation matrices R_1 and R_2 with

$$R_2 = \tilde{R} R_1, \tag{32}$$

where $\tilde{R}$ is the rotation difference between R_1 and R_2. We now derive conditions on the rotation difference $\tilde{R}$ such that $A(R_1) = A(R_2)$ and $B(R_1) = B(R_2)$. More precisely, in the two following subsection we derive that

1. $A(R_1) = A(R_2)$ for all $\mathbf{r} \in \mathbb{R}^3$ when $\tilde{R}\mathbf{m}_1 = \mathbf{m}_1$, where $\mathbf{m}_1 = R_1 \sum_{l=1}^{L} \mathbf{b}_l$.

2. $A(R_1) = A(R_2)$ and $B(R_1) = B(R_2)$ for all $\mathbf{r} \in \mathbb{R}^3$ when $\tilde{R}\mathbf{m}_1 = \mathbf{m}_1$ and $\tilde{R}\tilde{R} = I$, i.e., a 180° rotation around $\mathbf{m}_1$.

This means that any rotation 180° around $\mathbf{m}_1$ affects neither the first term, nor the second term in the Taylor expansion (31). This results in a bimodal likelihood where the difference between these two modes only relies on even higher order terms in the Taylor expansion.

A.1 First Term $A(R_k)$

The first term in the Taylor expansion (31) is equal to the corresponding single dipole model

$$A(R_k) = \sum_{l=1}^{L} J(\mathbf{r}) R_k \mathbf{b}_l = J(\mathbf{r}) R_k \mathbf{b} = J(\mathbf{r}) \mathbf{m}_k, \tag{33}$$

where

$$\mathbf{m}_k = R_k \mathbf{b}, \qquad \text{and} \qquad \mathbf{b} = \sum_{l=1}^{L} \mathbf{b}_l. \tag{34}$$

As for the single dipole model, only two out of three DoF for orientation are observable. We perform this analysis by considering the difference $A(R_2) - A(R_1)$. If this expression is zero for some certain choice of rotation difference $\tilde{R}$, the two orientations R_1 and R_2 cannot be resolved based on this term.

Theorem 1. *Consider the single dipole term* (33)*. Then*

$$A(R_1) = A(R_2) \quad \forall \mathbf{r} \in \mathbb{R}^3 \tag{35a}$$

if and only if

$$\tilde{R}\mathbf{m}_1 = \mathbf{m}_1, \quad \textit{where} \quad \mathbf{m}_1 = R_1 \sum_{l=1}^{L} \mathbf{b}_l \tag{35b}$$

and where $\tilde{R}$ is given by (32)*.*

Proof: Consider

$$A(R_2) - A(R_1) = J(\mathbf{r}) \tilde{R} R_1 \mathbf{b} - J(\mathbf{r}) R_1 \mathbf{b} = J(\mathbf{r})(\tilde{R}\mathbf{m}_1 - \mathbf{m}_1), \tag{36}$$

which is equal to zero for all $\mathbf{r}$ if and only if

$$\tilde{R}\mathbf{m}_1 = \mathbf{m}_1. \tag{37}$$

□

Consequently, if the object is far away from the sensor, the dipole term will dominate and the third DoF for the orientation (the one that corresponds to rotation around $\mathbf{m}_1$) will be the hardest one to estimate.

A.2 Second Term $B(R_k)$

The second term of the Taylor expansion (31) can be expressed as

$$\begin{aligned} B(R_k) &= \sum_{l=1}^{L} \frac{\partial J(\mathbf{r}) R_k \mathbf{b}_l}{\partial \mathbf{r}} R_k \mathbf{s}_l \\ &= \sum_{l=1}^{L} \frac{3}{\|\mathbf{r}\|^5} \left((\mathbf{b}_l^\mathsf{T} R_k^\mathsf{T} \mathbf{r}) I + \mathbf{r} \mathbf{b}_l^\mathsf{T} R_k^\mathsf{T} + R_k \mathbf{b}_l \mathbf{r}^\mathsf{T} - 5 \frac{\mathbf{r}\mathbf{r}^\mathsf{T}}{\mathbf{r}^\mathsf{T}\mathbf{r}} (\mathbf{b}_l^\mathsf{T} R_k^\mathsf{T} \mathbf{r}) \right) R_k \mathbf{s}_l \\ &= \sum_{l=1}^{L} \frac{3}{\|\mathbf{r}\|^5} \left(R_k \mathbf{s}_l \mathbf{b}_l^\mathsf{T} R_k^\mathsf{T} + (\mathbf{b}_l^\mathsf{T} \mathbf{s}_l) I + R_k \mathbf{b}_l \mathbf{s}_l^\mathsf{T} R_k^\mathsf{T} - \frac{5\mathbf{r}^\mathsf{T} (R_k \mathbf{s}_l \mathbf{b}_l^\mathsf{T} R_k^\mathsf{T} + R_k \mathbf{b}_l \mathbf{s}_l^\mathsf{T} R_k^\mathsf{T}) \mathbf{r}}{2\|\mathbf{r}\|^2} I \right) \mathbf{r} \\ &= \frac{3}{\|\mathbf{r}\|^5} \left(W_k - \frac{5\mathbf{r}^\mathsf{T} W_k \mathbf{r}}{2\|\mathbf{r}\|^2} I + \sum_{l=1}^{L} (\mathbf{b}_l^\mathsf{T} \mathbf{s}_l) I \right) \mathbf{r}, \end{aligned} \tag{38}$$

where

$$W_k = R_k U R_k^\mathsf{T}, \quad \text{with} \quad U = \sum_{l=1}^{L} \mathbf{s}_l \mathbf{b}_l^\mathsf{T} + \mathbf{b}_l \mathbf{s}_l^\mathsf{T}. \tag{39}$$

We want to analyze under which conditions $B(R_2) = B(R_1)$. The analysis is performed in the same manner as for the single dipole term by considering the difference

$$B(R_2) - B(R_1) = \frac{3}{\|\mathbf{r}\|^5} \left(\left(\tilde{R} W_1 \tilde{R}^\mathsf{T} - W_1 \right) - \frac{5\mathbf{r}^\mathsf{T} \left(\tilde{R} W_1 \tilde{R}^\mathsf{T} - W_1 \right) \mathbf{r}}{2\|\mathbf{r}\|^2} I \right) \mathbf{r}. \tag{40}$$

One can easily conclude that (40) is equal to zero for all $\mathbf{r} \in \mathbb{R}^3$ if and only if

$$\tilde{R} W_1 \tilde{R}^\mathsf{T} - W_1 = 0. \tag{41}$$

Before deriving the condition on W_1 and $\tilde{R}$ for this equality to hold, the center of the object needs to be defined.

The position of each dipole is described (in local coordinates) as a displacement $\mathbf{s}_l$ relative to the center of the object. Obviously, this center is not unique. We assume the center of the object has been selected such that

$$\underbrace{\left(\sum_{l=1}^{L} \mathbf{s}_l \mathbf{b}_l^\mathsf{T} + \mathbf{b}_l \mathbf{s}_l^\mathsf{T} \right)}_{=U} \underbrace{\sum_{l=1}^{L} \mathbf{b}_l}_{=\mathbf{b}} = \mathbf{0}. \tag{42}$$

If this property does not hold for the selected center of the object, a new center can be defined using the following translation

$$\mathbf{s}_l^{\text{new}} = \mathbf{s}_l - \Delta\mathbf{s}, \qquad \text{where} \qquad \Delta\mathbf{s} = \frac{1}{\mathbf{b}^\mathsf{T}\mathbf{b}} \left(I - \frac{\mathbf{b}\mathbf{b}^\mathsf{T}}{2\mathbf{b}^\mathsf{T}\mathbf{b}} \right) U \mathbf{b}. \tag{43}$$

By using $\mathbf{s}_l^{\text{new}}$ in (42), that property is indeed fulfilled. Note that W_k has also an eigenvalue 0 with the eigenvector $\mathbf{m}_k$ since $W_k\mathbf{m}_k = R_k U R_k^\mathsf{T}\mathbf{m}_k = R_k U\mathbf{b} = \mathbf{0}$. With this definition of the center of the object, the following theorem provides the condition for (40) to be equal to zero.

Lemma 1. *Consider a symmetric matrix $W = W^\mathsf{T} \in \mathbb{R}^{3\times3}$ and a rotation matrix $\tilde{R} \in \mathsf{SO}(3)$. Also assume that they have a common eigenvector $\mathbf{m}$ with*

$$W\mathbf{m} = \mathbf{0}, \qquad \tilde{R}\mathbf{m} = \mathbf{m}. \tag{44a}$$

Then

$$\tilde{R}W\tilde{R}^\mathsf{T} - W = 0 \tag{44b}$$

if and only if $\tilde{R}\tilde{R} = I$ or $\lambda_1 = \lambda_2$, where λ_1 and λ_2 are the two non-zero eigenvalues of W.

Proof: Since W is symmetric, it can be written

$$W = QDQ^\mathsf{T}, \tag{45}$$

where Q is orthogonal and $\det(Q) = 1$. Since $W\mathbf{m} = \mathbf{0}$, one of the diagonal components in D is zero. We can without loss of generality order the eigenvalues such that $D = \operatorname{diag}(\lambda_1, \lambda_2, 0)$. Since both W and $\tilde{R}$ share the same eigenvector $\mathbf{m}$ with eigenvalue 0 and 1, respectively, we have that

$$Q^\mathsf{T}WQ = \begin{bmatrix}\lambda_1 & 0 & 0\\ 0 & \lambda_2 & 0\\ 0 & 0 & 0\end{bmatrix}, \quad Q^\mathsf{T}\tilde{R}Q = \begin{bmatrix}\mathrm{c}(\theta) & -\mathrm{s}(\theta) & 0\\ \mathrm{s}(\theta) & \mathrm{c}(\theta) & 0\\ 0 & 0 & 1\end{bmatrix},$$

where θ corresponds to the rotation angle around $\mathbf{m}$ and $\mathrm{c}(\theta) = \cos(\theta)$, $\mathrm{s}(\theta) = \sin(\theta)$. By applying Q^T from the left and Q from the right on (44b), we obtain

$$\begin{aligned} Q^\mathsf{T}\tilde{R}W\tilde{R}^\mathsf{T}Q - Q^\mathsf{T}WQ &= Q^\mathsf{T}\tilde{R}QDQ^\mathsf{T}\tilde{R}^\mathsf{T}Q - D \\ &= \begin{bmatrix}\mathrm{c}(\theta) & -\mathrm{s}(\theta) & 0\\ \mathrm{s}(\theta) & \mathrm{c}(\theta) & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\lambda_1 & 0 & 0\\ 0 & \lambda_2 & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\mathrm{c}(\theta) & \mathrm{s}(\theta) & 0\\ -\mathrm{s}(\theta) & \mathrm{c}(\theta) & 0\\ 0 & 0 & 1\end{bmatrix} - \begin{bmatrix}\lambda_1 & 0 & 0\\ 0 & \lambda_2 & 0\\ 0 & 0 & 0\end{bmatrix} \\ &= (\lambda_1 - \lambda_2)\mathrm{s}(\theta)\begin{bmatrix}-\mathrm{s}(\theta) & \mathrm{c}(\theta) & 0\\ \mathrm{c}(\theta) & \mathrm{s}(\theta) & 0\\ 0 & 0 & 0\end{bmatrix}. \end{aligned} \tag{46}$$

This expression is equal to zero if and only if $\lambda_1 = \lambda_2$ or $\theta = n\pi$, where $n \in \mathbb{Z}$, which implies that $\tilde{R} = \tilde{R}^\mathsf{T}$. □

The conclusion is now formalized in the following theorem.

Theorem 2. *Consider the single dipole term* (33) *and the higher order term* (38). *Further, assume that the center of the object has been defined such that* (42) *is fulfilled. Then*

$$A(R_1) = A(R_2) \quad \textit{and} \quad B(R_1) = B(R_2) \quad \forall\mathbf{r} \in \mathbb{R}^3 \tag{47}$$

if and only if

$$\tilde{R}\mathbf{m}_1 = \mathbf{m}_1 \quad \textit{and} \quad \tilde{R} = \tilde{R}^\mathsf{T}, \tag{48}$$

or

$$\tilde{R}\mathbf{m}_1 = \mathbf{m}_1 \quad \textit{and} \quad \lambda_1 = \lambda_2, \tag{49}$$

where , $\mathbf{m}_1 = R_1 \sum_{l=1}^{L} \mathbf{b}_l$ *is given by* (34)*,* $\tilde{R}$ *is given by* (32)*, and* λ_1 *and* λ_2 *are eigenvalues of* U *defined in* (39)*.*

Proof: From Lemma 1, we know that $A(R_1) = A(R_2) \quad \forall \mathbf{r} \in \mathbb{R}^3$ is fulfilled if and only if $\tilde{R}\mathbf{m}_1 = \mathbf{m}_1$. For the second equality, we have

$$B(R_2) - B(R_1) = \frac{3}{\|\mathbf{r}\|^5}\left(V - \frac{5\mathbf{r}^\mathsf{T} V \mathbf{r}}{2\|\mathbf{r}\|^2} I\right)\mathbf{r}, \tag{50}$$

with

$$V = \tilde{R} W_1 \tilde{R}^\mathsf{T} - W_1, \tag{51}$$

where W_1 is given by (39). Further, since $U\mathbf{b} = \mathbf{0}$, we have that $W_1\mathbf{m}_1 = R_1 U R_1^\mathsf{T}\mathbf{m}_1 = R_1 U\mathbf{b} = \mathbf{0}$. According to Lemma 1, equation (51) is fulfilled if and only if $\tilde{R}\tilde{R} = I$ or $\lambda_1 = \lambda_2$, where λ_1 and λ_2 are the two non-zero eigenvalues of W_1. Finally we notice that $W_1 = R_1 U R_1^\mathsf{T}$ has the same eigenvalues as U, since R_1 is orthogonal. Consequently, λ_1 and λ_2 are also eigenvalues of U. This completes the proof. □

Remark 1. The condition $\lambda_1 = \lambda_2$ provides a condition on the dipole geometry encoded in U (39). The geometry should consequently not be chosen such that $\lambda_1 = \lambda_2$. Instead, it should preferably be chosen such that these two eigenvalues are as separated as possible.

B Performance measures

This appendix gives details about the performance measure that was used in the result section.

With the estimated position $\hat{\mathbf{r}}_k$ and the ground truth position $\mathbf{r}_k^0$, the estimation error was computed as

$$\boldsymbol{\epsilon}_k^{\text{pos}} = \hat{\mathbf{r}}_k - \mathbf{r}_k^0. \tag{52}$$

The orientation error was computed differently depending on the sensor model. For the single dipole model (3), the orientation error was computed as the angle between the true and estimated direction of the dipole moment

$$\boldsymbol{\epsilon}_k^{\text{ori}} = \arccos\left(\frac{\hat{\mathbf{m}}_{k|k}^\mathsf{T} R(\mathbf{q}_k^0)\mathbf{b}}{\|\hat{\mathbf{m}}_{k|k}\|\|\mathbf{b}\|}\right), \tag{53a}$$

where $\mathbf{q}_k^0$ is the quaternion describing the true orientation. This orientation error was used in Section 6.1.

For the multi-dipole model, the full orientation error was computed as the shortest angle between the estimated and true quaternion as

$$\boldsymbol{\epsilon}_k^{\text{ori,3D}} = 2\arccos\left(\frac{|\hat{\mathbf{q}}_{k|k}^{\mathsf{T}}\mathbf{q}_k^0|}{\|\hat{\mathbf{q}}_{k|k}\|\|\mathbf{q}_k^0\|}\right). \tag{53b}$$

For the multi-dipole model, the 2D orientation error was computed in the same manner as for the single dipole case as the angle between the true and estimated direction of the dipole moment vector

$$\boldsymbol{\epsilon}_k^{\text{ori,2D}} = \arccos\left(\left(R\left(\frac{\hat{\hat{\mathbf{q}}}_{k|k}}{\|\hat{\hat{\mathbf{q}}}_{k|k}\|}\right)\mathbf{b}\right)^{\mathsf{T}} R(\mathbf{q}_k^0)\mathbf{b}\right). \tag{53c}$$

The error (53b) and (53c) were used in Section 6.2.

For each of these errors, the corresponding root-mean-square error was computed as

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\|\boldsymbol{\epsilon}_k\|^2}. \tag{54}$$

This quantity can be decomposed in a bias and a variance part. A 4th order Butterworth-filter with the cutoff frequency $f_c = 10\,\text{Hz}$ was used to separate the error in a low and high frequency part

$$\boldsymbol{\epsilon}_k = \boldsymbol{\epsilon}_k^{\text{low}} + \boldsymbol{\epsilon}_k^{\text{high}} \tag{55}$$

and the variance and bias contributions to RMSE can be defined as

$$\sqrt{\text{variance}} = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\frac{1}{1-2Tf_c}\|\boldsymbol{\epsilon}_k^{\text{high}}\|^2}, \tag{56a}$$

$$\text{bias} = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\|\boldsymbol{\epsilon}_k\|^2 - \frac{1}{1-2Tf_c}\|\boldsymbol{\epsilon}_k^{\text{high}}\|^2}. \tag{56b}$$

Bibliography

D. Ashbrook, P. Baudisch, and S. White. Nenya: Subtle and eyes-free mobile input with a magnetically-tracked finger ring. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*, pages 2043–2046, Vancouver, BC, Canada, May 2011.

W. A. Beck. Localization of small magnets against a noisy background. In *Saratov Fall Meeting 2000*, pages 6–12, Saratov, Russia, 2001. International Society for Optics and Photonics.

M. Birsan. Non-linear Kalman filters for tracking a magnetic dipole. In *Proceedings of the International Conference on Marine Electromagnetics*, London, UK, March 2004.

M. Birsan. Unscented particle filter for tracking a magnetic dipole target. In *Proceedings of MTS/IEE OCEANS*, Washington, DC, USA, September 2005.

L. Chan, R.-H. Liang, M.-C. Tsai, K.-Y. Cheng, C.-H. Su, M. Y. Chen, W.-H. Cheng, and B.-Y. Chen. Fingerpad: Private and subtle interaction using fingertips. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST)*, pages 255–260, St Andrews, UK, October 2013.

N. A. Chaturvedi, A. K. Sanyal, and N. H. McClamroch. Rigid-body attitude control. *IEEE Control Systems*, 31(3):30–51, 2011.

M. S. Grewal and A. P. Andrews. *Kalman Filtering. Theory and Practice Using MATLAB.* John Wiley & Sons, Hoboken, NJ, USA, third edition, 2008.

R. Guignet, G. Bergonzelli, V. Schlageter, M. Turini, and P. Kucera. Magnet Tracking: a new tool for in vivo studies of the rat gastrointestinal motility. *Neurogastroenterology & Motility*, 18(6):472–478, 2006.

F. Gustafsson. *Statistical Sensor Fusion.* Studentlitteratur, 1 edition, 2012. Page 257-272.

C. Harrison and S. E. Hudson. Abracadabra: Wireless, high-precision, and unpowered finger input for very small mobile devices. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology (UIST)*, pages 121–124, Victoria, BC, Canada, October 2009.

D. R. Haynor, C. P. Somogyi, R. N. Golden, and G. B. Sanders. System and method to determine the location and orientation of an indwelling medical device, July 17 2001. US Patent 6,263,230.

C. Hu, M. Q.-H. Meng, and M. Mandal. Efficient magnetic localization and orientation technique for capsule endoscopy. *International Journal of Information Acquisition*, 2(1):23–36, 2005.

C. Hu, M.-H. Meng, M. Mandal, and X. Wang. 3-axis magnetic sensor array system for tracking magnet's position and orientation. In *Proceedings of the 6th World Congress on Intelligent Control and Automation (WCICA)*, volume 2, pages 5304–5308, Dalian, China, 2006. IEEE.

C. Hu, W. Yang, D. Chen, M.-H. Meng, and H. Dai. An improved magnetic localization and orientation algorithm for wireless capsule endoscope. In *Proceedings of the 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*, pages 2055–2058, Vancouver, BC, Canada, August 2008.

C. Hu, M. Li, S. Song, R. Zhang, M.-H. Meng, et al. A cubic 3-axis magnetic sensor array for wirelessly tracking magnet position and orientation. *IEEE Sensors Journal*, 10(5):903–913, 2010.

A. H. Jazwinski. *Stochastic Processes and Filtering Theory*, volume 64 of *Mathematics in Science and Engineering*. Academic Press, New York, NY, USA, 1970.

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.

R. J. Kozick and B. M. Sadler. Joint processing of vector-magnetic and acoustic-sensor data. In *SPIE Unattended Ground, Sea, and Air Sensor Technologies and Applications IX*, volume 6562, 2007a.

R. J. Kozick and B. M. Sadler. Classification via information-theoretic fusion of vector-magnetic and acoustic sensor data. In *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 2, pages 953–956, Honolulu, HI, USA, April 2007b.

R. J. Kozick and B. M. Sadler. Algorithms for tracking with an array of magnetic sensors. In *Sensor Array and Multichannel Signal Processing Workshop (SAM)*, pages 423–427, Darmstadt, Germany, July 2008.

J. B. Kuipers. *Quaternions and rotation sequences*. Princeton university press, Princeton, NJ, USA, 1999.

X. R. 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, 2003.

R.-H. Liang, K.-Y. Cheng, C.-H. Su, C.-T. Weng, B.-Y. Chen, and D.-N. Yang. Gausssense: Attachable stylus sensing using magnetic sensor grid. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST)*, pages 319–326, Cambridge, MA, USA, October 2012.

V. Schlageter, P.-A. Besse, R. Popovic, and P. Kucera. Tracking system with five degrees of freedom using a 2D-array of hall sensors and a permanent magnet. *Sensors and Actuators A: Physical*, 92(1):37–42, 2001.

J. T. Sherman, J. K. Lubkert, R. S. Popovic, and M. R. DiSilvestro. Characterization of a novel magnetic tracking system. *IEEE Transactions on Magnetics*, 43(6): 2725–2727, 2007.

E. Stathopoulos, V. Schlageter, B. Meyrat, Y. Ribaupierre, and P. Kucera. Magnetic pill tracking: a novel non-invasive tool for investigation of human digestive motility. *Neurogastroenterology & Motility*, 17(1):148–154, 2005.

N. Wahlström and F. Gustafsson. Magnetometer modeling and validation for tracking metallic targets. *IEEE Transactions on Signal Processing*, 62(3):545–556, 2014.

N. Wahlström and F. Gustafsson. Tracking position and orientation of magnetic objects using magnetometer networks. *IEEE Transactions on Signal Processing*, 2015. Submitted.

X. Wang, M.-H. Meng, and C. Hu. A localization method using 3-axis magnetoresistive sensors for tracking of capsule endoscope. In *Proceedings of the 8th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*, pages 2522–2525, New York, NY, USA, August 2006.

S. Yabukami, H. Kikuchi, M. Yamaguchi, K. Arai, K. Takahashi, A. Itagaki, and N. Wako. Motion capture system of magnetic markers using three-axial magnetic field sensor. *IEEE Transactions on Magnetics*, 36(5):3646–3648, 2000.

S. Yabukami, H. Kanetaka, N. Tsuji, A. Itagaki, M. Yamaguchi, K. Arai, and H. Mitani. A new tracking system of jaw movement using two magnets. *IEEE Transactions on Magnetics*, 38(5):3315–3317, 2002.

S. Yabukami, K. Arai, K. Arai, and S. Tsuji. Magnetic wireless motion capturing system and its application for jaw tracking system and 3D computer input device. *Journal of Magnetics*, 8(1):70–73, 2003.

Paper B

Classification of Driving Direction in Traffic Surveillance Using Magnetometers

Authors: Niklas Wahlström, Roland Hostettler, Fredrik Gustafsson and Wolfgang Birk

Edited version of the paper:

N. Wahlström, R. Hostettler, F. Gustafsson, and W. Birk. Classification of driving direction in traffic surveillance using magnetometers. *IEEE Transactions on Intelligent Transportation Systems*, 15(4):1405–1418, 2014.

Early version of this work was presented in:

N. Wahlström, R. Hostettler, F. Gustafsson, and W. Birk. Rapid classification of vehicle heading direction with two-axis magnetometer. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3385–3388, Kyoto, Japan, March 2012b.

Classification of Driving Direction in Traffic Surveillance Using Magnetometers

Niklas Wahlström[⋆], Roland Hostettler[†], Fredrik Gustafsson[⋆] and Wolfgang Birk[†]

[⋆]Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
{nikwa,fredrik}@isy.liu.se

[†]Division of Systems and Interaction
Luleå University of Technology
SE-971 87, Luleå, Sweden
{rolhos,wolfgang}@ltu.se

Abstract

Traffic monitoring using low-cost 2-axis magnetometers is considered. Though detection of metallic vehicles is rather easy, detecting the driving direction is more challenging. We propose a simple algorithm based on a nonlinear transformation of the measurements, which is simple to implement in embedded hardware. A theoretical justification is provided, and the statistical properties of the test statistic are presented in closed form. The method is compared to the standard likelihood ratio test on both simulated data and real data from field tests, where very high detection rates are reported, despite the presence of sensor saturation, measurement noise and near-field effects of the magnetic field.

1 Introduction

Traffic counting along particular roads is done either manually or electronically for the purpose of road improvement in the long term or re-routing traffic on the shorter time scale. The electronic devices used today include piezo-electric sensors or inductive loops under the road surface or pneumatic tubes on the road surface. Newer developments include radar, infrared light beams or cameras. For collecting road statistics, pneumatic tubes are today the most common solution. Recently the safety aspect of workers deploying the tubes has been in focus Wood et al. (2011). Another drawback of this solution is the short life-length of the tubes, which can be as short as 48 hours (Federal Highway Administration, 2013).

Magnetometers deployed along the road-side or hidden in the lane markers is a promising alternative, since they are cheap and small (Klein et al., 2006). Compared to pneumatic tubes, solutions using magnetometers have a longer lifetime since they are not subject to the same amount of wear and do not expose personnel to traffic when mounting and dismounting, hence improving workplace safety.

The magnetometer is preferably part of a wireless sensor network (WSN), where

sensor data from several magnetometers are transmitted to nearby nodes for a centralized or decentralized implementation of detection and classification algorithms (Birk et al., 2009; Cheung and Varaiya, 2007; Chinrungrueng et al., 2010). However, such sensor nodes also bring certain challenges. Generally, the energy budget is limited as the units are powered by batteries and/or solar panels (Girban and Popa, 2010). Furthermore, computational resources are scarce for reasons such as power saving (e.g. duty-cycling of the computations or low-power processors) or sharing of the microcontroller between different tasks (measuring, computing, communication, etc.), see, for example, Giannecchini et al. (2004) or Yu and Prasanna (2005). Thus, it is very important that the computation time for each task is reduced to a minimum, which emphasizes the need for low-complexity data processing algorithms.

One of the quantities of interest for road administrations, urban planners, or traffic management centers is traffic flow and, associated with that, the driving direction. Consider, for example, a single sensor monitoring a two-way two-lane highway. In order to be able to quantify the traffic volume on the individual lanes, the driving direction is crucial or, if one sensor for each lane is used, one would like to exclude vehicles on the farther lane. Thus, the traffic volume that is normally measured by a simple detector can be analyzed more thoroughly and better conclusions for future measures such as road planning can be drawn. In a similar setting, the driving direction can be used for detecting vehicles driving in the wrong lane, for example while overtaking. This information in turn can be used in order to warn upcoming traffic about a possible hazard in a cooperative collision warning system (Hegeman et al., 2009). A third application where the driving direction is of utmost importance is the detection of wrong-way drivers. Wrong-way drivers are a very hazardous threat to other road users and can cause serious accidents (Moler, 2002). Particularly on freeways, wrong-way drivers can cause serious head-on collisions and in 2010, wrong-way drivers accounted for 3.1 % of fatal crashes in the USA, causing 1,356 fatalities (National Highway Traffic Safety Administration, 2010). Thus, a system for detecting this kind of driving behavior can be of much help for authorities to detect vehicles driving in the wrong direction early on and warning other road users about the threat.

There are a number of different methods using various types of sensors for estimating the driving direction of a vehicle available today. One straight-forward approach is to use imaging sensors such as cameras or infrared laserscanners for tracking vehicles (Zhang and Forshaw, 1997; Goyat et al., 2006). Such sensors can provide very rich information and the driving direction of vehicles can be determined based on the estimated vehicle trajectory. However, challenging weather or illumination conditions degrade the performance significantly. One way of addressing this problem is to fuse the visual data with another type of sensor as proposed in Pucher et al. (2010) where a combination of a camera and a microphone array was used. While these approaches are viable and used in practice, they are not well suited for large scale deployment (for example at every freeway ramp) due to their requirements. Solutions more tailored for a system following the requirements stated in the beginning are often based on two spatially separated sensors (Cheung and Varaiya, 2007; Mimbela and Klein, 2000). However,

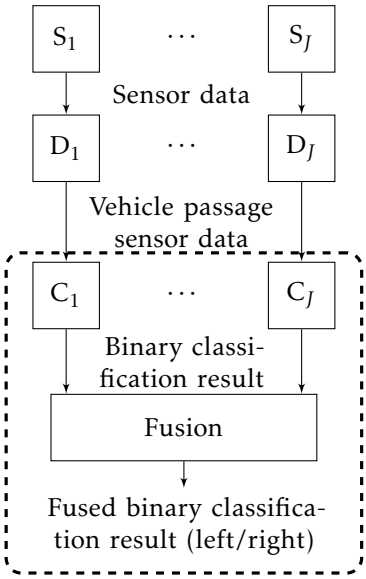


Figure 1: *Illustration of the signal flow. The sensor data from J sensors (S_1 through S_J) is used for detection (D_1 through D_J). Each sensor classifies the driving direction individually (C_1 through C_J) and a last, optional fusion step can be included to fuse these classification results. The dashed box indicates the parts of the system considered in this paper.*

the need for a second sensor can increase the cost considerably (up to twice the cost compared to only one sensor in the worst case) and introduces other challenges such as vehicle re-identification (Sun et al., 2004; Kwong et al., 2009) or the requirement of communication between the sensors (Pantazis and Vergados, 2007). Each of these activities will inevitably increase the energy consumption, which is a limited resource. Furthermore, a system based on only one single sensor is presumably more reliable since it does not depend on any second sensor that could break down.

In contrast to these approaches, this paper introduces a method for classifying the driving direction of a vehicle in a fast and efficient way addressing the initially stated requirements of a wireless sensor node. The method is based on one single magnetometer which measures magnetic field distortions induced by vehicles in its vicinity. Intuitively, extracting size and speed from this signal is rather straightforward. The basic principle is that the peak value of the measured signal is related to the size of the car, and the duration of the response is related to the speed of the vehicle.

However, obtaining the driving direction requires more physical insight about the signal and this problem has not been addressed before. In its simplest form, the proposed driving direction classification method only comprises a difference of two inner products of two vectors as it was first shown in (Wahlström et al., 2012). This work is an extension of these findings and provides a more thorough

statistical analysis as well as the evaluation of the classifier using more simulation and real measurement data. Specifically, the contributions of this work are:

- Extended version of the driving direction classification method presented in Wahlström et al. (2012) including an analysis of its statistical properties based on one single sensor.
- Verification of the proposed method using simulations as well as real measurement data.
- Comparison of the proposed method with a standard likelihood classification scheme.
- A sensor fusion strategy for multi-sensor scenarios.

In order to implement a complete system, also the detection of vehicles has to be considered. This can be accomplished by using adaptive thresholds Cheung and Varaiya (2007) which is proven to be both robust and computationally cheap. However, in this paper, we will only consider the classification as well as the optional fusion step and assume that the detection and association problems are already solved (Figure 1).

The outline of this paper is as follows. The signal model describing the magnetic field distortion caused by a vehicle is presented in Section 2. The proposed classifier and its statistical properties are given in Section 3 and a likelihood classifier is presented in Section 4, which will be used as a comparison to the proposed algorithm. The properties are verified and discussed by using Monte Carlo simulations in Section 5 and finally, the method is applied to real data in Section 6, followed by conclusions in Section 7.

2 Signal Model

The magnetometer signal induced by a metallic vehicle contains rich information which depends on both, the target trajectory as well as target specific parameters. A typical signal is displayed in Figure 2. In this work, we are only interested in determining the driving direction of the target. Consequently, the method should be insensitive to other quantities such as velocity, distance between the sensor and the trajectory, magnetic signature, and target extension.

One way of solving the problem is to approximately model the target as a magnetic dipole. This approximation holds if the distance between the target and the sensor is large in comparison to the characteristic magnetic length of the target (Jackson, 1998) which is typically in the range of 1 m to 2 m for passenger cars and 4 m and more for larger vehicles such as busses and trucks. This gives raise to a magnetic dipole field $\mathbf{h}(t)$ expressed as

$$\mathbf{h}(t) = \frac{3(\mathbf{r}(t) \cdot \mathbf{m})\mathbf{r}(t) - \|\mathbf{r}(t)\|^2 \mathbf{m}}{\|\mathbf{r}(t)\|^5}, \tag{1a}$$

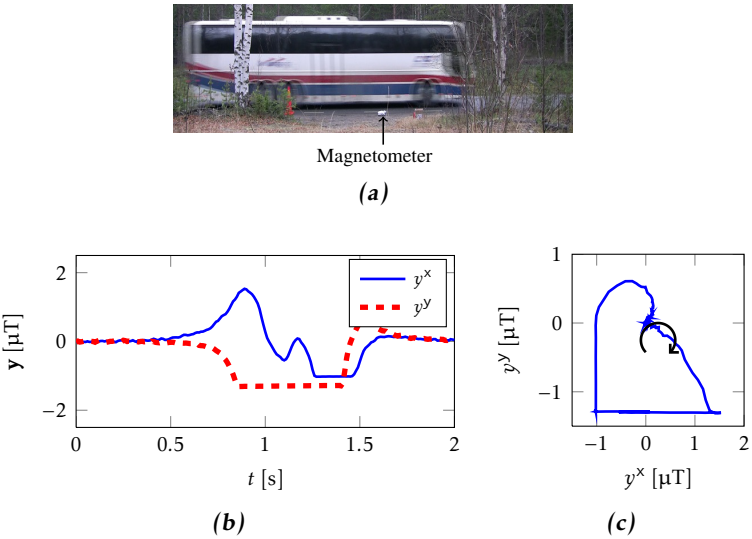


Figure 2: *A metallic vehicle (a) gives rise to a magnetic field distortion (b). The driving direction can be revealed by estimating the rotation direction of the magnetic field components (c).*

where $\mathbf{r}(t) = \begin{bmatrix} r^x(t) & r^y(t) & r^z(t) \end{bmatrix}^T$ is the position of the target relative to the sensor and $\mathbf{m} = \begin{bmatrix} m^x & m^y & m^z \end{bmatrix}^T$ is the magnetic dipole moment, which can be considered as a target specific parameter (Wahlström et al., 2010). Two components of the magnetic field (1a) can then be measured with a 2-axis magnetometer

$$\begin{aligned} \mathbf{y}_k = \begin{bmatrix} y^x(kT) \\ y^y(kT) \end{bmatrix} &= \begin{bmatrix} h^x(kT) \\ h^y(kT) \end{bmatrix} + \begin{bmatrix} e^x(kT) \\ e^y(kT) \end{bmatrix} \\ &= \bar{\mathbf{h}}_k + \mathbf{e}_k, \end{aligned} \tag{1b}$$

where T is the sampling time, k denotes the sampling instant, $\bar{\mathbf{h}}_k$ is a 2×1 vector containing the x- and y-components (the first two components) of the 3×1 vector $\mathbf{h}_k = \mathbf{h}(kT)$, and $\mathbf{e}_k$ is measurement noise assumed to be independent, identically distributed, zero mean white Gaussian noise of the form

$$\mathbf{e}_k \sim \mathcal{N}(0, \sigma^2 I_2), \tag{1c}$$

where I_2 is the 2×2 identity matrix. Furthermore, the following vector notation will be used

$$Y_{m:n}^{\alpha} = \begin{bmatrix} y_m^{\alpha} & y_{m+1}^{\alpha} & \cdots & y_n^{\alpha} \end{bmatrix}^T, \tag{2}$$

where $\alpha \in \{x, y\}$.

The model in (1) can now be used to classify the driving direction of the vehicle. The two hypotheses of this binary classification problem will be denoted

$\mathcal{H}_L$ and $\mathcal{H}_R$ representing that the vehicle passes the sensor from the left or from the right, respectively. One way of doing this is by estimating the unknowns $\mathbf{r}(t)$ and $\mathbf{m}$ from the measurement of $\mathbf{y}_k$ and extract the direction information from the estimated trajectory $\hat{\mathbf{r}}(t)$. This can either be done in a batch approach where a whole data batch is used at once or through object tracking using, for example, a Kalman or particle filter as it has been done in Birsan (2004); Rakijas et al. (2001) and Wahlström and Gustafsson (2014). However, this is a nonlinear problem and convergence to a global optimum is not guaranteed. Furthermore, if the target is close to the sensor, a higher order model including more parameters is needed to describe the signal accurately, for example by including higher order moments of the magnetic field or by modeling the target as a grid of dipoles (Wynn, 1999). Unfortunately, the computational cost of the corresponding estimation problem would in the worst case grow exponentially with the number of parameters (Boyd and Vandenberghe, 2004).

Instead, an alternative method based on computing the cross-correlation between the different channels of the measurement is suggested and evaluated in this work. Furthermore, the proposed method will be compared to a likelihood ratio test based on the dipole model (1). This test can be seen as a common practice procedure and is often used in detection and classification problems in all kinds of disciplines and is thus used as a benchmark (Kay, 1998; Root, 1970).

3 Correlation-based Classifier

3.1 Method and Algorithm

It has been shown in Wahlström et al. (2012) that the driving direction information can be obtained by computing the rotation direction of the magnetic field vector components, which is illustrated in Figure 2c. Specifically, the sign of the area

$$f^\star = \int \begin{vmatrix} h^{\mathrm{x}} & dh^{\mathrm{x}} \\ h^{\mathrm{y}} & dh^{\mathrm{y}} \end{vmatrix} = \int \begin{vmatrix} h^{\mathrm{x}}(t) & dh^{\mathrm{x}}(t)/dt \\ h^{\mathrm{y}}(t) & dh^{\mathrm{y}}(t)/dt \end{vmatrix} dt \tag{3}$$

is the same as the sign of the area spanned by the position vector $\mathbf{r}(t)$. Using discrete time measurements, (3) can be approximated by

$$f_1 = \sum_{k=1}^{N-1} \begin{vmatrix} h_k^{\mathrm{x}} & (h_{k+1}^{\mathrm{x}} - h_k^{\mathrm{x}})/T \\ h_k^{\mathrm{y}} & (h_{k+1}^{\mathrm{y}} - h_k^{\mathrm{y}})/T \end{vmatrix} T = \sum_{k=1}^{N-1} (h_k^{\mathrm{x}} h_{k+1}^{\mathrm{y}} - h_k^{\mathrm{y}} h_{k+1}^{\mathrm{x}}), \tag{4}$$

which corresponds to the sum of the triangles spanned by two adjacent samples of the trajectory, see Figure 3a. Note that since all the triangles are completely enclosed in the true trajectory, f_1 systematically underestimates the true area $f^\star$. Using (4) and the measurement model (1b) the proposed correlation-based driving direction classifier is summarized in Algorithm 1, which is an extension of the earlier results in (Wahlström et al., 2012). Note that using the vector notation in (2), subtractions and inner products of vectors are used in order to calculate (5)

Algorithm 1 Correlation-based driving direction classification

1. Calculate the test statistic

$$\hat{f}_p = \frac{1}{p}\left((Y^{\mathrm{x}}_{1:N})^{\mathsf{T}} Y^{\mathrm{y}}_{(1+p:N+p)} - (Y^{\mathrm{y}}_{1:N})^{\mathsf{T}} Y^{\mathrm{x}}_{(1+p):N+p}\right) \tag{5a}$$

   with $\mathbf{Y}^{\alpha}_{m:n}$ as defined in (2) and p the correlation lag (parameter).

2. Determine the driving direction by testing the sign of (5a)

$$\hat{f}_p \underset{\mathcal{H}_L}{\overset{\mathcal{H}_R}{\gtrless}} 0. \tag{5b}$$

3. Estimate the variance of the test statistic

$$\begin{aligned}\hat{\sigma}^2_{\hat{f}_p} = \frac{\sigma^2}{p^2}\Big(&(Y^{\mathrm{x}}_{1+p:N+p} - Y^{\mathrm{x}}_{1-p:N-p})^{\mathsf{T}}(Y^{\mathrm{x}}_{1+p:N+p} - Y^{\mathrm{x}}_{1-p:N-p}) \\ &+ (Y^{\mathrm{y}}_{1+p:N+p} - Y^{\mathrm{y}}_{1-p:N-p})^{\mathsf{T}}(Y^{\mathrm{y}}_{1+p:N+p} \quad - Y^{\mathrm{y}}_{1-p:N-p})\Big) - \frac{2N}{p^2}\sigma^4.\end{aligned} \tag{5c}$$

4. Estimate the error probability

$$\hat{P}_E = \frac{1}{2}\operatorname{erfc}\left(\frac{|\hat{f}_p|}{\sqrt{2}\hat{\sigma}_{\hat{f}_p}}\right). \tag{5d}$$

which is very beneficial for efficient implementation. Also, the algorithm is only parametrized by one single averaging parameter p introduced in (5a) which can be chosen as described in Section 3.3 below. The properties of the proposed algorithm are derived in the following section.

3.2 Properties

Given the algorithm as introduced in Algorithm 1, its properties are shown and derived in this section. Note that the assumption

$$\bar{\mathbf{h}}_k = \mathbf{0} \quad \text{for} \quad k \leq 0 \ \vee \ k > N, \tag{6}$$

that is, that the magnetic signal has decayed to zero within the given window of N samples (however, $\mathbf{e}_k$ is non-zero) is made throughout the remainder of the paper.

Correlation with Lag $p = 1$

The natural choice for the averaging parameter p introduced in (5a) is to use $p = 1$ since it is essentially directly replacing the true magnetic field vector $\bar{\mathbf{h}}_k$ in (4) with its noisy counterpart $\mathbf{y}_k$ as

$$\hat{f}_1 = \sum_{k=1}^{N} (y_k^{\mathrm{x}} y_{k+1}^{\mathrm{y}} - y_k^{\mathrm{y}} y_{k+1}^{\mathrm{x}}), \tag{7}$$

which can be interpreted as being the difference of the cross-correlations between y^{x} and y^{y} with lag 1 and –1 respectively. Since the measurement noise is assumed to be zero mean and i.i.d. (1c), it can be shown that the estimator (7) is unbiased

$$\begin{aligned} \mathbb{E}\left[\hat{f}_1\right] &= \mathbb{E}\left[\sum_{k=1}^{N} (y_k^{\mathrm{x}} y_{k+1}^{\mathrm{y}} - y_k^{\mathrm{y}} y_{k+1}^{\mathrm{x}})\right] = \sum_{k=1}^{N} \mathbb{E}\left[y_k^{\mathrm{x}} y_{k+1}^{\mathrm{y}} - y_k^{\mathrm{y}} y_{k+1}^{\mathrm{x}}\right] \\ &= \sum_{k=1}^{N} h_k^{\mathrm{x}} h_{k+1}^{\mathrm{y}} - h_k^{\mathrm{y}} h_{k+1}^{\mathrm{x}} = f_1. \end{aligned} \tag{8}$$

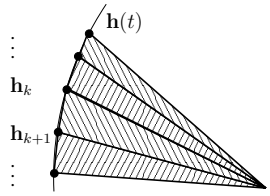
Further, the variance of (7) is given by

$$\begin{aligned} \sigma_{\hat{f}_1}^2 &\triangleq \operatorname{Var}\left(\hat{f}_1\right) = \mathbb{E}\left[(\hat{f}_1 - \mathbb{E}[\hat{f}_1])^2\right] \\ &= \mathbb{E}\left[\left(\sum_{k=1}^{N} (h_k^{\mathrm{x}} + e_k^{\mathrm{x}})(h_{k+1}^{\mathrm{y}} + e_{k+1}^{\mathrm{y}}) - (h_k^{\mathrm{y}} + e_k^{\mathrm{y}})(h_{k+1}^{\mathrm{x}} + e_{k+1}^{\mathrm{x}}) - (h_k^{\mathrm{x}} h_{k+1}^{\mathrm{y}} - h_k^{\mathrm{y}} h_{k+1}^{\mathrm{x}})\right)^2\right] \\ &= \mathbb{E}\left[\left(\sum_{k=1}^{N} h_k^{\mathrm{x}} e_{k+1}^{\mathrm{y}} + h_{k+1}^{\mathrm{y}} e_k^{\mathrm{x}} - h_k^{\mathrm{y}} e_{k+1}^{\mathrm{x}} - h_{k+1}^{\mathrm{x}} e_k^{\mathrm{y}} + e_{k+1}^{\mathrm{y}} e_k^{\mathrm{x}} - e_{k+1}^{\mathrm{x}} e_k^{\mathrm{y}}\right)^2\right]. \end{aligned} \tag{9}$$

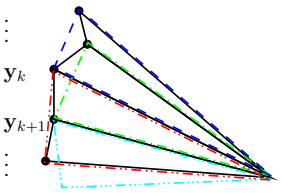
Analyzing the sum in (9) it can be seen that every e_k^{α} appears twice in the whole sum, once scaled by h_{k+1}^{β} and once by $-h_{k-1}^{\beta}$ (where the superscript β denotes the in-plane component perpendicular to α). Making use of this and (6) yields

$$\begin{aligned} \sigma_{\hat{f}_1}^2 &= \mathbb{E}\left[\left(\sum_{k=1}^{N} (h_{k+1}^{\mathrm{y}} - h_{k-1}^{\mathrm{y}}) e_k^{\mathrm{x}} - (h_{k+1}^{\mathrm{x}} - h_{k-1}^{\mathrm{x}}) e_k^{\mathrm{y}} + e_{k+1}^{\mathrm{y}} e_k^{\mathrm{x}} - e_{k+1}^{\mathrm{x}} e_k^{\mathrm{y}}\right)^2\right] \\ &= \sigma^2 \sum_{k=1}^{N} \left\| (\bar{\mathbf{h}}_{k+1} - \bar{\mathbf{h}}_{k-1}) \right\|^2 + 2N\sigma^4. \end{aligned} \tag{10}$$

From (10) it is seen that the variance is increased by the norm of the (approximate) gradient of the magnetic field vector $\nabla \mathbf{h}_k \approx \left\| (\bar{\mathbf{h}}_{k+1} - \bar{\mathbf{h}}_{k-1}) \right\| / 2$ as well as the window length. Finally, note that the distribution of $\hat{f}_1$ is given by $\hat{f}_1 \sim \mathcal{N}(f, \sigma_{\hat{f}_1}^2)$ as $N \to \infty$ as shown in Proposition 1 in Appendix A using $p = 1$.



(a) The quantity f_1 in (4) corresponds to the sum of the triangles spanned by two adjacent samples of the trajectory.



(b) The estimator $\hat{f}_1$ sums over the black solid line triangles whereas $\hat{f}_2$ averages over the colored dashed triangles.

Figure 3: *Geometrical interpretation of the estimators.*

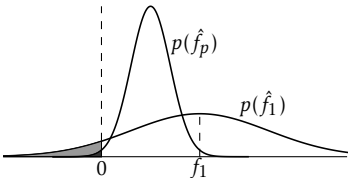


Figure 4: *Comparison of the probability density functions of the estimators $\hat{f}_1$ and $\hat{f}_p$. The expected value of $\hat{f}_p$ is biased towards zero compared to $\hat{f}_1$, however, the error probability (shaded areas) is much smaller.*

Correlation with Lag $p > 1$

As shown above, the variance for $p = 1$ scales badly if the noise is large since the second term in (10) scales with σ^4. It is thus desirable to reduce this effect. This can be achieved by using an averaging estimator with lag $p > 1$ in order to reduce the noise sensitivity. Instead of calculating the triangular area of two neighboring measurement points k and $k+1$ on the trajectory, larger area segments between the points k and $k+p$ are considered (Figure 3b). This yields the cross-correlation estimator with lag p

$$\hat{f}_p = \frac{1}{p}\sum_{k=1}^{N}(y_k^{\mathrm{x}} y_{k+p}^{\mathrm{y}} - y_k^{\mathrm{y}} y_{k+p}^{\mathrm{x}}) \tag{11}$$

by conducting similar calculations as in (8) it is straightforward to show that for $p \neq 1$

$$\mathbb{E}\left[\hat{f}_p\right] = \underbrace{\frac{1}{p}\sum_{k=1}^{N} h_k^{\mathrm{x}} h_{k+p}^{\mathrm{y}} - h_k^{\mathrm{y}} h_{k+p}^{\mathrm{x}}}_{\triangleq f_p} \neq f_1 \tag{12}$$

and (11) is thus a biased estimator of (4). However, since only the sign of f_1 is of interest, this is acceptable. The variance is given by

$$\begin{aligned}
\sigma^2_{\hat{f}_p} &\triangleq \mathrm{Var}(\hat{f}_p) = \mathbb{E}\Big[(\hat{f}_p - \mathbb{E}[\hat{f}_p])^2\Big] \\
&= \mathbb{E}\left[\left(\frac{1}{p}\sum_{k=1}^{N}(h_k^{\mathrm{x}} + e_k^{\mathrm{x}})(h_{k+p}^{\mathrm{y}} + e_{k+p}^{\mathrm{y}}) - (h_k^{\mathrm{y}} + e_k^{\mathrm{y}})(h_{k+p}^{\mathrm{x}} + e_{k+p}^{\mathrm{x}}) - (h_k^{\mathrm{x}} h_{k+p}^{\mathrm{y}} - h_k^{\mathrm{y}} h_{k+p}^{\mathrm{x}})\right)^2\right] \\
&= \mathbb{E}\left[\left(\frac{1}{p}\sum_{k=1}^{N} h_k^{\mathrm{x}} e_{k+p}^{\mathrm{y}} + h_{k+p}^{\mathrm{y}} e_k^{\mathrm{x}} - h_k^{\mathrm{y}} e_{k+p}^{\mathrm{x}} - h_{k+p}^{\mathrm{x}} e_k^{\mathrm{y}} + e_{k+p}^{\mathrm{y}} e_k^{\mathrm{x}} - e_{k+p}^{\mathrm{x}} e_k^{\mathrm{y}}\right)^2\right].
\end{aligned} \tag{13}$$

As in (9)-(10), the coefficients for e_k^α can be grouped which gives

$$\begin{aligned}\sigma^2_{\hat{f}_p} &= \mathbb{E}\left[\left(\frac{1}{p}\sum_{k=1}^{N}(h^{\mathrm{y}}_{k+p}-h^{\mathrm{y}}_{k-p})e^{\mathrm{x}}_k-(h^{\mathrm{x}}_{k+p}-h^{\mathrm{x}}_{k-p})e^{\mathrm{y}}_k+e^{\mathrm{y}}_{k+p}e^{\mathrm{x}}_k-e^{\mathrm{x}}_{k+p}e^{\mathrm{y}}_k\right)^2\right] \\ &= \frac{\sigma^2}{p^2}\sum_{k=1}^{N}\left\|\bar{\mathbf{h}}_{k+p}-\bar{\mathbf{h}}_{k-p}\right\|^2+\frac{2N}{p^2}\sigma^4.\end{aligned} \tag{14}$$

From (14) it is seen that the variance is greatly reduced compared to (10). The second term scales with $1/p^2$ compared to the second term in (10). As for the unbiased estimator $\hat{f}_1$, the distribution of $\hat{f}_p$ converges to the normal distribution $\hat{f}_p \sim \mathcal{N}(f_p, \sigma^2_{\hat{f}_p})$ where f_p is the mean value as illustrated in (12) (see Proposition 1 in Appendix A).

The averaging effect is illustrated in Figure 4. Assuming that the true value f_1 is positive, the expected value of the averaging estimator is moved towards zero due to the bias. However, the averaging reduces the variance and thus the total error probability (the shaded area under the probability density function up to zero) is reduced significantly.

In practice, it is of interest to estimate the error probability coupled to the estimate $\hat{f}_p$ obtained from (11). For that reason, the variance of (11), which is given in (14), is of interest. Noting that

$$\mathbb{E}\left[\left\|\mathbf{y}_{k+p}-\mathbf{y}_{k-p}\right\|^2\right]=\mathbb{E}\left[(y^{\mathrm{x}}_{k+p}-y^{\mathrm{x}}_{k-p})^2+(y^{\mathrm{y}}_{k+p}-y^{\mathrm{y}}_{k-p})^2\right]$$

and letting $z_k^\alpha = y^\alpha_{k+p} - y^\alpha_{k-p}$ it follows that $z_k^\alpha \sim \mathcal{N}(h^\alpha_{k+p} - h^\alpha_{k-p}, 2\sigma^2)$. Thus,

$$\mathbb{E}\left[\left(z^{\mathrm{x}}_k\right)^2+\left(z^{\mathrm{y}}_k\right)^2\right]=(h^{\mathrm{x}}_{k+p}-h^{\mathrm{x}}_{k-p})^2+2\sigma^2+(h^{\mathrm{y}}_{k+p}-h^{\mathrm{y}}_{k-p})^2+2\sigma^2$$

and finally

$$\mathbb{E}\left[\left\|\mathbf{y}_{k+p}-\mathbf{y}_{k-p}\right\|^2\right]=\left\|\bar{\mathbf{h}}_{k+p}-\bar{\mathbf{h}}_{k-p}\right\|^2+4\sigma^2. \tag{15}$$

Using (14) and (15) we can then estimate $\mathrm{Var}(\hat{f}_p)$ as follows

$$\begin{aligned}\hat{\sigma}^2_{\hat{f}_p} &= \frac{\sigma^2}{p^2}\sum_{k=1}^{N}\left(\|\mathbf{y}_{k+p}-\mathbf{y}_{k-p}\|^2-4\sigma^2\right)+\frac{2N}{p^2}\sigma^4 \\ &= \frac{\sigma^2}{p^2}\sum_{k=1}^{N}\|\mathbf{y}_{k+p}-\mathbf{y}_{k-p}\|^2-\frac{2N}{p^2}\sigma^4.\end{aligned} \tag{16}$$

The probability that the sign of the estimated $\hat{f}_p$ is wrong compared to the sign of f_p is given by

$$\Pr\left(\mathrm{sgn}(\hat{f}_p)\neq\mathrm{sgn}(f_p)\right)=\frac{1}{2}\mathrm{erfc}\left(\frac{|f_p|}{\sqrt{2}\sigma_{\hat{f}_p}}\right). \tag{17}$$

Since neither the true f_p nor $\sigma^2_{\hat{f}_p}$ are known, the estimated values $\hat{\hat{f}}_p$ and $\hat{\sigma}^2_{\hat{f}_p}$ can be used instead. The estimated error probability (17) then becomes

$$\hat{P}_E = \frac{1}{2}\mathrm{erfc}\left(\frac{|\hat{\hat{f}}_p|}{\sqrt{2}\hat{\sigma}_{\hat{f}_p}}\right), \tag{18}$$

which can be evaluated numerically. Note, however, that (18) has a slightly different meaning than (17). It indicates the probability of f_p having a different sign compared to the given $\hat{\hat{f}}_p$.

3.3 Parameter Tuning

The lag p introduced in (11) will improve the classification result as explained in Section 3.2. Our objective is to choose a value p which minimizes the overall estimated error probability (18). In theory this could be done for each detection separately. However, that would require a non-linear search in the parameter p for each detection, which does not meet the needs for a computationally efficient implementation. Instead we will minimize the mean of the estimated error probabilities from a training set of estimation data $l = 1, 2, \ldots, L$ and use this value p afterwards. Given a set of estimation data $(Y_{1:N})_{1:L}$ we compute p as

$$p = \arg\min_p \frac{1}{L}\sum_{l=1}^{L}\hat{P}_{E,l} = \arg\min_p \sum_{l=1}^{L}\mathrm{erfc}\left(\frac{|\hat{\hat{f}}_{p,l}|}{\sqrt{2}\hat{\sigma}_{\hat{f}_{p,l}}}\right). \tag{19}$$

After finding this value p, all future classifications can be performed by using Algorithm 1 and the given value for p.

3.4 Sensor Fusion

Information from multiple sensors can be fused together in order to arrive at a joint-classification of multiple sensors. The fusion rule (49) for Bernoulli random variables as derived in Appendix B is used in order to reach a joint decision of the driving direction of J sensors as follows. Let p_j be the probability that the car is passing the sensor from left to right (hypothesis $\mathcal{H}_L$ true) which is given by

$$p_j = \Pr(\mathcal{H}_L|\hat{\hat{f}}_{p,j}, \hat{\sigma}_{\hat{f}_{p,j}}) = \frac{1}{2}\mathrm{erfc}\left(\frac{\hat{\hat{f}}_{p,j}}{\sqrt{2}\hat{\sigma}_{\hat{f}_{p,j}}}\right). \tag{20}$$

A Bernoulli random variable k with

$$p(k|p_j) = p_j^k(1-p_j)^{1-k} \quad \text{for} \quad k \in \{0, 1\} \tag{21}$$

can now be used to represent the probability of each hypothesis where the value $k = 0$ is assigned to $\mathcal{H}_R$ and $k = 1$ to $\mathcal{H}_L$. Finally, using (49) (see Appendix B)

and (21) yields the sensor fusion decision rule given by

$$\Pr(\mathcal{H}_L|\mathbf{p}) = \frac{\prod_{j=1}^{J} p_j}{\prod_{j=1}^{J}(1-p_j) + \prod_{j=1}^{J} p_j} \underset{\mathcal{H}_R}{\overset{\mathcal{H}_L}{\gtrless}} \frac{1}{2}, \tag{22}$$

where $\mathbf{p} = \begin{bmatrix} p_1 & \dots & p_J \end{bmatrix}^{\mathsf{T}}$. Equation (22) can also be rewritten as

$$\Pr(\mathcal{H}_L|\mathbf{p}) = \frac{\prod_{j=1}^{J} \Pr(\mathcal{H}_L|\hat{f}_{p,j})}{\prod_{j=1}^{J} \Pr(\mathcal{H}_R|\hat{f}_{p,j}) + \prod_{j=1}^{J} \Pr(\mathcal{H}_L|\hat{f}_{p,j})}, \tag{23}$$

which can be interpreted as the joint probability for all sensors indicating $\mathcal{H}_L$ at the same time, normalized by the sum of the same and its complementary event, that is, all sensors indicating $\mathcal{H}_R$ at the same time.

Notice that the classification is performed in a distributed manner by first computing the ratios p_j in each sensor according to (20), and then, these values are fused according to (23).

4 Likelihood Test

In order to benchmark the proposed driving direction classification algorithm proposed in Section 3, a generalized likelihood ratio test based on the measurement model (1) is derived in this section which is then compared to the proposed method in the following sections.

4.1 Single Sensor

Consider a vehicle passing the sensor with a constant velocity and constant lateral distance. The position $\mathbf{r}_k$ can be rewritten as

$$\mathbf{r}_k = \mathbf{r}(kT) = \begin{bmatrix} v(kT - t_{\text{CPA}}) \\ r^{\text{y}} \\ 0 \end{bmatrix}, \tag{24}$$

where v is the vehicle speed, t_{CPA} is the closest point of approach time, and r^{y} the lateral distance between the target and the sensor.

It can be safely assumed that most of the vehicles will adhere to the known speed limit v_{limit} for a given road and thus, vehicles passing the sensor can be classified according to the following two hypotheses:

$$\mathcal{H}_L: \quad \boldsymbol{\theta}_1^{\star} = \begin{bmatrix} v_{limit} & r_1^{\text{y}} \end{bmatrix}^{\mathsf{T}}, \tag{25a}$$

$$\mathcal{H}_R: \quad \boldsymbol{\theta}_2^{\star} = \begin{bmatrix} -v_{limit} & r_2^{\text{y}} \end{bmatrix}^{\mathsf{T}}, \tag{25b}$$

where $\boldsymbol{\theta}_i^{\star}$ is the hypothesis parametrization which is known. The lateral distances r_1^{y} and r_2^{y} are derived from the road geometry. For example, in a traffic counting

scenario, they would correspond to the distances to the closer and farther lane, respectively. On the other hand, when detecting wrong-way drivers on a freeway ramp, they would be equal. The hypothesis $\mathcal{H}_L$ corresponds to a vehicle passing the sensor from left to right and $\mathcal{H}_R$ to a vehicle passing the sensor from right to left.

The remaining parameters

$$\boldsymbol{\theta} = \begin{bmatrix} m^{\mathsf{x}} & m^{\mathsf{y}} & m^{\mathsf{z}} & t_{\mathrm{CPA}} \end{bmatrix}^{\mathsf{T}} \tag{26}$$

in (1a) are all unknown (opposite to $\boldsymbol{\theta}^{\star}$ which holds the known parameters determined as described above) and the measurement model can be rewritten as

$$\bar{\mathbf{h}}_k(\boldsymbol{\theta}_i^{\star}, \boldsymbol{\theta}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot \frac{\left(3\mathbf{r}_k\mathbf{r}_k^{\mathsf{T}} + \|\mathbf{r}_k\|^2 I_3\right)}{\|\mathbf{r}_k\|^5}\mathbf{m}, \tag{27}$$

where the position $\mathbf{r}_k$ is a function of the parameters r_i^{y}, v_i and t_{CPA} as given by (24). The measurement model is linear in the unknown vehicle dependent parameters $\mathbf{m}$ and non-linear in t_{CPA}. These have to be estimated before the actual likelihood test can be performed. This can be done by using a maximum likelihood estimator which yields a generalized likelihood ratio test (GLRT) (Kay, 1998).

The joint probability density function for all N vector samples is given by

$$\begin{aligned} p(Y_{1:N}; \boldsymbol{\theta}_i^{\star}, \boldsymbol{\theta}) &= \prod_{k=1}^{N} p(\mathbf{y}_k; \boldsymbol{\theta}_i^{\star}, \boldsymbol{\theta}) \\ &= \frac{1}{\left(2\pi\sigma^2\right)^{2N/2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{k=1}^{N} \|\mathbf{y}_k - \bar{\mathbf{h}}_k(\boldsymbol{\theta}_i^{\star}, \boldsymbol{\theta})\|_2^2 \right) \\ &= \frac{1}{\left(2\pi\sigma^2\right)^{2N/2}} \exp\left(-\frac{1}{2\sigma^2} \|Y_{1:N} - H_{1:N}(\boldsymbol{\theta}_i^{\star}, \boldsymbol{\theta})\|_2^2 \right), \end{aligned} \tag{28}$$

where the measurement samples are stacked as

$$Y_{m:n} = \begin{bmatrix} {Y_{m:n}^{\mathsf{x}}}^{\mathsf{T}} & {Y_{m:n}^{\mathsf{y}}}^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}}, \tag{29}$$

$Y_{m:n}^{\alpha}$ is as defined in (2), and equivalently for $H_{1:N}$. The maximum likelihood estimator (Kay, 1998) for the parameters $\boldsymbol{\theta}$ is then simply

$$\hat{\boldsymbol{\theta}}_i = \arg\max_{\boldsymbol{\theta}} \; p(Y_{1:N}; \boldsymbol{\theta}_i^{\star}, \boldsymbol{\theta}) \quad \text{for} \quad i = 1, 2. \tag{30}$$

where the estimate $\hat{\boldsymbol{\theta}}_i$ depends on the hypothesis $\boldsymbol{\theta}_i^{\star}$.

Once the estimation $\hat{\boldsymbol{\theta}}_i$ is obtained, the likelihood ratio can be calculated as

$$l = \frac{p(Y_{1:N}; \boldsymbol{\theta}_1^{\star}, \hat{\boldsymbol{\theta}}_1)}{p(Y_{1:N}; \boldsymbol{\theta}_2^{\star}, \hat{\boldsymbol{\theta}}_2)} \underset{\mathcal{H}_R}{\overset{\mathcal{H}_L}{\gtrless}} 1. \tag{31}$$

If $l > 1$, the hypothesis $\mathcal{H}_L$ is more likely to be true and $\mathcal{H}_R$ otherwise. Using (28) and (31), the log-likelihood ratio is given by

$$\begin{aligned}\lambda &= \log(l) \\ &= -\frac{1}{2\sigma^2}\left\|Y_{1:N} - H_{1:N}(\boldsymbol{\theta}_1^\star, \hat{\boldsymbol{\theta}}_1)\right\|_2^2 + \frac{1}{2\sigma^2}\left\|Y_{1:N} - H_{1:N}(\boldsymbol{\theta}_2^\star, \hat{\boldsymbol{\theta}}_2)\right\|_2^2 \underset{\mathcal{H}_R}{\overset{\mathcal{H}_L}{\gtrless}} 0\end{aligned} \tag{32}$$

and the decision rule becomes

$$\tilde{\lambda}_1 \underset{\mathcal{H}_R}{\overset{\mathcal{H}_L}{\gtrless}} \tilde{\lambda}_2 \tag{33}$$

with

$$\tilde{\lambda}_i = -\left\|Y_{1:N} - H_{1:N}(\boldsymbol{\theta}_i^\star, \hat{\boldsymbol{\theta}}_i)\right\|_2^2. \tag{34}$$

Note that the two test statistics $\tilde{\lambda}_1$ and $\tilde{\lambda}_2$ are easily calculated. However, the parameter estimation step to be executed still requires solving a (separable) non-linear problem. Hence, this method is not well tailored for implementation in systems with limited computational power since it requires an iterative solver which might not converge to the global optimum. Finally, the likelihood algorithm is summarized in Algorithm 2.

Algorithm 2 Generalized likelihood ratio test driving direction classification

1. Estimate the model parameters $\boldsymbol{\theta}$ as defined in (26) for the hypotheses $\{\mathcal{H}_L, \mathcal{H}_R\}$ using the maximum likelihood estimator

$$\hat{\boldsymbol{\theta}}_i = \arg\max_{\boldsymbol{\theta}} \; p(Y_{1:N}; \boldsymbol{\theta}_i^\star, \boldsymbol{\theta}) \quad \text{for} \quad i = 1, 2 \tag{35a}$$

   with $\boldsymbol{\theta}_i^\star$ as defined in (25).

2. Calculate the test statistic

$$\begin{aligned}\lambda &= \log(l) \\ &= -\frac{1}{2\sigma^2}\left\|Y_{1:N} - H_{1:N}(\boldsymbol{\theta}_1^\star, \hat{\boldsymbol{\theta}}_1)\right\|_2^2 + \frac{1}{2\sigma^2}\left\|Y_{1:N} - H_{1:N}(\boldsymbol{\theta}_2^\star, \hat{\boldsymbol{\theta}}_2)\right\|_2^2\end{aligned} \tag{35b}$$

   with $\mathbf{Y}_{1:N}$ and $\mathbf{H_{1:N}}(\boldsymbol{\theta}_i^\star, \hat{\boldsymbol{\theta}}_i)$ as defined in (29).

3. Determine the driving direction by testing the sign of (35b)

$$\lambda \underset{\mathcal{H}_R}{\overset{\mathcal{H}_L}{\gtrless}} 0. \tag{35c}$$

4.2 Sensor Fusion

If there is data from more than one sensor available, the likelihood test can take advantage of all this information in order to make a well-balanced decision. Let

$\boldsymbol{\theta}_j$ be the parameters and $Y_{1:N,j}$ be the measurements for the jth sensor. Then the joint PDF for all measurement is given by

$$p(\Upsilon; \boldsymbol{\theta}_i^\star, \Theta) = \prod_{j=1}^{J} p(Y_{1:N,j}; \boldsymbol{\theta}_i^\star, \boldsymbol{\theta}_j), \tag{36}$$

where $\Upsilon = \begin{bmatrix} Y_{1:N,1}{}^\mathsf{T} & Y_{1:N,2}{}^\mathsf{T} & \dots & Y_{1:N,J}{}^\mathsf{T} \end{bmatrix}^\mathsf{T}$ and $\Theta = \begin{bmatrix} \theta_1{}^\mathsf{T} & \theta_2{}^\mathsf{T} & \dots & \theta_J{}^\mathsf{T} \end{bmatrix}^\mathsf{T}$. The overall likelihood ratio and decision rule is then given by

$$l_J = \frac{p(\Upsilon; \boldsymbol{\theta}_1^\star, \hat{\boldsymbol{\Theta}}_1)}{p(\Upsilon; \boldsymbol{\theta}_2^\star, \hat{\boldsymbol{\Theta}}_2)} = \frac{\prod_{j=1}^{J} p(Y_{1:N,j}; \boldsymbol{\theta}_1^\star, \hat{\boldsymbol{\theta}}_{1,j})}{\prod_{j=1}^{J} p(Y_{1:N,j}; \boldsymbol{\theta}_2^\star, \hat{\boldsymbol{\theta}}_{2,j})} \underset{\mathcal{H}_R}{\overset{\mathcal{H}_L}{\gtrless}} 1, \tag{37}$$

which results in the log likelihood

$$\begin{aligned} \lambda_J &= \log(l_J) \\ &= \sum_{j=1}^{J} \frac{1}{2\sigma_j^2} \tilde{\lambda}_{1,j} - \sum_{j=1}^{J} \frac{1}{2\sigma_j^2} \tilde{\lambda}_{2,j} = \sum_{j=1}^{J} \frac{1}{2\sigma_j^2} \left(\tilde{\lambda}_{1,j} - \tilde{\lambda}_{2,j} \right) \underset{\mathcal{H}_R}{\overset{\mathcal{H}_L}{\gtrless}} 0. \end{aligned} \tag{38}$$

Finally, (38) can be rewritten as

$$\sum_{j=1}^{J} \frac{1}{\sigma_j^2} \tilde{\lambda}_{1,j} \underset{\mathcal{H}_R}{\overset{\mathcal{H}_L}{\gtrless}} \sum_{j=1}^{J} \frac{1}{\sigma_j^2} \tilde{\lambda}_{2,j}. \tag{39}$$

Notice that the classification is performed in a distributed manner by first computing the ratios $\tilde{\lambda}_{i,j}$ in each sensor according to (34), and then, these values are fused according to (39).

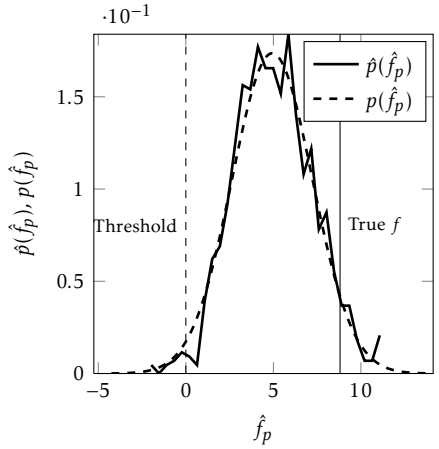
5 Simulation

Before applying the classifier derived in the previous section on real data, a simulation will be used to visualize and validate the properties of the proposed estimator. In the end of this section, the proposed classifier is also compared to the generalized likelihood test presented in Section 4.

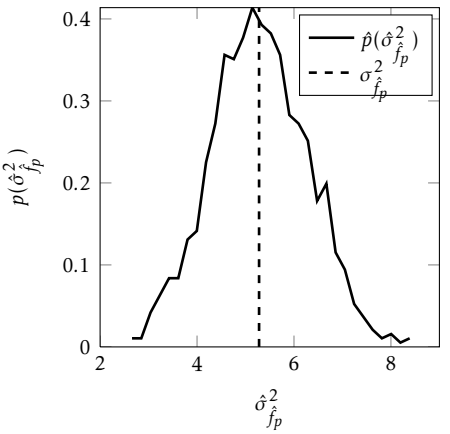
Consider a simulation setup with a vehicle heading in positive x-direction, starting at $\mathbf{r}_1 = \begin{bmatrix} -5 & 1 & 0 \end{bmatrix}^\mathsf{T}$ and ending at $\mathbf{r}_N = \begin{bmatrix} 5 & 1 & 0 \end{bmatrix}^\mathsf{T}$ divided into $N = 100$ data points in between. Furthermore, consider a magnetic dipole moment of $\mathbf{m} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^\mathsf{T}$. We will simulate this example with different levels of the signal to noise ratio (SNR), which is defined as

$$\text{SNR} = 10 \log_{10} \left(\frac{\frac{1}{N} \sum_{k=1}^{N} \|\bar{\mathbf{h}}_k\|^2}{\sigma^2} \right) \text{dB}, \tag{40}$$

where σ^2 is the variance of the measurement noise.



(a) *Empirical distribution $\hat{p}(\hat{f}_p)$ together with the theoretical distribution $p(\hat{f}_p)$.*



(b) *Empirical distribution $\hat{p}(\hat{\sigma}^2_{\hat{f}_p})$ together with the theoretical variance.*

Figure 5: *Simulation results of the estimator (a) and its estimated variance (b) for a scenario with SNR $= -10$ dB and $p = 15$.*

5.1 Estimate and Variance Estimate

In (12) and (14) expressions for the mean $f_p = \mathbb{E}\left[\hat{f}_p\right]$ and variance $\sigma^2_{\hat{f}_p} = \mathbb{E}\left[(\hat{f}_p - f_p)^2\right]$ of the estimators are given. These expressions are verified by performing 1,000 Monte Carlo simulations for the presented example with different noise realization for each run. The result is presented in Figure 5a with SNR $= -10$ dB and $p = 15$ together with the theoretical distribution of the estimator $\hat{f}_p \sim \mathcal{N}\left(f_p, \sigma^2_{\hat{f}_p}\right)$.

According to the result, the theoretical distribution corresponds well to the empirical one. Furthermore, note that the estimate is biased $\mathbb{E}\left[\hat{f}_p\right] \neq f_1$, as already stated in (12). We can also conclude that the Gaussian assumption of the estimator distribution is indeed valid.

Each sequence of data does not only provide us with the estimate (11), but also with an estimate of its variance (16). In Figure 5b, this estimated variance is compared with the true variance, using the same 1,000 Monte Carlo simulations as previously. According to this result, the variance estimate seems to be unbiased as expected from the derivation.

5.2 Dependency of P_E on SNR and p

In (17) a scheme for computing the error probability is proposed by knowing the true mean and variance of the estimator. This value has been compared with the actual error classification rate by performing 1,000 Monte Carlo simulations for

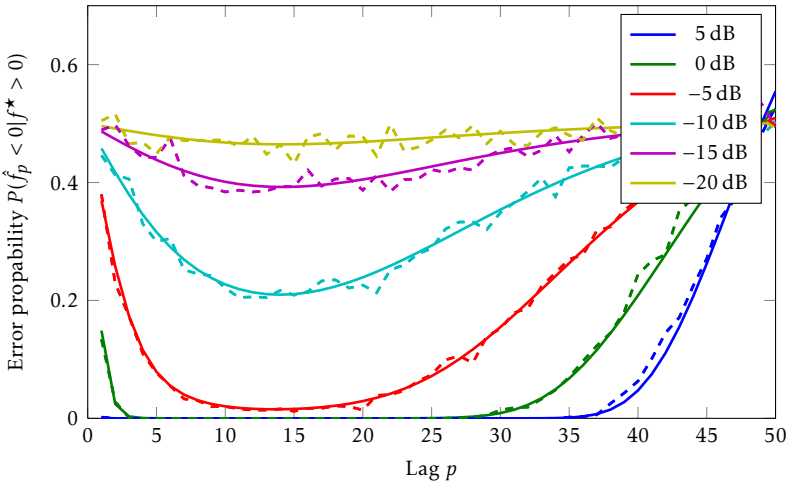


Figure 6: *Classification performance as a function of the lag p for different SNR. The solid line is the theoretical performance P_E according to (17) and the dashed line the average error probability of the 1,000 Monte Carlo simulations.*

different values of p and SNR. The result is provided in Figure 6 and the theoretical values display a good agreement with the simulations. Also the classification performance increases with higher SNR which is natural.

Furthermore, this result also shows the classification improvement of the cross-correlation method by choosing a lag $p > 1$. Also note, that for the chosen simulation scenario, there is an optimal $p \approx 15$ which is fairly independent of the SNR. However, this will depend on the magnetic moment of the vehicle $\mathbf{m}$, the trajectory $\mathbf{r}_k$, and the data length N.

5.3 Comparison with Likelihood Test

As a reference, the proposed classifier can be compared to the likelihood test presented in Section 4. Again, 1,000 Monte Carlo simulations with $p = 15$ for the correlation classifier and the true values for v_1, v_2, r_1^y and r_2^y for the likelihood test were run. Figure 7 shows the error rates for the two classifiers as functions of the SNR. As can be expected, both classifiers perform well for high SNRs , down to about −5 dB where the error rates start to increase until the point of "tossing a coin" somewhere below −20 dB is reached. It should be noted, however, that the correlation classifier requires an SNR of about 5 dB higher than the likelihood classifier in order to achieve the same classification rate. This is not very surprising since the likelihood test is expected to be the optimal test for this scenario since the likelihood test is performed under the same model and model parameter as have been used in the simulation. However, it will be shown that the likelihood test is more sensitive to violations of the model assumptions such as

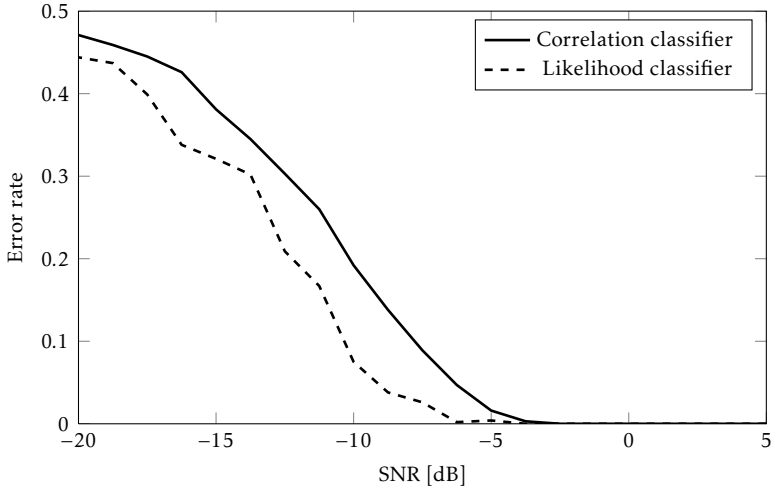


Figure 7: *Classification performance compared with the likelihood test as a function of the SNR.*

dipole model, speed, or trajectory, which were used to derive the GLRT. These results are presented in the next Section.

6 Experimental Results and Discussion

The simulations in the preceding section indicate that the proposed classifier works according to the expectations from its derivation. This section will now show how the classifier performs on real data where a bigger amount of uncertainty and challenges are to be expected.

6.1 Experiment Setup

In order to verify the proposed algorithm, real world experiments have been conducted on a two-way country road with moderate traffic density and a speed limit of 90 km/h. Two commercially available 2-axis magnetometers (Honeywell HMC6042, Honeywell (2007)) sampling at 100 Hz were deployed on both sides of the road as illustrated in Figure 8. The traffic was measured during three separate periods for a total of 158 minutes. In total 362 vehicles traveling south-north (close to sensor 1) and 305 vehicles traveling north-south (close to sensor 2) were measured.

In addition, a video recording was conducted along with the magnetometer measurements. From the video recording the time when the vehicles were in front of the sensor (passing time) as well as their driving directions were manually determined in order to establish a ground truth. From this ground truth, a 1.5 s sequence of the magnetometer signal centered around the true passing

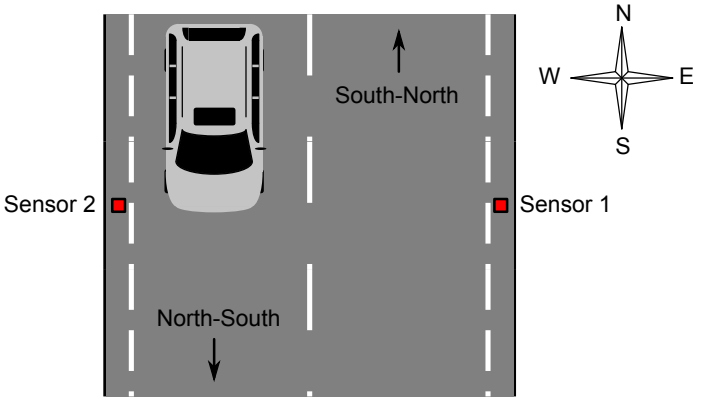


Figure 8: *Illustration of the experiment setup showing the two sensors on each side of the road as well as the driving directions.*

time for each vehicle was extracted from the raw measurement data. These data sequences were then used in order to evaluate the direction classification algorithms as presented in the previous sections. In this way, the detection problem does not affect the comparison of the two classification algorithms.

Note that the two sensor setup is only used for increasing the amount of data and for evaluating the presented fusion framework. The presented classifier in its simplest form still only needs one sensor for classifying the driving direction.

6.2 Results

As indicated above, the ground truth data was used in order to measure the performance of the two classifiers. For each passage, a 1.5 s long time window from the magnetometer signal was extracted and the driving direction was determined. For the likelihood classifier introduced in Section 4, the chosen parametrization was as follows:

$$\begin{aligned} \mathcal{H}_L: &\quad \theta_1^\star = \begin{bmatrix} 25\,\mathrm{m/s} & 3.5\,\mathrm{m} \end{bmatrix} \\ \mathcal{H}_R: &\quad \theta_2^\star = \begin{bmatrix} -25\,\mathrm{m/s} & 6.5\,\mathrm{m} \end{bmatrix} \end{aligned}$$

which corresponds to the actual road geometry and speed limit at the place where the measurements were performed. The correlation classifier was tuned using the tuning algorithm described in Section 3.3 using the first measurement set (71 vehicles close to sensor 1 and 85 vehicles close to sensor 2) resulting in $p = 11$.

Finally, the two algorithms were run on the remaining two datasets and the results are shown in Table 1. As it can be seen in the table, the correlation classifier performs very well. For the vehicles passing close to the sensor, only one out of the 511 vehicles is misclassified. As expected due to the lower SNR compared to vehicles passing close to the sensor, the performance is worse for the vehicles passing on the lane farther from the sensor. Here, in total 57 vehicles are wrongly classified.

Table 1: *Results of applying the driving direction classification to the measurement data. For example, 290 out of the 291 vehicles traveling south-north were classified correctly by the correlation classifier using the measurements of sensor 1.*

Direction	**# Vehicles**	**Sensor 1**		**Sensor 2**		**Fusion**	
		Correlation	**Likelihood**	**Correlation**	**Likelihood**	**Correlation**	**Likelihood**
South-North [a]	291	290	278	265	215	283	273
North-South [b]	220	189	164	220	210	211	207
Total	511	479	442	485	425	494	480

[a] Close to sensor 1
[b] Close to sensor 2

Table 2: *Results of applying the driving direction classification to the measurement data. For example, 710 out of the 722 vehicles which are more then 2 seconds separated from the closest vehicle, were classified correctly by the cross-correlation based classifier.*

Distance to closest vehicle	# Vehicles	Correlation	Likelihood
More than 2 s	722	710	642
Less than 2 s	300	254	225

Also the likelihood classifier performs well for vehicles close to the sensor but not quite as well as the cross-correlation classifier. 23 vehicles are wrongly classified for the case where the vehicles pass close to the sensor and 132 – around twice as many as for the correlation classifier – are wrongly classified for vehicles passing far from the sensor.

The incorrect classifications can be divided into the following three main sources of error which affect the two classifiers differently:

1. If multiple vehicles are present at the same time, the single target assumption made in the dipole model (1a) is violated. This situation occurs if two vehicles heading in different directions are passing each other in front of the sensors or if a train of vehicles is passing the sensors with short distance between the vehicles.

2. For large vehicles, the dipole model (1a) is also violated. This is because the dipole model does not assume any geometrical extent of the vehicle. Furthermore, for very big vehicles this will give raise to a saturated signal as displayed in Figure 2, which clearly violates the assumption in (1a).

3. Finally, if the SNR is poor the classification result will become worse as also concluded in the simulations in Section 5.

In order to quantify the impact of the first source of error on the two classifiers, the available data was first split into two groups – one group where the vehicles are more than 2 seconds separated from each other and one group where the vehicles are less than 2 seconds separated from each other. Each vehicle gives rise to two data sets since we have two sensors. Therefore, in total $511 \times 2 = 1,022$ data sets are considered here. The classification results for these two groups are presented in Table 2. According to these results both classifiers are degraded when the single target assumption is violated, however the likelihood test suffers more from this violation than the correlation classifier does.

Furthermore, when the distance between the vehicles is larger than 2 seconds and the single target assumption applies, the correlation classifier also performs better than the likelihood test. In order to further investigate these differences, this group of data has been grouped into 8 classes of different SNR levels, each class having an interval of 5 dB. In Figure 9 the classification result for these

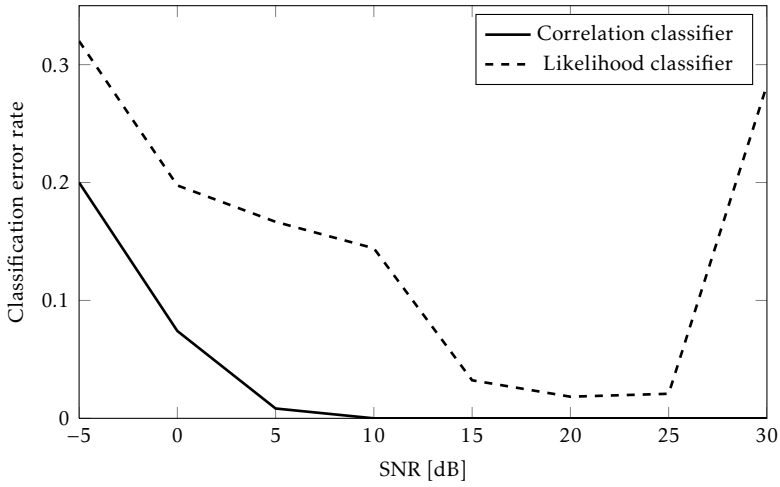


Figure 9: *Classification performance compared with the likelihood test as a function of SNR.*

groups is presented as a function of the SNR, which was computed as

$$\text{SNR} = 10\log_{10}\left(\frac{\frac{1}{N}\sum_{k=1}^{N}\|\bar{\mathbf{h}}_k\|^2}{\sigma^2}\right)\text{dB} \approx 10\log_{10}\left(\frac{\frac{1}{N}\sum_{k=1}^{N}\|\mathbf{y}_k\|^2 - \sigma^2}{\sigma^2}\right)\text{dB},$$

where σ^2 is the variance of the measurement noise.

According to Figure 9 the correlation classifier performs better than the likelihood test for all present SNR levels. Also notice that the correlation classifier has a zero error rate where $\text{SNR} \gtrapprox 10\,\text{dB}$, which the likelihood does not. As a matter of fact, the likelihood test performs even worse for $\text{SNR} \gtrapprox 25\,\text{dB}$ corresponding to large vehicles close to the sensor. As explained above, this is because large vehicles violate the point target assumption, which the dipole model (1a) is based upon. However, it is important to note that the performance of the correlation classifier is not affected by these model violations since it does not use that model explicitly, but only one property of it and thus, the second source of error only affects the likelihood classifier. According to the experimental results, this property is still valid even in a near field scenario since we still have excellent classification results, even for high SNRs .

It is instructive to compare the classification performance evaluated on real data in Figure 9 with the performance on simulated data in Figure 7. When evaluating the classification on simulated data, the likelihood classifier performs better than the correlation classifier does. However, for real data the likelihood classifier is heavily disadvantaged and it performs worse than the correlation classifier due to the reasons discussed above.

Finally, the last two columns in Table 1 show the results for sensor fusion where the classification of both sensors were taken into account. Clearly, the overall performance is improved; the total number of correct classifications including

vehicles on both, the closer and farther lane, is larger than for the case when only one sensor was used. Note that it appears as if the fusion results were worse when comparing it to, for example, the results of sensor 1 and vehicles travelling in the south-north direction. However, this is a biased comparison since one preselects the cases favorable for sensor 1 by looking at the isolated results for the south-north direction. The important result is the overall performance where the fused result is better than that of the individual sensors.

6.3 Discussion

As the results in the preceding sections indicate, the proposed driving direction classifier performs very well and generally outperforms the likelihood classifier. The challenges encountered in the previous section are discussed in more detail in this section.

The first problem was related to scenarios with multiple vehicles in the scene. This violates the single target assumption, which both classifiers are based upon. The likelihood test could be extended to handle this case by modifying the model (27) to include multiple dipoles, which requires even more parameters to estimate. For the proposed correlation classifier such an extension is not straightforward since the feature has been extracted under the single target assumption and is otherwise not valid. This might be considered as a limitation of this classification method.

However, in case of multiple targets both classifiers most likely will classify according to the vehicle with the highest SNR. Since the SNR decays cubically with the distance to the magnetic source, see (1), the vehicle with the highest SNR is most likely the one closest to the sensor. However, this assumption might not always be valid. In Figure 10 a scenario with two vehicles of different sizes is presented where the classification result for the sensor closest to the smallest vehicles can not be resolved with the present algorithm, since both vehicles affect the sensor signal in the same order of magnitude. This also reveals the robustness of the proposed algorithm in cluttered environments. If the magnitude of the clutter is lower than the magnitude of signal induced by the vehicle, the driving direction will most likely be classified correctly. Due to the cubically decay in SNR as a function of distance, only clutter in the immediate vicinity of the sensor will be a potential problem. The same reasoning would be valid for the detection performance as well. However, as explained previously, the detection problem is not analyzed further in this work.

A possibility of handling multiple vehicles in the scene is to design a more elaborate post-processing strategy in a multisensor scenario. For example, instead of just fusing the individual classifications from two sensors on opposite sides of the road as proposed in Section 3.4, one could inspect the uncertainty of the two classifications more thoroughly. If both sensors classify an event in the opposite driving directions with a high certainty, this could be an indication of two vehicles passing each other in front of the sensors. Further, in a more extensive multi-sensor scenario, the detection and classification from each sensor could be associated and processed to perform tracking in a multitarget-multisensor sce-

nario (Bar-Shalom and Li, 1995). In such scenarios the possibility to fuse information from other sensor types can also be analyzed.

The second problem for the correlation classifier was related to poor signal-to-noise ratios. Obviously, this is a problem that affects any algorithm to some extent; the important aspect is at which level the degradation becomes critical. Taking the experimental results in Figure 9 into account, one sees that even at an SNR as low as 0 dB, the classification error rate is below 10 %. This indicates that the correlation classifier performs relatively well even in these conditions. Nevertheless, it is apparent that the SNR can not be arbitrarily low.

Furthermore, a disadvantage of the proposed method is the fact that large averaging windows (large p) will cause problems at high speeds (compared to the sampling rate). If a vehicle passes the sensor very fast, only few points of the loop trajectory will be measured. Averaging over these few points will have unwanted effects and it might happen that the estimation becomes wrong. However, note that the correlation classifier behaves better in general since no assumption on the vehicle trajectory and/or speed was made, compared to the likelihood test in Section 4. Clearly, the likelihood test could also be extended to take variations in these parameters into account, however, at a cost of higher complexity.

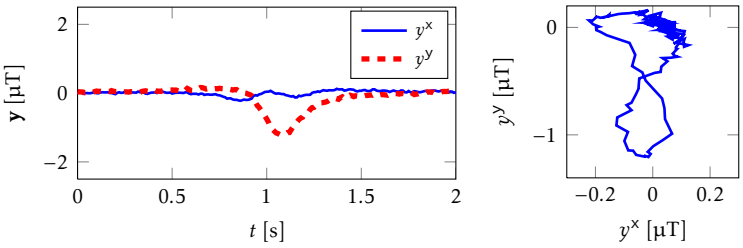
Finally, it is important to obtain an indication about whether the classification was successful or not for the two problematic cases for the correlation classifier. For the case of signals with poor SNR, this is again reflected in the classification's uncertainty (that is, in the variance of the feature $\hat{f}_p$). For multiple vehicles, the only remedy is to rely on the data of multiple sensors as indicated above.

7 Conclusions

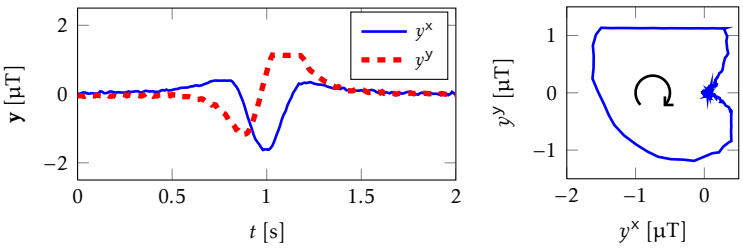
By using measurements from a 2-axis magnetometer, a fast and efficient method for classifying the driving direction of a vehicle has been proposed. Its properties were first analyzed theoretically and then verified by using Monte Carlo simulations before it was applied to real measurement data from commercially available sensor. The method was also compared to a generalized likelihood ratio test and it was shown how the method can be extended to incorporate data from multiple independent sensor nodes in a sensor network.

The results show that the performance of the proposed method is very good for vehicles passing on the lane close to the sensor where more than 99 % of all vehicles were classified correctly (sample size: 511 vehicles). As it can be expected, performance degrades as the signal-to-noise ratio decreases which reflects in the fact that the classification rate for vehicles on the lane farther from the sensor dropped to approximately 89 %. Comparing this method to the generalized likelihood ratio test, it is seen that the latter is outperformed in every aspect. It is very likely that one particular reason for this behavior is the fact that one or more of the assumptions (for example, dipole model, speed, or trajectory) made to derive the GLRT are violated. The biggest difficulties arise in cases where two vehicles meet right in front of the sensor. Apparently, the vehicle generating the stronger field distortion will dominate the signal and the second vehicle is shadowed.

(a) Two meeting vehicles



(b) Signal from sensor 1 (closer to the camera)



(c) Signal from sensor 2 (farther from the camera)

Figure 10: *A scenario is presented with vehicles of different sizes meeting in front of the sensors. Sensor 2 will classify according to the driving direction of the larger close-by white vehicle (c), whereas the classification result for sensor 1 is ambiguous since the the smaller close-by white vehicle and the farther larger vehicle affect the signal with the same magnitude.*

The proposed algorithm as presented in this work is restricted to work on single target scenarios, where possible extensions to handle multiple targets have been discussed. Future work should focus on developing these ideas further and evaluate them on real data. Furthermore, the proposed method will be implemented in a real sensor platform and the performance in a real time system will be analyzed.

Appendix

A Distributions

Proposition 1. *Consider the estimator $\hat{f}_p$ given by* (11) *and the measurement signal $\mathbf{y}_k$ as in* (1b) *and* (1c)*. Then, the estimator $\hat{f}_p$ is distributed according to*

$$\hat{f}_p \sim \mathcal{N}(f_p, \sigma^2_{\hat{f}_p}) \quad \textit{when} \quad N \to \infty \tag{41}$$

where f_p and $\sigma^2_{\hat{f}_p}$ are given by (12) *and* (14)*, respectively.*

Proof: As stated above, the mean and variance of $\hat{f}_p$ have been derived in (12) and (14). What remains to show is that $\hat{f}_p$ is normal distributed. Starting by expanding the original expression for $\hat{f}_p$ yields

$$\begin{aligned}
\hat{f}_p &= \frac{1}{p}\sum_{k=1}^{N}(y^{\mathrm{x}}_k y^{\mathrm{y}}_{k+p} - y^{\mathrm{y}}_k y^{\mathrm{x}}_{k+p}) \\
&= \frac{1}{p}\sum_{k=1}^{N}(h^{\mathrm{x}}_k + e^{\mathrm{x}}_k)(h^{\mathrm{y}}_{k+p} + e^{\mathrm{y}}_{k+p}) - (h^{\mathrm{y}}_k + e^{\mathrm{y}}_k)(h^{\mathrm{x}}_{k+p} + e^{\mathrm{x}}_{k+p}) \\
&= \frac{1}{p}\sum_{k=1}^{N} h^{\mathrm{x}}_k h^{\mathrm{y}}_{k+p} + h^{\mathrm{x}}_k e^{\mathrm{y}}_{k+p} + e^{\mathrm{x}}_k h^{\mathrm{y}}_{k+p} + e^{\mathrm{x}}_k e^{\mathrm{y}}_{k+p} - (h^{\mathrm{y}}_k h^{\mathrm{x}}_{k+p} + h^{\mathrm{y}}_k e^{\mathrm{x}}_{k+p} + e^{\mathrm{y}}_k h^{\mathrm{x}}_{k+p} + e^{\mathrm{y}}_k e^{\mathrm{x}}_{k+p}).
\end{aligned} \tag{42}$$

Similar to (13)–(14), the sum in (42) can now be rearranged so that all the deterministic coefficients of e^{α}_k are gathered together:

$$\hat{f}_p = \frac{1}{p}\sum_{k=1}^{N} h^{\mathrm{x}}_k h^{\mathrm{y}}_{k+p} - h^{\mathrm{y}}_k h^{\mathrm{x}}_{k+p} + (h^{\mathrm{y}}_{k+p} - h^{\mathrm{y}}_{k-p})e^{\mathrm{x}}_k - (h^{\mathrm{x}}_{k+p} - h^{\mathrm{x}}_{k-p})e^{\mathrm{y}}_k + e^{\mathrm{y}}_{k+p}e^{\mathrm{x}}_k - e^{\mathrm{x}}_{k+p}e^{\mathrm{y}}_k. \tag{43}$$

Distributing the sum to the individual terms finally gives

$$\begin{aligned}
\hat{f}_p =& \frac{1}{p}\sum_{k=1}^{N}(h^{\mathrm{x}}_k h^{\mathrm{y}}_{k+p} - h^{\mathrm{y}}_k h^{\mathrm{x}}_{k+p}) + \frac{1}{p}\sum_{k=1}^{N}(h^{\mathrm{y}}_{k+p} - h^{\mathrm{y}}_{k-p})e^{\mathrm{x}}_k \\
&- \frac{1}{p}\sum_{k=1}^{N}(h^{\mathrm{x}}_{k+p} - h^{\mathrm{x}}_{k-p})e^{\mathrm{y}}_k + \frac{1}{p}\sum_{k=1}^{N} e^{\mathrm{y}}_{k+p}e^{\mathrm{x}}_k - \frac{1}{p}\sum_{k=1}^{N} e^{\mathrm{x}}_{k+p}e^{\mathrm{y}}_k.
\end{aligned} \tag{44}$$

Since $\mathbf{e}_k \sim \mathcal{N}\left(0, \sigma^2\right)$, the second and third term in (44) will also be normal distributed. Furthermore, since all $e^{\mathrm{x}}_1, \ldots, e^{\mathrm{x}}_N, e^{\mathrm{y}}_1, \ldots, e^{\mathrm{y}}_N$ are independent, the overall variance is simply the sum of the individual variances for each variable and the distributions for these two terms are

$$\mathcal{N}\left(0, \frac{\sigma^2}{p^2}\sum_{k=1}^{N}(h^{\mathrm{y}}_{k+p} - h^{\mathrm{y}}_{k-p})^2\right) \quad \text{and} \quad \mathcal{N}\left(0, \frac{\sigma^2}{p^2}\sum_{k=1}^{N}(h^{\mathrm{x}}_{k+p} - h^{\mathrm{x}}_{k-p})^2\right). \tag{45a}$$

The last two terms in (44) are sums of normal product distributed variables. The variance for each such variable is

$$\mathrm{Var}(e^{\mathsf{y}}_{k+p}e^{\mathsf{x}}_k) = \mathrm{Var}(e^{\mathsf{y}}_{k+p})\mathrm{Var}(e^{\mathsf{x}}_k) = \sigma^2\sigma^2 = \sigma^4 \tag{45b}$$

and since $e^{\mathsf{y}}_{k+p}e^{\mathsf{x}}_k$ are all independent and $\sigma^4 < \infty$, the Central Limit Theorem (see, for example, Billingsley (1995)) yields

$$\frac{1}{p}\sum_{k=1}^{N} e^{\mathsf{y}}_{k+p}e^{\mathsf{x}}_k \xrightarrow{d} \mathcal{N}\left(0, \frac{N\sigma^4}{p^2}\right) \quad \text{and} \quad \frac{1}{p}\sum_{k=1}^{N} e^{\mathsf{y}}_k e^{\mathsf{x}}_{k+p} \xrightarrow{d} \mathcal{N}\left(0, \frac{N\sigma^4}{p^2}\right) \tag{45c}$$

as $N \to \infty$.

Consequently, $\hat{f}_p$ is a sum of one deterministic constant and four zero mean normal distributed variables and since a sum of (possibly dependent) normal distributed variables is also normal distributed we have that

$$\hat{f}_p \sim \mathcal{N}(f_p, \sigma^2_{\hat{f}_p}) \quad \text{when} \quad N \to \infty. \tag{46}$$

□

B Fusion of Conditional Bernoulli Random Variables

Given multiple conditional PDFs $p(x|y_1), \ldots, p(x|y_J)$ it is of interest to find the joint-conditional PDF $p(x|y_1, \ldots, y_J)$. Using Bayes rule and assuming that all $y_1, \ldots, y_J$ are statistically independent given the true x (conditional independence) yields

$$p(x|y_1, \ldots y_J) = \frac{p(y_1, \ldots, y_J|x)p(x)}{p(y_1, \ldots, y_J)} = p(x)\frac{\prod_{j=1}^{J} p(y_j|x)}{p(y_1, \ldots, y_J)}$$

and using Bayes' rule on each $p(y_j|x)$ gives

$$p(x|y_1, \ldots, y_J) = \frac{p(x)}{p(y_1, \ldots, y_J)}\prod_{j=1}^{J}\frac{p(x|y_j)p(y_j)}{p(x)} = \frac{\prod_{j=1}^{J} p(y_j)}{p(x)^{J-1}p(y_1, \ldots, y_J)}\prod_{j=1}^{J} p(x|y_j).$$

Assuming the uninformative prior $p(x) \propto 1$ yields that $p(x|y_1, \ldots, y_J) \propto \prod_{j=1}^{J} p(x|y_j)$ and thus

$$p(x|\mathbf{y}) = \frac{1}{\eta}\prod_{j=1}^{J} p(x|y_j) \quad \text{where} \quad \eta = \int_{\mathcal{D}}\prod_{j=1}^{J} p(x|y_j)dx, \tag{47}$$

with $\mathbf{y} = \begin{bmatrix} y_1 & y_2 & \ldots & y_J \end{bmatrix}^{\mathsf{T}}$ and η being a normalization constant.

Using (47) for a set of Bernoulli random variables described by the PDF

$$p(k|p_j) = p_j^k(1-p_j)^{1-k} \quad \text{for} \quad k \in 0,1, \tag{48}$$

where p_j is the probability of success, the fused PDF becomes

$$p(k|\mathbf{p}) = \frac{\prod_{j=1}^{J} p_j^k(1-p_j)^{1-k}}{\prod_{j=1}^{J}(1-p_j) + \prod_{j=1}^{J} p_j}. \tag{49}$$

Bibliography

Y. Bar-Shalom and X.-R. Li. *Multitarget-Multisensor Tracking: Principles and Techniques*. Storrs, CT: YBS Publishing, 1995.

P. Billingsley. *Probability and measure*. Jown Wiley & Sons, 1995.

W. Birk, E. Osipov, and J. Eliasson. iRoad – cooperative road infrastructure systems for driver support. In *Proceedings of the 16th ITS World Congress and Exhibition on Intelligent Transport Systems and Services*, Stockholm, Sweden, September 2009.

M. Birsan. Non-linear Kalman filters for tracking a magnetic dipole. In *Proceedings of the International Conference on Marine Electromagnetics*, London, UK, March 2004.

S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

S.-Y. Cheung and P. Varaiya. Traffic surveillance by wireless sensor networks: Final report. Technical report, Traffic surveillance, University of Califonia, Berkeley, 2007.

J. Chinrungrueng, S. Kaewkamnerd, R. Pongthornseri, S. Dumnin, U. Sunantachaikul, S. Kittipiyakul, S. Samphanyuth, A. Intarapanich, S. Charoenkul, and P. Boonyanant. Wireless sensor network: Application to vehicular traffic. *Advances in Wireless Sensors and Sensor Networks*, pages 199–220, 2010.

Federal Highway Administration. Traffic Monitoring Guide. Technical Report FHWA-PL-13-015, U.S. Department of Transportation, September 2013.

S. Giannecchini, M. Caccamo, and C.-S. Shih. Collaborative resource allocation in wireless sensor networks. In *Proceedings of the 16th Euromicro Conference on Real-Time Systems, ECRTS*, pages 35–44, Catania, Sicily, Italy, June 2004.

G. Girban and M. Popa. A glance on WSN lifetime and relevant factors for energy consumption. In *Proceedings of the International Joint Conference on Computational Cybernetics and Technical Informatics (ICCC-CONTI)*, pages 523–528, Timisoara, Romania, May 2010.

Y. Goyat, T. Chateau, L. Malaterre, and L. Trassoudaine. Vehicle trajectories evaluation by static video sensors. In *Proceedings of the 9th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2006*, pages 864–869, Toronto, Canada, September 2006.

G. Hegeman, A. Tapani, and S. Hoogendoorn. Overtaking assistant assessment using traffic simulation. *Transportation Research Part C: Emerging Technologies*, 17(6):617–630, 2009.

Honeywell. *2-Axis Magnetic Sensor Circuit HMC6042*, 2007. Datasheet.

J. D. Jackson. *Classical Electrodynamics*. John Wiley & Sons, Inc, 3rd edition, 1998.

S. M. Kay. *Fundamentals of Statistical Signal Processing: Detection Theory*. Prentice Hall, 1998.

L. A. Klein, M. K. Mills, and D. R. P. Gibson. Traffic detector handbook. Technical Report Volume I, FHWA-HRT-06-108, Federal Highway Administration, October 2006.

K. Kwong, R. Kavaler, R. Rajagopal, and P. Varaiya. Arterial travel time estimation based on vehicle re-identification using wireless magnetic sensors. *Transportation Research Part C: Emerging Technologies*, 17(6):586–606, 2009.

L. E. Y. Mimbela and L. A. Klein. Summary of vehicle detection and surveillance technologies used in intelligent transportation systems. Technical report, Federal Highway Administration, 2000.

S. Moler. Stop. You're going the wrong way! *Public Roads*, 66(2):24–29, September/October 2002.

National Highway Traffic Safety Administration. FARS encyclopedia: Manner of collision vs. total lanes in roadway, 2010.

N. A. Pantazis and D. D. Vergados. A survey on power control issues in wireless sensor networks. *Communications Surveys & Tutorials, IEEE*, 9(4):86–107, 2007.

M. Pucher, D. Schabus, P. Schallauer, Y. Lypetskyy, F. Graf, H. Rainer, M. Stadtschnitzer, S. Sternig, J. Birchbauer, W. Schneider, and B. Schalko. Multimodal highway monitoring for robust incident detection. In *Proceedings of the 13th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 837–842, Madeira Island, Portugal, September 2010.

M. Rakijas, A. Saglembeni, K. K. Kohnen, and H. C. Gilbert. Magnetic object tracking based on direct observation of magnetic sensor measurements, July 31 2001. US Patent 6,269,324.

W. L. Root. An introduction to the theory of the detection of signals in noise. *Proceedings of the IEEE*, 58(5):610–623, May 1970.

C. C. Sun, G. S. Arr, R. P. Ramachandran, and S. G. Ritchie. Vehicle reidentification using multidetector fusion. *IEEE Transaction on Intelligent Transportation Systems*, 5(3):155–164, 2004.

N. Wahlström and F. Gustafsson. Magnetometer modeling and validation for tracking metallic targets. *IEEE Transactions on Signal Processing*, 62(3):545–556, 2014.

N. Wahlström, J. Callmer, and F. Gustafsson. Magnetometers for tracking metallic targets. In *Proceedings of 13th International Conference on Information Fusion (FUSION)*, Edinburgh, Scotland, July 2010.

N. Wahlström, R. Hostettler, F. Gustafsson, and W. Birk. Rapid classification of vehicle heading direction with two-axis magnetometer. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3385–3388, Kyoto, Japan, March 2012.

N. Wahlström, R. Hostettler, F. Gustafsson, and W. Birk. Classification of driving direction in traffic surveillance using magnetometers. *IEEE Transactions on Intelligent Transportation Systems*, 15(4):1405–1418, 2014.

R. Wood, M. Palmer, L. Walter, and I. Rillie. Driver Interaction with Temporary Traffic Management. Technical Report RPN 1813, Transport Research Laboratory, UK, 2011.

W. M. Wynn. Detection, localization, and characterization of static magnetic dipole sources. In *Detection and Identification of Visually Obscured Targets*, pages 337–374. Taylor and Francis, 1999.

Y. Yu and V. K. Prasanna. Energy-balanced task allocation for collaborative processing in wireless sensor networks. *Mobile Networks and Applications*, 10: 115–131, 2005. ISSN 1383-469X.

X. Zhang and M. R. B. Forshaw. A parallel algorithm to extract information about the motion of road traffic using image analysis. *Transportation Research Part C: Emerging Technologies*, 5(2):141–152, 1997.

Paper C

Modeling Magnetic Fields Using Gaussian Processes

Authors: Niklas Wahlström, Manon Kok, Thomas B. Schön, and Fredrik Gustafsson

Edited version of the paper:

N. Wahlström, M. Kok, T. B. Schön, and F. Gustafsson. Modeling magnetic fields using Gaussian processes. In *Proceedings of the the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 3522–3526, Vancouver, Canada, May 2013.

Modeling Magnetic Fields Using Gaussian Processes

Niklas Wahlström, Manon Kok, Thomas B. Schön, and Fredrik Gustafsson

Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
{nikwa,manko,schon,fredrik}@isy.liu.se

Abstract

Starting from the electromagnetic theory, we derive a Bayesian nonparametric model allowing for joint estimation of the magnetic field and the magnetic sources in complex environments. The model is a Gaussian process which exploits the divergence- and curl-free properties of the magnetic field by combining well-known model components in a novel manner. The model is estimated using magnetometer measurements and spatial information implicitly provided by the sensor. The model and the associated estimator are validated on both simulated and real world experimental data producing Bayesian nonparametric maps of magnetized objects.

1 Introduction

The magnetic field has for a long time been used in navigation for providing seafarers and merchants as well as orienteers and migrating birds with heading information. In indoor environments this navigation task is challenged by the magnetic distortions caused by the ferromagnetic structure in buildings. However, these distortions can also provide position information using a magnetic map of the environment, either by navigating within a precomputed map or by performing simultaneous localization and mapping (SLAM). This requires good models of the magnetic field which will be investigated more deeply in this work by addressing the electromagnetic theory. The relation between the magnetic sources and their induced magnetic field is well understood and was already formulated by Maxwell (1865), see also Jackson (1998). However, little work has been done incorporating this knowledge into a statistical framework suitable for estimating magnetic fields in complex magnetic environments based on noisy data. We present a Bayesian nonparametric model (a particular Gaussian process) capable of modeling the magnetic field as well as the magnetic sources, see Figure 3. Our model exploits the divergence- and curl-free properties of the magnetic field inherited by the electromagnetic theory.

Gaussian processes (Rasmussen and Williams, 2006) have previously been

used for modeling magnetic fields in an indoor environment to enable SLAM (Vallivaara et al., 2010, 2011). Their navigation platform is equipped with a three axis magnetometer and the positioning is aided with odometry. However, in contrast to our work, the model of the magnetic map does not incorporate knowledge from Maxwell's equations and is not able to estimate the location of the magnetic sources. The same model has been investigated further by Kemppainen et al. (2011). Gaussian processes have recently also been used for SLAM in a scalar potential field (Murphy and Godsill, 2012). However, we consider multiple vector fields rather than one scalar potential field in this work. Vissière et al. (2007) also use the magnetic disturbances to improve IMU-based position estimation. That work uses the restrictions induced by the electromagnetic theory, but does not construct any magnetic map and does not localize the magnetic sources. (Chung et al., 2011) uses only four tilt-compensated magnetometers to accomplish indoor localization. The magnetic map is captured in advance consisting of a collection of magnetic signatures. The localization is performed using magnetic map fingerprints, where the performance is enhanced by the multiplicity of the magnetometers. Also fusion of magnetic field anomalies and laser has been investigated (Zhang and Martin, 2011).

This work only addresses the modeling aspects of the magnetic field and the magnetic sources. The localization problem we consider separately (Kok et al., 2013), but the model is also suitable for being used in the applications presented above. The contributions of this work are:

- A Gaussian process model which in a novel manner exploits the divergence- and curl-free properties of the magnetic field.
- The model enables the magnetic field and the magnetic sources to be estimated jointly.
- Interference with both magnetic field measurements and spatial information is possible.
- We validate the model using both simulations and real world experimental data.

The divergence- and curl-free properties of a vector field have previously been used for estimating fluid flows using Gaussian processes (Macêdo and Castro, 2008). However, to the best authors' knowledge this has previously not been used in modeling magnetic fields.

2 Magnetic Fields

A magnetic field is a mathematical construction used for describing forces induced by magnetic materials and electric currents. For each point in space the magnetic field can be described using a vector and as such it is a vector field. There are two different, but closely related ways to describe the magnetic field, denoted with the symbols $\mathbf{B}$ and $\mathbf{H}$, where boldface denotes vector-valued quantities.

These fields can not be any two arbitrary vector fields, but need to obey physical laws, which in their most general form are known as Maxwell's equations (Jackson, 1998). By assuming absence of free currents and time-dependent effects, these equations will reduce to

$$\nabla \cdot \mathbf{B} = 0, \tag{1a}$$

$$\nabla \times \mathbf{H} = \mathbf{0}, \tag{1b}$$

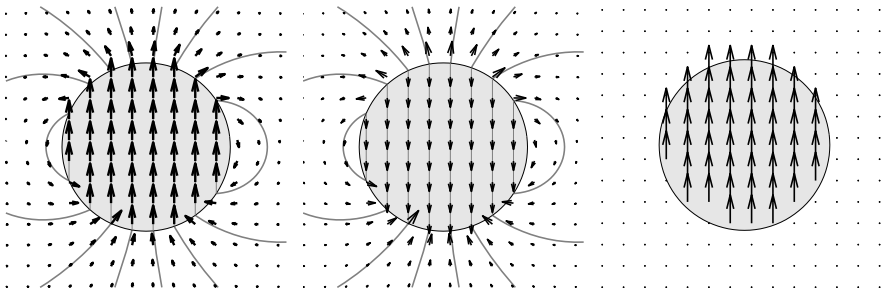
which means that the **B**-field is divergence-free and the **H**-field is curl-free. Further, these two fields are coupled as

$$\mathbf{M} = \frac{1}{\mu_0}\mathbf{B} - \mathbf{H}, \tag{2}$$

where **M** is the magnetization describing our magnetic environment and μ_0 is the vacuum permeability, which is a physical constant having the value $\mu_0 = 4\pi \times 10^{-7}\,\mathrm{V\,s\,A^{-1}\,m^{-1}}$. These fields will be illustrated with the following example. More details on the derivation can be found in Jackson (1998).

Example 1: Uniformly magnetized sphere

Consider a sphere with a uniform permanent magnetization as depicted in Figure 1c. By solving (1) and (2) for this special geometry we will end up in a dipole field outside the sphere as depicted in Figure 1a and 1b. Note that the **B**- and the **H**-field will be identical (up to the proportional constant μ_0) outside the sphere, which follows directly from (2) using $\mathbf{M} = \mathbf{0}$. However, inside the sphere the **B**- and the **H**-field will be aligned in opposite directions in order to ensure that the **B**-field is divergence-free (no sources or sinks) and that the **H**-field is curl-free (no swirls).



(a) **B**-*field* *(b)* **H**-*field* *(c)* **M**-*field*

Figure 1: *The* **B**-*,* **H**- *and* **M**-*field of a uniformly magnetized sphere. The* **B**-*field is here normalized with* μ_0.

By using (1)-(2) and prior knowledge of the magnetic environment, a number of things can be concluded, which will be used in Section 4 when modeling the magnetic fields:

1. **Additional information** In all non-magnetic materials the magnetization is equal to zero, $\mathbf{M} = \mathbf{0}$. This is especially true in locations where we measure the magnetic field, since air is non-magnetic. Due to physical constraints the sensor cannot be inside a magnetic material and we know that $\mathbf{M} = \mathbf{0}$ at these positions. This additional information will be used in our framework as an extra measurement. Sensor fusion with other sensors such as camera and laser range sensor, providing even richer information of where $\mathbf{M} = \mathbf{0}$, is possible. However, this is not considered in this work.

2. **External field** Most environments of interest consist of an external homogeneous field $\mathbf{B}_0$ or $\mathbf{H}_0$, usually the earth magnetic field or a slight deformation of it. Due to the linearity of the field equations (1), this external field can be superimposed throughout all space, where (2) gives the relation $\mathbf{B}_0 = \mu_0 \mathbf{H}_0$. We will therefore later model the $\mathbf{B}$- and the $\mathbf{H}$-field to have a common, but unknown mean.

3. **Smoothness** If $\mathbf{M}(\mathbf{u}) = \mathbf{0}$ in a neighborhood of $\mathbf{u}$, the field equations (1) will ensure that $\mathbf{B}$ and $\mathbf{H}$ are infinitely continuously differentiable at $\mathbf{u}$. This gives the magnetic field a "smooth" character and the magnetic field at $\mathbf{u}_1$ will be very similar to the magnetic field at $\mathbf{u}_2$ if $\mathbf{u}_1$ and $\mathbf{u}_2$ are close. This property motivates us to employ Gaussian processes in modeling these fields, as will be explained in the next section.

3 Gaussian Processes

A Gaussian process (GP) (Rasmussen and Williams, 2006) is a stochastic process suitable for modeling spatially correlated measurements. GPs can be seen as a distribution over functions

$$\mathbf{f}(\mathbf{u}) \sim \mathcal{GP}\Big(\boldsymbol{\mu}(\mathbf{u}), K(\mathbf{u}, \mathbf{u}')\Big), \tag{3}$$

where the process is uniquely defined with its mean function $\boldsymbol{\mu}(\mathbf{u})$ and covariance function $K(\mathbf{u}, \mathbf{u}')$.

The GP is a generalization of the multivariate Gaussian probability distribution in the sense that the function values evaluated for a finite number of inputs $\mathbf{u}_1, \ldots, \mathbf{u}_N$ are normally distributed

$$\begin{bmatrix} \mathbf{f}(\mathbf{u}_1) \\ \vdots \\ \mathbf{f}(\mathbf{u}_N) \end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}, K), \quad \text{where} \quad \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}(\mathbf{u}_1) \\ \vdots \\ \boldsymbol{\mu}(\mathbf{u}_N) \end{bmatrix}, \quad K = \begin{bmatrix} K(\mathbf{u}_1, \mathbf{u}_1) & \cdots & K(\mathbf{u}_1, \mathbf{u}_N) \\ \vdots & & \vdots \\ K(\mathbf{u}_N, \mathbf{u}_1) & \cdots & K(\mathbf{u}_N, \mathbf{u}_N) \end{bmatrix}. \tag{4}$$

3.1 Mean Function

In this work we will consider a constant, but unknown mean function $\boldsymbol{\mu}(\mathbf{u}) = \boldsymbol{\beta}$, where we put a Gaussian prior on the mean

$$\mathbf{f}(\mathbf{u}) \sim \mathcal{GP}\Big(\boldsymbol{\beta}, K(\mathbf{u}, \mathbf{u}')\Big), \quad \text{where} \quad \boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \sigma_\beta^2 I). \tag{5a}$$

By integrating out the parameter β, this can be reformulated as a zero mean GP

$$\mathbf{f}(\mathbf{u}) \sim \mathcal{GP}\Big(\mathbf{0}, K(\mathbf{u}, \mathbf{u}') + \sigma_\beta^2 I\Big). \tag{5b}$$

3.2 Vector-Valued Covariance Functions

The covariance function (a.k.a. kernel) is the crucial component when modeling using a GP. This function encodes the assumptions we make on the functions to be learned. For modeling smooth functions (as desired in Item 3 in Section 2) with scalar output the most common choice is the squared exponential (SE) covariance function

$$K(\mathbf{u}, \mathbf{u}') = k(\mathbf{u}, \mathbf{u}') = \sigma_f^2 e^{-\frac{\|\mathbf{u}-\mathbf{u}'\|^2}{2l^2}}, \tag{6}$$

where σ_f is the expected amplitude and l the expected length-scale of the function we want to learn. This covariance function can be extended for learning functions with multiple outputs as presented below. Learning functions with multiple outputs has recently attracted more attention. A review can be found in Álvarez et al. (2012), which discusses different kernels for learning vector-valued functions.

Diagonal Squared Exponential Covariance Function

The most obvious extension of (6) to multiple outputs is to model each component $f_i(\mathbf{u})$ separately using a scalar SE covariance functions resulting in a diagonal SE kernel

$$K(\mathbf{u}, \mathbf{u}') = \sigma_f^2 e^{-\frac{\|\mathbf{u}-\mathbf{u}'\|^2}{2l^2}} \cdot I_{n_\mathbf{y}}, \tag{7}$$

where $n_\mathbf{y}$ is the dimension of the output. The kernel (7) can be extended to have different hyperparameters l and σ_f for each output dimension. This kernel was used by Vallivaara et al. (2010) and Vallivaara et al. (2011) in modeling the magnetic field of an indoor environment. However, this kernel does not allow for the possibility of modeling correlations between the different components $f_i(\mathbf{u})$. Specially, it does not produce functions which necessarily obey the field equations (1). This is made possible by the two covariance functions introduced below.

Divergence- and Curl-Free Covariance Functions

A kernel for learning divergence-free vector fields was first introduced by Narcowich and Ward (1994). Based on the scalar SE kernel (6), this kernel reads

$$K_\mathbf{B}(\mathbf{u}, \mathbf{u}') = \sigma_f^2 e^{-\frac{\|\mathbf{u}-\mathbf{u}'\|^2}{2l^2}} \cdot \left(\left(\frac{\mathbf{u}-\mathbf{u}'}{l}\right)\left(\frac{\mathbf{u}-\mathbf{u}'}{l}\right)^\mathsf{T} + \left(n_\mathbf{y} - 1 - \frac{\|\mathbf{u}-\mathbf{u}'\|^2}{l^2}\right) I_{n_\mathbf{y}}\right) \tag{8}$$

which ensures that all functions sampled from a GP with such a kernel will be divergence-free. Similarely, Fuselier Jr (2006) introduced a kernel for learning curl-free vector fields, where the extension of (6) reads

$$K_{\mathbf{H}}(\mathbf{u},\mathbf{u}') = \sigma_f^2 e^{-\frac{\|\mathbf{u}-\mathbf{u}'\|^2}{2l^2}} \left(I_{n_\mathbf{y}} - \left(\frac{\mathbf{u}-\mathbf{u}'}{l}\right)\left(\frac{\mathbf{u}-\mathbf{u}'}{l}\right)^\mathsf{T} \right). \tag{9}$$

The interested reader can refer to Fuselier Jr (2006); Macêdo and Castro (2008); Baldassarre et al. (2010) for more analysis and discussion on these two kernels.

3.3 Regression

GPs are also capable of handling noisy measurements $\mathbf{y}_k$ of the function $\mathbf{f}(\mathbf{u}_k)$. We consider the measurement model

$$\mathbf{y}_k = \mathbf{f}(\mathbf{u}_k) + \mathbf{e}_k, \qquad \mathbf{e}_k \sim \mathcal{N}(\mathbf{0}, \Sigma), \tag{10}$$

where $\mathbf{e}_k$ has the interpretation of being measurement noise. Our objective is to use a set of measurements together with their corresponding inputs $\{\mathbf{u}_k, \mathbf{y}_k | k = 1,\dots,N\}$ to learn the function values for other test inputs $\mathbf{f}_* = [\mathbf{f}(\mathbf{u}_1^*)^\mathsf{T} \ \dots \ \mathbf{f}(\mathbf{u}_{N_*}^*)^\mathsf{T}]^\mathsf{T}$. In the same manner as in (4) the joint distribution for the measurements $\mathbf{y} = \begin{bmatrix}\mathbf{y}_1^\mathsf{T} & \dots & \mathbf{y}_N^\mathsf{T}\end{bmatrix}^\mathsf{T}$ and the test output $\mathbf{f}_*$ is

$$\begin{bmatrix}\mathbf{y}\\ \mathbf{f}_*\end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(U,U) + I_N \otimes \Sigma & K(U,U_*) \\ K(U_*,U) & K(U_*,U_*)\end{bmatrix}\right), \tag{11}$$

where $\otimes$ denote the Kronecker product,

$$K(U,U_*) = \begin{bmatrix} K(\mathbf{u}_1,\mathbf{u}_1^*) & \dots & K(\mathbf{u}_1,\mathbf{u}_{N_*}^*) \\ \vdots & & \vdots \\ K(\mathbf{u}_N,\mathbf{u}_1^*) & \dots & K(\mathbf{u}_N,\mathbf{u}_{N_*}^*)\end{bmatrix} \tag{12}$$

and similarly for the other matrices $K(U,U)$, $K(U_*,U)$ and $K(U_*,U_*)$. From the joint Gaussian distribution $p(\mathbf{y},\mathbf{f}_*)$ in (11) the conditional distribution $p(\mathbf{f}_*|\mathbf{y})$ can easily be computed as

$$\mathbf{f}_*|\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{f}_*}, \Sigma_{\mathbf{f}_*}), \tag{13a}$$

$$\boldsymbol{\mu}_{\mathbf{f}_*} = K_*^\mathsf{T} K_\mathbf{y}^{-1}\mathbf{y}, \qquad \Sigma_{\mathbf{f}_*} = K_{**} - K_*^\mathsf{T} K_\mathbf{y}^{-1} K_*, \tag{13b}$$

where $K = K(U,U)$, $K_* = K(U,U_*)$, $K_{**} = K(U_*,U_*)$ and $K_\mathbf{y} = K(U,U) + I_N \otimes \Sigma$.

3.4 Estimating Hyperparameters

The hyperparameters of the covariance function $K(\mathbf{u},\mathbf{u}')$ and the measurement noise covariance matrix Σ can be estimated from the data $\{\mathbf{u}_k, \mathbf{y}_k | k = 1,\dots N,\}$, which makes the learning of the function values $\mathbf{f}_*$ completely data driven in

the sense that no tuning parameters are needed. This will be accomplished by maximizing the log marginal likelihood $\log p(\mathbf{y}, |U, \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ denote the hyperparameters of $K(\mathbf{u}, \mathbf{u}')$ and Σ. From (11) we have that $\mathbf{y}|U, \boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, K_{\mathbf{y}})$, which gives

$$\log p(\mathbf{y}|U, \boldsymbol{\theta}) = -\frac{1}{2}\mathbf{y}^\top K_{\mathbf{y}}^{-1}\mathbf{y} - \frac{1}{2}\log|K_{\mathbf{y}}| - \frac{n_{\mathbf{y}}N}{2}\log 2\pi. \tag{14}$$

Following Rasmussen and Williams (2006), the gradient of the log marginal likelihood w.r.t. the hyperparameters can be computed as

$$\frac{\partial}{\partial \boldsymbol{\theta}_j}\log p(\mathbf{y}|U, \boldsymbol{\theta}) = \frac{1}{2}tr\left((\boldsymbol{\alpha}\boldsymbol{\alpha}^\top - K_{\mathbf{y}}^{-1})\frac{\partial K_{\mathbf{y}}}{\partial \boldsymbol{\theta}_j}\right), \tag{15}$$

where $\boldsymbol{\alpha} = K_{\mathbf{y}}^{-1}\mathbf{y}$. This enables an efficient gradient based optimizing routine for maximizing (14). In this work the BFGS method (Nocedal and Wright, 1999) has been used.

4 Modeling

The GP framework will now be combined with the electromagnetic theory to construct a model for jointly estimating the **B**- and the **M**-field using a three-axis magnetometer. We assume that the measurements of the magnetic field are corrupted with Gaussian noise

$$\mathbf{y}_{\mathbf{B},k} = \mathbf{f}_{\mathbf{B}}(\mathbf{u}_k) + \mathbf{e}_{\mathbf{B},k}, \qquad \mathbf{e}_{\mathbf{B},k} \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 I_3), \tag{16a}$$

where $\mathbf{y}_{\mathbf{B},k}$ is a three-axis magnetometer measurement transformed into world coordinates and $\mathbf{f}_{\mathbf{B}}(\mathbf{u}_k)$ is a function being equal to the $\mathbf{B}/\mu_0$-field (the **B**-field normalized with μ_0) at location $\mathbf{u}_k$. As discussed in Item 1 in Section 2 we also know that the **M**-field is zero at location $\mathbf{u}_k$, where the measurement $\mathbf{y}_{\mathbf{B},k}$ was collected. This information is incorporated by considering a noise free measurement $\mathbf{y}_{\mathbf{M},k} = \mathbf{0}$ with the following measurement equation

$$\mathbf{y}_{k,\mathbf{M}} = \mathbf{f}_{\mathbf{M}}(\mathbf{u}_k) = \mathbf{f}_{\mathbf{B}}(\mathbf{u}_k) - \mathbf{f}_{\mathbf{H}}(\mathbf{u}_k), \tag{16b}$$

where $\mathbf{f}_{\mathbf{M}}(\mathbf{u}_k)$ and $\mathbf{f}_{\mathbf{H}}(\mathbf{u}_k)$ are functions corresponding to the **M**- and the **H**-field and where we have used the coupling given by (2). Note that this coupling is the key equation for our model since it enables us to jointly estimate the **B**-field as well as the **M**-field in contrast to prior work.

We put this into a statistical framework by considering $\mathbf{f}_{\mathbf{B}}$ and $\mathbf{f}_{\mathbf{H}}$ (and consequently also $\mathbf{f}_{\mathbf{M}}$ via (16b)) to be GPs. Following the discussion in Item 2 in Section 2 we consider $\mathbf{f}_{\mathbf{B}}$ and $\mathbf{f}_{\mathbf{H}}$ to have a common constant mean function (corresponding to the earth magnetic field) and we use the covariance functions given in (8) and (9) to preserve the divergence- and curl-free properties of $\mathbf{f}_{\mathbf{B}}$ and $\mathbf{f}_{\mathbf{H}}$ according to the field equations (1). This gives

$$\mathbf{f}_{\mathbf{B}} \sim \mathcal{GP}\big(\boldsymbol{\beta}, K_{\mathbf{B}}(\mathbf{u}, \mathbf{u}')\big), \quad \mathbf{f}_{\mathbf{H}} \sim \mathcal{GP}\big(\boldsymbol{\beta}, K_{\mathbf{H}}(\mathbf{u}, \mathbf{u}')\big), \tag{16c}$$

$$\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \sigma_\beta^2 I_3), \tag{16d}$$

where we have used a Gaussian prior on the unknown mean β.

The model (16) can be reformulated into the standard model description outlined in Section 3

$$\mathbf{y}_k = \mathbf{f}(\mathbf{u}_k) + \mathbf{e}_k, \tag{17a}$$

$$\mathbf{f}(\mathbf{u}) \sim \mathcal{GP}\Big(\mathbf{0}, K(\mathbf{u}, \mathbf{u}')\Big), \qquad \mathbf{e}_k \sim \mathcal{N}(\mathbf{0}, \Sigma), \tag{17b}$$

by augmenting the measurements and the noise covariance matrices

$$\mathbf{y}_k = \begin{bmatrix} \mathbf{y}_{\mathbf{B},k} \\ \mathbf{y}_{\mathbf{M},k} \end{bmatrix} \quad \text{and} \quad \Sigma = \begin{bmatrix} \sigma_n^2 I_3 & 0_3 \\ 0_3 & 0_3 \end{bmatrix} \tag{17c}$$

as well as the outputs of the functions that we want to learn

$$\mathbf{f}(\mathbf{u}) = \begin{bmatrix} \mathbf{f}_{\mathbf{B}}(\mathbf{u}) \\ \mathbf{f}_{\mathbf{M}}(\mathbf{u}) \end{bmatrix} = \begin{bmatrix} I_3 & 0_3 \\ I_3 & -I_3 \end{bmatrix} \begin{bmatrix} \mathbf{f}_{\mathbf{B}}(\mathbf{u}) \\ \mathbf{f}_{\mathbf{H}}(\mathbf{u}) \end{bmatrix} \sim \mathcal{GP}(\mathbf{0}, K), \tag{17d}$$

where $K = K(\mathbf{u}, \mathbf{u}')$. The augmented function $\mathbf{f} : \mathbb{R}^3 \to \mathbb{R}^6$ will then have the covariance function

$$K = \begin{bmatrix} K_{\mathbf{B}} + \sigma_\beta^2 I_3 & K_{\mathbf{B}} \\ K_{\mathbf{B}} & K_{\mathbf{B}} + K_{\mathbf{H}} \end{bmatrix}, \tag{17e}$$

where the relation $\mathbf{f}(\mathbf{u}) \sim \mathcal{GP}(\mathbf{0}, K) \Rightarrow C\mathbf{f}(\mathbf{u}) \sim \mathcal{GP}(\mathbf{0}, CKC^\mathsf{T})$ has been used as well as (5) to reformulate this as a zero mean GP. Finally, we encode $\boldsymbol{\theta} \triangleq [\log \sigma_f^2 \;\; \log l^2 \;\; \log \sigma_\beta^2 \;\; \log \sigma_n^2]$, where the logarithm ensures the positiveness of σ_f^2, l^2, σ_β^2 and σ_n^2.

5 Results

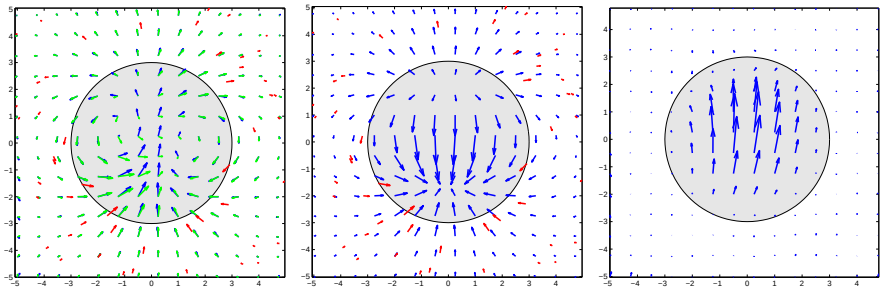
The ability of the proposed model to model magnetic fields will be evaluated by using a simulated data set as well as a real world data set. The results will be reported in this section.

5.1 Simulated Experiment

The setup with a uniformly magnetized sphere presented in Example 1 is used to estimate the **B**-, **H**- and **M**-field given in Figure 1. Consider a sphere centered at the origin with radius 3 m having a uniform magnetization of $\mathbf{M} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^\mathsf{T} \mathrm{A\,m^{-1}}$. In total $N = 50$ training inputs are chosen from a region outside the sphere and inside a square with dimension $10\,\mathrm{m} \times 10\,\mathrm{m}$ aligned with the xy-plane, which also is centered at the origin. For each training input the corresponding training output is computed using the true field perturbed with Gaussian noise having a standard deviation of $\sigma_n = 0.01$. The test inputs are chosen from a grid xy-plane with an interval of 0.75 m. The estimated magnetic field at these test inputs is then compared with the true magnetic fields. Both

the SE kernel (7) and the proposed kernel (17) are applied to the data, where the hyperparameters for each kernel are estimated as described in Section 3.4. The results are given in Figure 2.

Both the SE kernel and the proposed kernel (17) are able to reproduce the character of the true **B**-field as given in Figure 1a. By comparing the estimated **B**-field with the true **B**-field, the proposed covariance function is only slightly better with a root mean square error of $0.33\,\mathrm{A\,m^{-1}}$, whereas the corresponding number for the SE covariance function is $0.38\,\mathrm{A\,m^{-1}}$. However, the great advantage with the proposed covariance function is its ability to estimate the **M**-field as shown in Figure 2c, which resembles the true **M**-field in Figure 1c. Both the location of the magnetic source and the direction of its magnetization are correctly captured.



(a) *Estimated* **B***-field* ***(b)*** *Estimated* **H***-field* ***(c)*** *Estimated* **M***-field*

Figure 2: *Estimated fields induced by a uniformly magnetized sphere (see Example 1) using our proposed kernel* (17) *(blue) and the* SE *kernel* (6) *(green) together with the training data (red).*

(a) *Estimated shape of table* ***(b)*** *Real shape of table*

Figure 3: *Estimated magnetic content in a table turned upside down.*

5.2 Real World Experiment

A real world experiment was conducted in a magnetic environment consisting of a table with metal frame turned upside down as displayed in Figure 3b. A three axis magnetometer has been used to measure the magnetic field at various locations around that table and the position and the orientation of the magnetometer unit was measured using an optical reference system (Vicon). The magnetometer measurements were then transformed into world coordinates using the orientation provided by the reference. This data has been downsampled to 2 Hz to reduce the number of data points. Together with the position estimate from the reference this comprises the training data as displayed in Figure 4. For this dataset the hyperparameters have not been estimated but rather tuned to $\sigma_f = 0.3$, $l = 0.15$, $\sigma_b = 1$ and $\sigma_n = 0.3$ for reasons discussed below. In Figure 3a the region of the **M**-field which exceeds 30% of the maximal estimated **M**-field is displayed. This estimated magnetic map has visual similarities with the real table in Figure 3b. All four table legs can be distinguished as well as the frame on which the table top is attached.

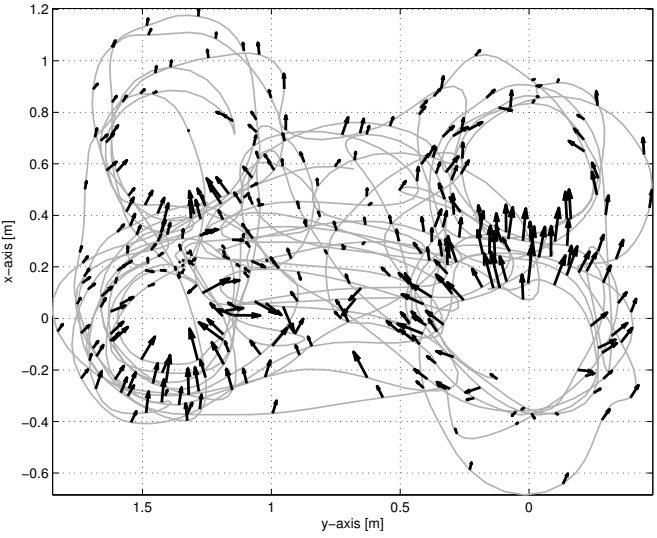


Figure 4: *The training data in the real world experiment seen from above together with the trajectory that the magnetometer has followed.*

The proposed GP (as any other stationary GP) is restricted to using the same set of hyperparameters for all data. This is problematic in environments which have different characteristic length scales and signal amplitudes in different regions in space. When estimating the hyperparameters in the proposed manner using data collected in such environments, the result might not be sound. The hyperparameters have therefore been considered as tuning parameters.

6 Conclusion and Future Work

We have introduced a Bayesian nonparametric model for jointly estimating both the magnetic field and the magnetic sources. The model is based on a vector-valued stationary Gaussian process (GP) with a covariance function exploiting the divergence- and curl-free properties of the magnetic field derived from the electromagnetic theory. The model has been compared with a component-wise GP proposed by Vallivaara et al. (2010) for modeling magnetic fields. In the comparison only a small improvement in estimation performance could be reported. However, the great advantage of the proposed method is its ability to also model the magnetic sources in a nonparametric manner, which has been illustrated using both simulated and real world data.

In future work we will extend our nonparametric model to handle more complex environments. One promising idea is to use a multiplicity of GPs governed by a hierarchical Dirichlet process (Teh et al., 2006). Our final target is a full SLAM framework.

Bibliography

M. A. Álvarez, L. Rosasco, and N. D. Lawrence. Kernels for vector-valued functions: A review. *Foundations and Trends in Machine Learning*, 4(3):195–266, March 2012.

L. Baldassarre, L. Rosasco, A. Barla, and A. Verri. Vector field learning via spectral filtering. *Machine Learning and Knowledge Discovery in Databases*, pages 56–71, 2010.

J. Chung, M. Donahoe, C. Schmandt, I. Kim, P. Razavai, and M. Wiseman. Indoor location sensing using geo-magnetism. In *Proceedings of the 9th International Conference on Mobile systems, Applications, and Services*, pages 141–154, Bethesda, USA, June 2011.

E. J. Fuselier Jr. *Refined error estimates for matrix-valued radial basis functions*. PhD thesis, Texas A&M University, 2006.

J. D. Jackson. *Classical Electrodynamics*. John Wiley & Sons, Inc, 3rd edition, 1998.

A. Kemppainen, J. Haverinen, I. Vallivaara, and J. Röning. Near-optimal Exploration in Gaussian Process SLAM: Scalable Optimality Factor and Model Quality Rating. In *Proceedings of European Conference on Mobile Robots*, pages 283–290, Örebro, Sweden, September 2011.

M. Kok, N. Wahlström, T. B. Schön, and F. Gustafsson. MEMS-based inertial navigation based on a magnetic field map. In *Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 6466–6470, Vancouver, Canada, May 2013.

I. Macêdo and R. Castro. Learning divergence-free and curl-free vector fields with matrix-valued kernels. Technical report, Technical report, Instituto Nacional de Matematica Pura e Aplicada, 2008.

J. C. Maxwell. A dynamical theory of the electromagnetic field. *Philosophical Transactions of the Royal Society of London*, 155:459–512, 1865.

J. Murphy and S. Godsill. Simultaneous localization and mapping for non-parametric potential field environments. In *Proceedings of the 15th International Conference on Information Fusion*, Singapore, July 2012.

F. J. Narcowich and J. D. Ward. Generalized hermite interpolation via matrix-valued conditionally positive definite functions. *Mathematics of Computation*, 63(208):661–688, 1994.

J. Nocedal and S. J. Wright. *Numerical optimization*. Springer verlag, 1999.

C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.

Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

I. Vallivaara, J. Haverinen, A. Kemppainen, and J. Roning. Simultaneous localization and mapping using ambient magnetic field. In *Proceedings of the IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 14 –19, Salt Lake City, USA, September 2010.

I. Vallivaara, J. Haverinen, A. Kemppainen, and J. Roning. Magnetic field-based SLAM method for solving the localization problem in mobile robot floor-cleaning task. In *Proceedings of the 15th International Conference on Advanced Robotics (ICAR)*, pages 198–203, Tallin, Estonia, June 2011.

D. Vissière, A. P. Martin, and N. Petit. Using magnetic disturbances to improve IMU-based position estimation. In *Proceedings of the European Control Conference*, pages 2853–2858, Kos, Greece, July 2007.

N. Wahlström, M. Kok, T. B. Schön, and F. Gustafsson. Modeling magnetic fields using Gaussian processes. In *Proceedings of the the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 3522–3526, Vancouver, Canada, May 2013.

H. Zhang and F. Martin. Robotic mapping assisted by local magnetic field anomalies. In *Proceedings of the IEEE Conference on Technologies for Practical Robot Applications (TePRA)*, pages 25–30, Woburn, USA, April 2011.

Paper D

Extended Target Tracking Using Gaussian Processes

Authors: Niklas Wahlström, and Emre Özkan

Edited version of the paper:

N. Wahlström and E. Özkan. Extended target tracking using Gaussian processes. *IEEE Transactions on Signal Processing*, 63(16):4165–4178, 2015.

Extended Target Tracking Using Gaussian Processes

Niklas Wahlström, and Emre Özkan

Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
{nikwa,emre}@isy.liu.se

Abstract

In this paper, we propose using Gaussian processes to track an extended object or group of objects, that generates multiple measurements at each scan. The shape and the kinematics of the object are simultaneously estimated, and the shape is learned online via a Gaussian process. The proposed algorithm is capable of tracking different objects with different shapes within the same surveillance region. The shape of the object is expressed analytically, with well-defined confidence intervals, which can be used for gating and association. Furthermore, we use an efficient recursive implementation of the algorithm by deriving a state space model in which the Gaussian process regression problem is cast into a state estimation problem.

1 Introduction

Target tracking involves estimating the kinematics of an unknown number of objects in a surveillance region based on a set of measurements collected by one or multiple sensors. In the most common formulation of the problem, each object is considered to be a point source, and the measurements are assumed to be generated from the vicinity of the object's center. This assumption will simplify, for example, the computation of the possible association hypothesis between the estimated targets and the available measurements. Thanks to the increasing availability of computational resources, more complex models can now be used for defining object(s), and the inference techniques for such models are drawing considerable interest.

In extended target tracking models, each target is assumed to have an *extent* from which the measurements are generated. The extent of a target can be modeled as a circle, ellipse, rectangle or other simple shapes (Koch, 2008; Feldmann et al., 2011; Granström et al., 2011). Several models have been proposed in the literature for extended object tracking.

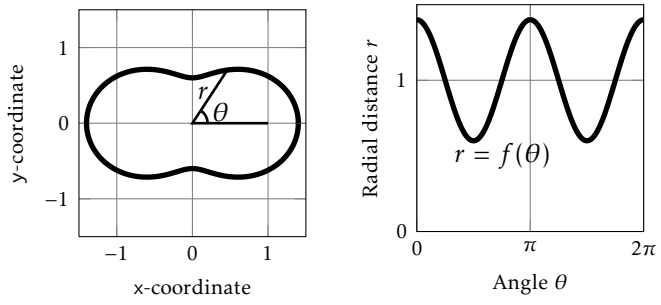
A Bayesian approach was first proposed in Koch (2008) for elliptical extended targets, where the inverse Wishart distribution is used as a prior for the unknown

elliptical target extent. However, in this model, there exists a coupling between the target extent and the target kinematic state, which is restrictive. This issue was later addressed in Feldmann et al. (2011), where an approximate inference method was proposed for using random matrices in extended target tracking. In Orguner (2012), an approximate measurement update that relies on a variational Bayes approximation was proposed for the random matrix-based extended target tracking. Two different random matrix-based models were proposed in Lan and Li (2012a). The extension of these models to multiple ellipses was presented in Lan and Li (2012b). In a recent report (Granström and Orguner, 2014), a new prediction update for random matrix-based extended target models was proposed. The method proposed in that work focuses on the time update, and the possible rotation of the target extent is taken into account. In Granström et al. (2011), rectangular and elliptical extended objects were considered, and an extended Kalman filter was used for inference. The target extent can also be defined as a parametric surface generating measurements (Baum and Hanebeck, 2009, 2011). Trackers for the more general class of star-convex shapes were introduced by Baum and Hanebeck (2009) and further discussed in Sun et al. (2012), Baum (2013) and Baum and Hanebeck (2014). Image-based contour trackers have also been proposed and discussed in the literature. Among the different parameterizations of the contour, B-splines have been the most common. Using Kalman filters for tracking both the shape and position of such contours has been proposed (Blake et al., 1993, 1995; Li et al., 2004). However, particle filter solutions are preferred to achieve robust trackers (Isard and Blake, 1998; Li et al., 2003).

In this article, we propose using Gaussian processes (GPs) to model the boundary of an unknown object. GPs have been widely used by the machine learning, statistics, and signal processing communities for identification, classification, and regression (Rasmussen and Williams, 2006) because of their tractable posterior computation and attractive analytical properties. The model proposed herein is flexible enough to represent a large variety of shapes and provides an analytical representation of the objects' extents. The boundaries and the measurement predictions are more precise than rough elliptical approximations. We believe that the assumptions of point targets or elliptical extents are restrictive and that tracking objects with unknown and complex shapes is becoming more important with the increasing accuracy and resolution of the sensors. The ability to learn and track unknown shapes also provides better accuracy and a priori information for the detection algorithms in which the measurements (features of interest) are extracted from raw sensor data, e.g., images. Accurate knowledge of the object's shape, which is summarized into an analytical expression, is also a critical element for target tracking algorithms in which the association between the measurements and the targets has a crucial role on the performance of the tracker. Furthermore, the ability to learn the shapes of the targets can be used for classifying and extracting the attribute information of the targets.

2 Target Extent Model

In conventional tracking methods, targets are considered to be point sources that result in sensor detections. Moreover, in most cases, they are assumed to generate at most one measurement per scan (Bar-Shalom and Fortmann, 1987). In contrast to these traditional assumptions, we will assume that each target can generate multiple measurements from multiple sources on its extent, e.g., multiple reflection points can be detected on the same object by a radar. The object extent can be modeled as a simple geometric primitive, such as a rectangle or an ellipse. In this study, the target extent will be described via *star-convex* shapes. A set $\mathcal{S}(x_k)$ is called star-convex if each line segment from the center to any point is fully contained in $\mathcal{S}(x_k)$. By definition, convex sets are subsets of star-convex sets. The contour of star-convex shapes can be described in polar coordinates with a radial function $r = f(\theta)$ that maps the angle to the radius, which is convenient for representing and learning abstract shapes. An example shape is shown in Figure 1.



(a) *Star-convex target shape* ***(b)*** *Radial function*

Figure 1: *Description of star-convex shapes using a radial function $r = f(\theta)$.*

Based on the star-convex description, two different measurement models will be considered in this work

1. **Target contour model:** The measurement equation for the noisy detections originating from the target contour can be written as

$$\mathbf{y}_{k,l} = \mathbf{r}_k + \mathbf{p}(\theta_{k,l}) f(\theta_{k,l}) + \mathbf{e}_{k,l}, \tag{1}$$

where $\mathbf{r}_k$ is the target position at time index k; $\{\mathbf{y}_{k,l}\}_{l=1}^{n_k}$ are the n_k measurements collected at time index k; $\{\theta_{k,l}\}_{l=1}^{n_k}$ denote the angles describing the origin of these measurements on the target contour; $\mathbf{e}_{k,l} \sim \mathcal{N}(0,\ R)$ represents the zero-mean measurement noise, which is assumed to be Gaussian with covariance R; and $\mathbf{p}(\theta_{k,l})$ is an orientation vector defined as

$$\mathbf{p}(\theta_{k,l}) \triangleq \begin{bmatrix} \cos(\theta_{k,l}) \\ \sin(\theta_{k,l}) \end{bmatrix}. \tag{2}$$

2. **Target surface model:** If the measurements are assumed to originate from the interior of the target region, the model (1) can be extended by considering

$$\mathbf{y}_{k,l} = \mathbf{r}_k + s_{k,l}\mathbf{p}(\theta_{k,l})f(\theta_{k,l}) + \mathbf{e}_{k,l}, \tag{3}$$

where the scaling component is $s_{k,l} \in [0, 1]$.

By augmenting a parametrized version of the unknown radial function $r = f(\theta)$ with the target position $\mathbf{r}_k$ and its kinematics, in Section 5, we will derive a state space model that enables the simultaneous estimation of the target position and its shape. An orientation state ψ_k will also be added to track orientation changes of the target. The surface model is further described in Section 8. Additionally, note that by parameterizing star-convex shapes in this manner, the target position $\mathbf{r}_k$ and target contour $f(\theta)$ will not be unique. Different combinations of $\mathbf{r}_k$ and $f(\theta)$ can provide the same target contour.

Star-convex shapes have been introduced in the context of target tracking by Baum and Hanebeck (2009) and further elaborated in Sun et al. (2012), Baum (2013) and Baum and Hanebeck (2014). In all these contributions, a Fourier series expansion was used to parametrize the unknown radial function $f(\theta)$. This approach provides great flexibility and is also a standard choice for describing periodic signals. However, in a stochastic setting, this approach has a number of limitations. In this work, we will instead investigate the use of GPs to model the radial function. In contrast to the Fourier series expansion in its basic form, the GP is a probabilistic model that allows the specification of the posterior distribution of the learned function in a natural way. Furthermore, it is defined in the spatial domain rather than in the frequency domain, which enables local learning of the target contour. In other words, while learning the observable parts of the target extent, the uncertainty around the unobserved sections can be maintained, which provides more accurate gates for future observations.

Unfortunately, the GP regression is a batch (off-line) method that requires all the data to be available prior to the inference. To enable the simultaneous estimation of the kinematic state and the extent, we will seek approximations and formulate the GP model as a state space model. By augmenting this state space model with the state space description of the kinematic states (position, orientation and velocities), an extended Kalman filter is used to simultaneously estimate all parameters. Prior to introducing the details of this special inference technique, the standard GP regression and its extension will be explained in the following section.

Throughout this paper, scalars or scalar-valued functions are denoted with non-bold symbols, e.g., θ, vectors or vector-valued functions are denoted with bold symbols, e.g., $\mathbf{y}$, and matrices are denoted with capitalized symbols, e.g., P. Furthermore, Cartesian coordinates are denoted using Sans-serif font, e.g., x and y, to distinguish them from other variables.

3 Gaussian Processes

A Gaussian process (GP) (Rasmussen and Williams, 2006) is a stochastic process suitable for modeling spatially correlated measurements. GPs can be considered a distribution over functions. This distribution is uniquely defined with its mean function $\mu(u)$ and covariance function $k(u, u')$ of a function $f(u)$ as

$$\mu(u) = \mathbb{E}[f(u)], \tag{4a}$$

$$k(u, u') = \mathbb{E}[(f(u) - \mu(u))(f(u') - \mu(u'))^{\mathsf{T}}], \tag{4b}$$

and the Gaussian process is denoted as

$$f(u) \sim \mathcal{GP}\Big(\mu(u), k(u, u')\Big), \tag{5}$$

where u is the function input.

The GP is a generalization of the multivariate Gaussian probability distribution in the sense that the function values evaluated for a finite number of inputs $u_1, \ldots, u_N$ are normally distributed

$$\begin{bmatrix} f(u_1) \\ \vdots \\ f(u_N) \end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}, K), \quad \text{where} \quad \boldsymbol{\mu} = \begin{bmatrix} \mu(u_1) \\ \vdots \\ \mu(u_N) \end{bmatrix}, \quad K = \begin{bmatrix} k(u_1, u_1) & \cdots & k(u_1, u_N) \\ \vdots & & \vdots \\ k(u_N, u_1) & \cdots & k(u_N, u_N) \end{bmatrix}. \tag{6}$$

In this work, our model will be formulated with a zero-valued mean function. The following description will be based on that assumption. However, the formulation can easily be generalized to a non-zero mean function; see Rasmussen and Williams (2006) for further details.

3.1 Gaussian Process Regression

The GP model is primarily used to incorporate training data to learn an unknown function. Consider the following measurement model

$$y_k = f(u_k) + e_k, \qquad e_k \sim \mathcal{N}(0, R), \tag{7}$$

where y_k is a noisy measurement of the function $f(\cdot)$ at the training input u_k and e_k is the measurement noise. The objective is to use a set of measurements $\mathbf{y} \triangleq \begin{bmatrix} y_1 & \ldots & y_N \end{bmatrix}^{\mathsf{T}}$ together with their corresponding inputs $\mathbf{u} \triangleq \begin{bmatrix} u_1 & \ldots & u_N \end{bmatrix}^{\mathsf{T}}$ to learn the function values $\mathbf{f} \triangleq \begin{bmatrix} f(u_1^{\mathbf{f}}) & \ldots & f(u_{N^{\mathbf{f}}}^{\mathbf{f}}) \end{bmatrix}^{\mathsf{T}}$ for other test inputs $\mathbf{u}^{\mathbf{f}} \triangleq \begin{bmatrix} u_1^{\mathbf{f}} & \ldots & u_N^{\mathbf{f}} \end{bmatrix}^{\mathsf{T}}$. In the same manner as in (6), the joint distribution for the measurements $\mathbf{y}$ and the function values $\mathbf{f}$ is

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(\mathbf{u}, \mathbf{u}) + I_N \otimes R & K(\mathbf{u}, \mathbf{u}^{\mathbf{f}}) \\ K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}) & K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}}) \end{bmatrix}\right), \tag{8}$$

where $\otimes$ denotes the Kronecker product and

$$K(\mathbf{u}, \mathbf{u}^{\mathbf{f}}) = \begin{bmatrix} k(u_1, u_1^{\mathbf{f}}) & \dots & k(u_1, u_{N^{\mathbf{f}}}^{\mathbf{f}}) \\ \vdots & & \vdots \\ k(u_N, u_1^{\mathbf{f}}) & \dots & k(u_N, u_{N^{\mathbf{f}}}^{\mathbf{f}}) \end{bmatrix}. \tag{9}$$

From the joint Gaussian distribution $p(\mathbf{y}, \mathbf{f})$ in (8), the conditional distribution $p(\mathbf{f}|\mathbf{y})$ can easily be computed as

$$p(\mathbf{f}|\mathbf{y}) = \mathcal{N}(A\mathbf{y}, P), \tag{10a}$$

where

$$A = K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}) K_y^{-1}, \tag{10b}$$

$$P = K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}}) - K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}) K_y^{-1} K(\mathbf{u}, \mathbf{u}^{\mathbf{f}}), \tag{10c}$$

$$K_y = K(\mathbf{u}, \mathbf{u}) + I_N \otimes R. \tag{10d}$$

To compute (10b)-(10c), a Cholesky decomposition of K_y is preferred rather than computing its matrix inversion explicitly because it is faster and numerically more stable.

3.2 Recursive Gaussian Process Regression

In many applications (for example, target tracking), all the measurements may not be available as a batch, but they might be collected sequentially in time. In such cases, one should aim for recursive solutions for efficient implementation and online inference. In this setting, at each time index k, we are interested in computing the posterior $p(\mathbf{f}|y_{1:k})$ online. For such applications, it is not feasible to use the standard GP regression as presented in the previous section because of the following two reasons:

First, the formulation in (10) is a batch formulation in which all data $y_{1:k}$ are needed to perform inference and cannot be used to update the posterior recursively. Second, the complexity of the regression problem increases cubically with the number of measurements, which is not feasible for an online implementation. We are therefore aiming for an approximate recursive update of the posterior.

In the literature, a few methods have been proposed for recursive GP regression. In Osborne (2010), the recursive implementation is based on a sequential update of the Cholesky factor of the matrix K_y. In this manner, the full Cholesky decomposition does not have to be recomputed each time a new measurement is received. Hartikainen and Särkkä (2010) formulate the GP as a state space model and solve the regression problem with a Kalman filter. This approach only works for one-dimensional inputs, and the measurements are required to be processed in a sequential order with respect to the input dimension. In Huber (2013, 2014), the GP is approximated with a finite number of basis points, which are updated by further approximations.

In this work, we present a recursive method that resembles the one in Huber (2014). Similar to Huber (2013), we explicitly define the recursions as a state space model to which we can apply a Kalman filter. This state space description will be beneficial in extended target tracking because it can be reformulated for augmentation with other state space models, for example, a model that describes the dynamics of target position and target orientation.

We interpret the test inputs $\mathbf{u}^{\mathbf{f}}$ to be the basis points and $\mathbf{f}$ to be their corresponding outputs. By applying Bayes law consecutively on the posterior $p(\mathbf{f}|y_{1:N})$, we obtain

$$p(\mathbf{f}|y_{1:N}) \propto p(y_N|\mathbf{f}, y_{1:N-1})p(\mathbf{f}|y_{1:N-1}) \propto \cdots \underbrace{p(y_k|\mathbf{f}, y_{1:k-1}) \cdots p(\mathbf{f})}_{\propto p(\mathbf{f}|y_{1:k})}, \tag{11}$$

which results in the following recursion

$$\begin{aligned} p(\mathbf{f}|y_{1:k}) &\propto p(y_k|\mathbf{f}, y_{1:k-1}) \times p(\mathbf{f}|y_{1:k-1}). \\ \text{posterior} \propto \quad &\text{likelihood} \quad \times \quad \text{prior} \end{aligned} \tag{12}$$

We will now approximate $\mathbf{f}$ to be conditionally independent of the past measurements $y_{1:k-1}$, which means that $\mathbf{f}$ will be the sufficient statistic of all the past measurements

$$p(y_k|\mathbf{f}, y_{1:k-1}) \approx p(y_k|\mathbf{f}). \tag{13}$$

This approximation would be exact if the input values for $y_{1:k-1}$ were a subset of the input values for $\mathbf{f}$, and it would be a good approximation if the input values for $y_{1:k-1}$ were close to those of $\mathbf{f}$ relative to the characteristic length scale of the covariance function. In our application, the inputs are angles in the interval $[0, 2\pi]$. Because this interval is bounded, it can be adequately covered by a small number of basis points placed equidistantly within that interval.

As in (8), the measurement y_k and the function values $\mathbf{f}$ are jointly Gaussian

$$\begin{bmatrix} y_k \\ \mathbf{f} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} k(u_k, u_k) + R & K(u_k, \mathbf{u}^{\mathbf{f}}) \\ K(\mathbf{u}^{\mathbf{f}}, u_k) & K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}}) \end{bmatrix}\right). \tag{14}$$

Furthermore, we formulate the likelihood and the initial prior in the same manner as (10)

$$p(y_k|\mathbf{f}) = \mathcal{N}\left(y_k;\ H_k^{\mathbf{f}}\mathbf{f},\ R_k^{\mathbf{f}}\right), \tag{15a}$$

$$p(\mathbf{f}) = \mathcal{N}\left(0,\ P_0^{\mathbf{f}}\right), \tag{15b}$$

with

$$H^{\mathbf{f}}(u_k) = K(u_k, \mathbf{u}^{\mathbf{f}})[K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}})]^{-1}, \tag{16a}$$

$$\begin{aligned} R^{\mathbf{f}}(u_k) &= k(u_k, u_k) + R - K(u_k, \mathbf{u}^{\mathbf{f}})[K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}})]^{-1}K(\mathbf{u}^{\mathbf{f}}, u_k), \\ P_0^{\mathbf{f}} &= K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}}). \end{aligned} \tag{16b}$$

By exploiting the structure of this likelihood, the recursive regression can be computed by implementing a Kalman filter on the following state space model

$$\mathbf{x}^{\mathbf{f}}_{k+1} = \mathbf{x}^{\mathbf{f}}_{k}, \tag{17a}$$

$$y_k = H^{\mathbf{f}}(u_k)\mathbf{x}^{\mathbf{f}}_{k} + e^{\mathbf{f}}_{k}, \quad e^{\mathbf{f}}_{k} \sim \mathcal{N}\left(0,\ R^{\mathbf{f}}(u_k)\right), \tag{17b}$$

$$\mathbf{x}^{\mathbf{f}}_{0} \sim \mathcal{N}\left(0,\ P^{\mathbf{f}}_{0}\right), \tag{17c}$$

where $\mathbf{x}^{\mathbf{f}}_{k} = \mathbf{f} = \begin{bmatrix} f(u^{\mathbf{f}}_{1}) & \dots & f(u^{\mathbf{f}}_{N^{\mathbf{f}}}) \end{bmatrix}^{\mathsf{T}}$ is interpreted as the state.

The state space model not only enables efficient inference but also allows us to design the model for various purposes. First, a dynamical behavior can be added to the extent very easily. In the case where the unknown function changes over time, a process noise term can be added to the dynamics as follows:

$$\mathbf{x}^{\mathbf{f}}_{k+1} = F^{\mathbf{f}}\mathbf{x}^{\mathbf{f}}_{k} + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}\left(0,\ Q^{\mathbf{f}}\right), \tag{18}$$

with

$$F^{\mathbf{f}} = e^{-\alpha T} I, \quad Q^{\mathbf{f}} = \left(1 - e^{-2\alpha T}\right) K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}}). \tag{19}$$

The parameter $\alpha \geq 0$ will determine the speed of the dynamics and can be considered a forgetting factor. With $\alpha = 0$, all measurements that have been collected will be of equal importance, and as α increases, less weight is given to older measurements. The choice of dynamics in (19) ensures that the stationary state covariance is $K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}})$, regardless of the α value, because

$$P = F^{\mathbf{f}} P (F^{\mathbf{f}})^{\mathsf{T}} + Q^{\mathbf{f}} \quad \Rightarrow \quad P = K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}}). \tag{20}$$

Furthermore, the state space description will allow us to augment the model with other state space models for joint estimation. This will be presented in Section 5, where the kinematic state is augmented with the target extent.

4 Target Contour GP Model

As described earlier, our aim is to describe the target contour using a GP model. The input to the model is therefore chosen to be the polar angle $\theta = u$, and the output is the radius $r = y$, as shown in Figure 1b. The required mean and covariance functions that define the GP model will be described in the following subsections.

4.1 Mean Function

In this work, we will consider a constant but unknown mean function $\mu(\theta) = r$, which is interpreted as being the mean radius of the target contour.

$$f(\theta) \sim \mathcal{GP}\Big(r, k(\theta, \theta')\Big), \quad \text{where} \quad r \sim \mathcal{N}\left(0,\ \sigma_r^2\right). \tag{21a}$$

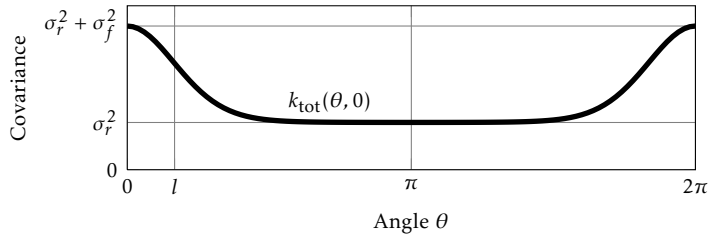


Figure 2: *Periodic covariance function $k_{tot}(\theta, \theta')$ given in* (25)*. The kernel is used for modeling the covariance of the radial extent.*

By integrating out the parameter r, the same model can be reformulated as a zero mean GP

$$f(\theta) \sim \mathcal{GP}\Big(0, k(\theta, \theta') + \sigma_r^2\Big). \tag{21b}$$

4.2 Covariance Function

The covariance function is the crucial component when modeling a GP. This function encodes the assumptions that we make on the functions to be learned. The most common choice is the squared exponential (SE) covariance function (Rasmussen and Williams, 2006)

$$k(\theta, \theta') = \sigma_f^2 e^{-\frac{|\theta-\theta'|^2}{2l^2}}, \tag{22}$$

where σ_f^2 is the prior variance of the signal amplitude and l is the length scale of the function that we want to learn. This covariance function provides high correlations for two radial distances $f(\theta)$ and $f(\theta')$ if their corresponding angles θ and θ' are close to each other and less correlation if they are further apart.

To encode the periodicity of $f(\cdot)$ in terms of the angle θ, (22) is slightly modified as

$$k(\theta, \theta') = \sigma_f^2 e^{-\frac{2\sin^2\left(\frac{|\theta-\theta'|}{2}\right)}{l^2}}, \tag{23}$$

which can be derived using a non-linear mapping of the input (MacKay, 1998). With this modification, $f(\theta)$ and $f(\theta + 2\pi)$ will be perfectly correlated as desired because

$$\rho(f(\theta), f(\theta + \pi)) = \frac{k(\theta, \theta + 2\pi)}{\sqrt{k(\theta, \theta)}\sqrt{k(\theta + 2\pi, \theta + 2\pi)}} = 1. \tag{24}$$

Finally, the contribution from the mean function as described in Section 4.1 is included in the covariance function, resulting in

$$k_{\text{tot}}(\theta, \theta') = \sigma_f^2 e^{-\frac{2\sin^2\left(\frac{|\theta-\theta'|}{2}\right)}{l^2}} + \sigma_r^2. \tag{25}$$

The final covariance function is shown in Figure 2. By design, the model function has a periodicity of 2π because $k_{\text{tot}}(\theta+2\pi,\theta') = k_{\text{tot}}(\theta,\theta')$. Additionally, note that the radius for different angles is always positively correlated, and the correlation increases if the angular distance is shorter. Furthermore, because we treat the radius r as a random variable, this approach will allow us to describe objects of various sizes.

4.3 Further Extensions

The covariance function (25) will be used throughout the results section in this work. However, to illustrate the flexibility of the GP modeling, some examples will be presented below on how further assumptions on the target shape could be included.

Symmetric Model

For many targets, there are symmetry assumptions on the shape of the target. Designing the covariance function to have a period of π rather than 2π gives

$$k(\theta,\theta') = \sigma_f^2 e^{-\frac{\sin^2(|\theta-\theta'|)}{2l^2}}. \tag{26}$$

By using this covariance function, only the functions where $f(\theta) = f(\theta+\pi)$ will be learned. Such an assumption will be valid for target shapes that do possess two lines of symmetry (symmetric left-right and back-forward). This would, for example, be valid for shapes S1 and S2 in Figure 4 but not for shape S3.

Conservative Model

If we know that the target has a certain predefined radius r_0, such information can be incorporated using a non-zero mean Gaussian prior on $r \sim \mathcal{N}(r_0, \sigma_r^2)$ in (21), giving

$$f(\theta) \sim \mathcal{GP}\big(r_0, k(\theta,\theta') + \sigma_r^2\big). \tag{27}$$

Furthermore, by selecting σ_r and σ_f to be small, the target radius can be kept close to radius r_0, resulting in a conservative and robust version of the algorithm. This property can be desired in high clutter scenarios or when the measurements are coming from the target surface, where the measurements are less informative compared to contour measurements. Moreover, note that to handle a non-zero mean $\mu(\theta) = r_0 \neq 0$, the GP regression presented in Sections 3.1 and 3.1 has to be changed accordingly. In particular, this requires performing a non-zero initialization such that $\mathbf{x}_0^{\mathbf{f}} \sim \mathcal{N}(\boldsymbol{\mu}_0^{\mathbf{f}}, P_0)$, where $\boldsymbol{\mu}_0^{\mathbf{f}} = [r_0,\dots,r_0]^\mathsf{T}$.

Explicit Basis Functions

The mean target shape does not necessarily have to be a circle. As a further extension, a set of fixed basis functions $\mathbf{h}(\theta)$ with coefficients $\boldsymbol{\beta}$ can be used to

specify the mean function

$$g(\theta) = f(\theta) + \mathbf{h}(\theta)^\mathsf{T}\boldsymbol{\beta}, \tag{28a}$$

where

$$f(\theta) \sim \mathcal{GP}\Big(0, k(\theta, \theta')\Big) \quad \text{and} \quad \boldsymbol{\beta} \sim \mathcal{N}(\mathbf{b}, B). \tag{28b}$$

Such a model indicates that the target shape can be described with a parametric model given by $\mathbf{h}(\theta)^\mathsf{T}\boldsymbol{\beta}$, with the residual being modeled with a GP. In a manner similar to that used in Section 4.1, by integrating out the coefficients $\boldsymbol{\beta}$, we obtain

$$g(\theta) \sim \mathcal{GP}\Big(\mathbf{h}(\theta)^\mathsf{T}\mathbf{b},\ k(\theta, \theta') + \mathbf{h}(\theta)^\mathsf{T} B \mathbf{h}(\theta')\Big). \tag{28c}$$

5 Augmented State-Space Model

In this section, we will derive the augmented state space model, consisting of the dynamic equation, the measurement equation, and the initial state covariance in the form

$$\mathbf{x}_{k+1} = F\mathbf{x}_k + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}(0,\ Q_k), \tag{29a}$$

$$\mathbf{y}_{k,l} = \mathbf{h}_{k,l}(\mathbf{x}_k) + \mathbf{e}_{k,l}, \quad \mathbf{e}_{k,l} \sim \mathcal{N}(0,\ R_{k,l}), \tag{29b}$$

$$\mathbf{x}_0 \sim \mathcal{N}\left(\boldsymbol{\mu}_0,\ P_0\right), \tag{29c}$$

for joint estimation of the target extent $\mathbf{x}_k^\mathbf{f}$ and the target state $\bar{\mathbf{x}}_k$. We will define the target position $\mathbf{r}_k$, the target orientation ψ_k and the optional additional state $\mathbf{x}_k^*$ separately within the target state vector because they are required in the update equations. Consequently, we consider the augmented state vector

$$\mathbf{x}_k \triangleq \begin{bmatrix}\bar{\mathbf{x}}_k^\mathsf{T} & (\mathbf{x}_k^\mathbf{f})^\mathsf{T}\end{bmatrix}^\mathsf{T}, \qquad \text{where} \tag{30a}$$

$$\bar{\mathbf{x}}_k \triangleq \begin{bmatrix}(\mathbf{r}_k)^\mathsf{T} & \psi_k & (\mathbf{x}_k^*)^\mathsf{T}\end{bmatrix}^\mathsf{T}. \tag{30b}$$

The optional additional state $\mathbf{x}_k^*$ denotes the remaining state variables. In this work, it corresponds to the kinematic state of the target (velocity and angular velocity).

5.1 Measurement Model

Each measurement $\mathbf{y}_{k,l}$ is associated with an angle $\theta_{k,l}^G$ that depends on its angular location relative to the target position $\mathbf{r}_k$

$$\theta_{k,l}^G(\mathbf{r}_k) = \angle(\mathbf{y}_{k,l} - \mathbf{r}_k). \tag{31a}$$

This angle can also be described in the local target coordinate frame using the target orientation state ψ_k

$$\theta_{k,l}^L(\mathbf{r}_k, \psi_k) = \theta_{k,l}^G(\mathbf{r}_k) - \psi_k; \tag{31b}$$

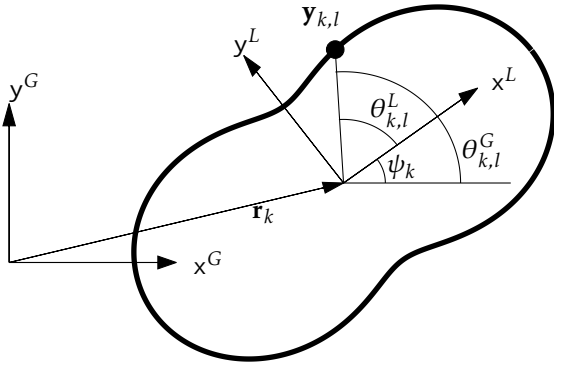


Figure 3: *Each measurement $\mathbf{y}_{k,l}$ is associated with an angle $\theta_{k,l}$ relative to the target center $\mathbf{r}_k$. This angle can either be expressed in global coordinates $\theta^G_{k,l}$ or local coordinates $\theta^L_{k,l}$, where $\mathrm{x}^G/\mathrm{y}^G$ and $\mathrm{x}^L/\mathrm{y}^L$ are the basis vectors of the two coordinate systems, respectively, and ψ_k is the target orientation angle.*

see Figure 3 for a graphical illustration of this geometry. Note that the angles depend on the unknown target position $\mathbf{r}_k$ and the unknown target orientation ψ_k.

The angles can now be used in (1) to describe the relation between the measurement and the state as

$$\mathbf{y}_{k,l} = \mathbf{r}_k + \mathbf{p}_{k,l}(\mathbf{r}_k) f\Big(\theta^L_{k,l}(\mathbf{r}_k, \psi_k)\Big) + \bar{\mathbf{e}}_{k,l}, \tag{32a}$$

where $\bar{\mathbf{e}}_{k,l} \sim \mathcal{N}(0,\ R)$ is the measurement noise. By combining (2) and (31a), the orientation vector can be expressed as

$$\mathbf{p}_{k,l}(\mathbf{r}_k) = \mathbf{p}\Big(\theta^G_{k,l}(\mathbf{r}_k)\Big) = \frac{\mathbf{y}_{k,l} - \mathbf{r}_k}{\|\mathbf{y}_{k,l} - \mathbf{r}_k\|}. \tag{33}$$

Note that the radial function $f(\cdot)$ describes the target extent in the local coordinate frame and the angles $\theta^L_{k,l}(\mathbf{r}_k, \psi_k)$ are the input arguments of that function.

By using the state space description of the GP as derived in Section 3.2, the standard measurement equation can be written as follows:

$$\begin{aligned}
\mathbf{y}_{k,l} &= \mathbf{r}_k + \mathbf{p}_{k,l}(\mathbf{r}_k)\Big[H^{\mathbf{f}}\Big(\theta^L_{k,l}(\mathbf{r}_k, \psi_k)\Big)\mathbf{x}^{\mathbf{f}}_k + \mathbf{e}^{\mathbf{f}}_{k,l}\Big] + \bar{\mathbf{e}}_{k,l} \\
&= \underbrace{\mathbf{r}_k + \tilde{H}_l(\mathbf{r}_k, \psi_k)\mathbf{x}^{\mathbf{f}}_k}_{=\mathbf{h}_{k,l}(\mathbf{x}_k)} + \underbrace{\mathbf{p}_{k,l}(\mathbf{r}_k)e^{\mathbf{f}}_{k,l} + \bar{\mathbf{e}}_{k,l}}_{=\mathbf{e}_{k,l}} && \text{(34a)} \\
&= \mathbf{h}_{k,l}(\mathbf{x}_k) + \mathbf{e}_{k,l}, \qquad \mathbf{e}_{k,l} \sim \mathcal{N}(0,\ R_{k,l}), && \text{(34b)}
\end{aligned}$$

where the measurement model and the new measurement noise covariance are given by

$$\tilde{H}_l(\mathbf{r}_k, \psi_k) = \mathbf{p}_{k,l}(\mathbf{r}_k) H^{\mathbf{f}}\Big(\theta^L_{k,l}(\mathbf{r}_k, \psi_k)\Big), \tag{34c}$$

$$\mathbf{h}_{k,l}(\mathbf{x}_k) = \mathbf{r}_k + \tilde{H}_l(\mathbf{r}_k, \psi_k)\mathbf{x}^{\mathbf{f}}_k, \tag{34d}$$

$$R_{k,l} = \mathbf{p}_{k,l} R^{\mathbf{f}}_{k,l} \mathbf{p}^{\mathsf{T}}_{k,l} + R, \tag{34e}$$

$$\mathbf{p}_{k,l} = \mathbf{p}_{k,l}(\mathbf{r}_k), \quad R^{\mathbf{f}}_{k,l} = R^{\mathbf{f}}\Big(\theta^L_{k,l}(\mathbf{r}_k, \psi_k)\Big). \tag{34f}$$

Note that since the measurement noise is already included in (34e), the measurement noise variance in (16b) can be omitted.

5.2 Motion Model

The target state $\bar{\mathbf{x}}_k = [(\mathbf{r}_k)^{\mathsf{T}}, \ \psi_k, \ (\mathbf{x}^*_k)^{\mathsf{T}}]^{\mathsf{T}}$ is described with a linear state space model

$$\bar{\mathbf{x}}_{k+1} = \bar{F}\bar{\mathbf{x}}_k + \bar{\mathbf{w}}_k, \quad \bar{\mathbf{w}}_k \sim \mathcal{N}\left(0, \ \bar{Q}\right), \tag{35a}$$

$$\bar{\mathbf{x}}_0 \sim \mathcal{N}\left(\bar{\boldsymbol{\mu}}_0, \ \bar{P}_0\right), \tag{35b}$$

as commonly performed in target tracking applications.

Together with the dynamical description of the target extent in (18), we construct an augmented description of the dynamics.

$$\mathbf{x}_{k+1} = F\mathbf{x}_k + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}(0, \ Q), \tag{36a}$$

$$\mathbf{x}_0 = \mathcal{N}\left(\boldsymbol{\mu}_0, \ P_0\right), \tag{36b}$$

where

$$\mathbf{x}_k = \begin{bmatrix} \bar{\mathbf{x}}_k \\ \mathbf{x}^{\mathbf{f}}_k \end{bmatrix}, \quad F = \begin{bmatrix} \bar{F} & 0 \\ 0 & F^{\mathbf{f}} \end{bmatrix}, \quad Q = \begin{bmatrix} \bar{Q} & 0 \\ 0 & Q^{\mathbf{f}} \end{bmatrix}, \quad \boldsymbol{\mu}_0 = \begin{bmatrix} \bar{\boldsymbol{\mu}}_0 \\ 0 \end{bmatrix}, \quad P_0 = \begin{bmatrix} \bar{P}_0 & 0 \\ 0 & P^{\mathbf{f}}_0 \end{bmatrix}, \tag{37}$$

where $P^{\mathbf{f}}_0$ is given by (16b), and $F^{\mathbf{f}}$ and $Q^{\mathbf{f}}$ are given by (19). The matrices $\bar{F}$ and $\bar{Q}$ are given later in Section 9.2.

5.3 Discussion

In Huber (2014), further augmenting the state with hyper-parameters is proposed. Such an approach may be suitable for a vanilla problem because the hyper-parameter tuning of GP models primarily requires batch processing of the measurements and multiple iterations. Unfortunately, in a problem such as online target tracking, neither processing the batch data nor performing iterations is feasible. Furthermore, the uncertainties in a target tracking problem arise from many sources, such as target dynamics model mismatch, false alarms, missed detections, and clutter measurements. Moreover, in ETT, the centroid of the target

extent is also unknown and needs to be estimated. Hence, under these uncertainties, we believe that it is more feasible to assume an expected size and length scale for a target rather than online tuning. Later, in the results section, it is shown that both in simulations and real data, targets of different size can be tracked using the same set of hyper-parameters.

6 Inference

An efficient implementation is essential in many target tracking applications. The state space model (29) derived in the previous section allows us to utilize the standard inference techniques to compute the posterior distribution of the target state vector. In this work, we will use an extended Kalman filter (EKF). To recursively update the posterior, we will construct an augmented model in which all measurements $\{\mathbf{y}_{k,l}\}_{l=1}^{n_k}$ within one scan are augmented

$$\mathbf{y}_k = \begin{bmatrix} \mathbf{y}_{k,1}^\mathsf{T}, & \cdots, & \mathbf{y}_{k,n_k}^\mathsf{T} \end{bmatrix}^\mathsf{T}, \tag{38a}$$

$$R_k = \operatorname{diag}[R_{k,1}, \cdots, R_{k,n_k}], \tag{38b}$$

$$\mathbf{h}_k(\mathbf{x}_k) = \begin{bmatrix} \mathbf{h}_{k,1}(\mathbf{x}_k)^\mathsf{T}, & \cdots, & \mathbf{h}_{k,n_k}(\mathbf{x}_k)^\mathsf{T} \end{bmatrix}^\mathsf{T}. \tag{38c}$$

The corresponding state space description is given as follows:

$$\mathbf{x}_{k+1} = F\mathbf{x}_k + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}(0,\ Q), \tag{39a}$$

$$\mathbf{y}_k = \mathbf{h}_k(\mathbf{x}_k) + \mathbf{e}_k, \quad \mathbf{e}_k \sim \mathcal{N}(0,\ R_k), \tag{39b}$$

$$\mathbf{x}_0 \sim \mathcal{N}\left(\boldsymbol{\mu}_0,\ P_0\right). \tag{39c}$$

An estimate $\hat{\mathbf{x}}_k$ can now be computed using a nonlinear filtering technique. In this work we have used an extended Kalman filter, see Appendix A for the required recursions. Note that these recursions require a gradient of the measurement function $\frac{d\mathbf{h}(\mathbf{x}_k)}{d\mathbf{x}_k}$, which can be computed analytically; see Appendix B.

7 Predictive Likelihood and Gating

In the presence of clutter and/or multiple targets, a gating step could optionally be included to reject the measurements that are unlikely to originate from the target. By using the GP model, the predictive likelihood and gates for the predicted measurements can be computed by using the standard Kalman filter equations (Blackman and Popoli, 1999, Chapter 6.3). We start by computing the predictive likelihood, which is available in analytical form

$$\mathbf{y}_{k,l} \sim p(\mathbf{y}_{k,l}|\mathbf{y}_{1:k-1}) = \mathcal{N}\left(\hat{\mathbf{y}}_{k|k-1,l},\ S_{k,l}\right), \tag{40a}$$

where

$$S_{k,l} = H_{k,l} P_{k|k-1} H_{k,l}^{\mathsf{T}} + R_{k,l}, \tag{40b}$$

$$H_{k,l} = \frac{d}{d\mathbf{x}_k} \mathbf{h}_{k,l}(\mathbf{x}_k)|_{\mathbf{x}_k = \hat{\mathbf{x}}_{k|k-1}}, \tag{40c}$$

$$\hat{\mathbf{y}}_{k|k-1,l} = \mathbf{h}_{k,l}(\hat{\mathbf{x}}_{k|k-1}), \tag{40d}$$

where $\mathbf{h}_{k,l}$ and $R_{k,l}$ are defined in (34d) and (34e). The gating is performed by comparing the statistics of the residual vector, $\tilde{\mathbf{y}}_{1:k-1,l} \triangleq \mathbf{y}_{k,l} - \hat{\mathbf{y}}_{1:k-1,l}$, with a threshold ρ,

$$\tilde{\mathbf{y}}_{1:k-1,l}^{\mathsf{T}} S_{k,l} \tilde{\mathbf{y}}_{1:k-1,l} \gtrless \rho. \tag{41}$$

The analytical expression of the target extent provides gates for future measurements that incorporate target shape information.

8 Surface Model using Scaling Parameter

The presented model can also be extended to address measurements that originate from an interior point of a target extent rather than the contour. The measurements originating inside the target boundary are modeled by including a scaling parameter $s_{k,l}$ for each measurement, similar to Sun et al. (2012); Baum et al. (2010b). The measurement equation (34a) can therefore be modified as

$$\mathbf{y}_{k,l} = \mathbf{r}_k + s_{k,l} \tilde{H}_l(\mathbf{r}_k, \psi_k) \mathbf{x}_k^{\mathbf{f}}, \quad \mathbf{e}_{k,l} \sim \mathcal{N}(0,\ R_{k,l}), \tag{42}$$

where $s_{k,l}$ is a random variable on the interval [0,1]. With the assumption that the measurement source is uniformly distributed over the star-convex region, Baum and Hanebeck (2014) have shown that the squared scaling factor is uniformly distributed

$$s_{k,l}^2 \sim U[0,1]. \tag{43}$$

The mean and variance of $s_{k,l}$ can be computed analytically

$$\mu_s = \mathbb{E}(s_{k,l}) = \frac{2}{3}, \tag{44a}$$

$$\sigma_s^2 = \mathrm{Var}(s_{k,l}) = \mathbb{E}(s_{k,l}^2) - (\mathbb{E}(s_{k,l}))^2 = \frac{1}{18}. \tag{44b}$$

To be able to use a Kalman filter for inference, the scaling factor is approximated by a Gaussian random variable that has the same first two moments,

$$s_{k,l} \overset{\text{approx.}}{\sim} \mathcal{N}\left(\mu_s,\ \sigma_s^2\right). \tag{45}$$

The measurement model can now be modified as

$$\begin{aligned} \mathbf{y}_{k,l} &= \underbrace{\mathbf{r}_k + \mu_s \tilde{H}_l(\mathbf{r}_k, \psi_k) \mathbf{x}_k^{\mathbf{f}}}_{=\tilde{\mathbf{h}}_l(\mathbf{x}_k)} + \underbrace{(s_{k,l} - \mu_s) \tilde{H}_l(\mathbf{r}_k, \psi_k) \mathbf{x}_k^{\mathbf{f}} + \mathbf{e}_{k,l}}_{=\tilde{\mathbf{e}}_k} \\ &= \tilde{\mathbf{h}}_l(\mathbf{x}_k) + \tilde{\mathbf{e}}_{k,l}, \qquad \mathbf{e}_{k,l} \sim \mathcal{N}\left(0,\ \tilde{R}_{k,l}\right), \end{aligned} \tag{46a}$$

where the measurement model and the measurement noise covariance are given by

$$\tilde{\mathbf{h}}_l(\mathbf{x}_k) = \mathbf{r}_k + \mu_s \tilde{H}_l(\mathbf{r}_k, \psi_k)\mathbf{x}_k^{\mathbf{f}}, \tag{47a}$$

$$\tilde{R}_{k,l} = R_{k,l} + \sigma_s^2 \tilde{H}_{k,l}\mathbf{x}_k^{\mathbf{f}}(\mathbf{x}_k^{\mathbf{f}})^{\mathsf{T}}\tilde{H}_{k,l}^{\mathsf{T}}, \tag{47b}$$

$$\tilde{H}_{k,l} = \tilde{H}_l(\mathbf{r}_k, \psi_k). \tag{47c}$$

By substituting the measurement function (34d) with (46a), the algorithm can be modified to track extended targets where the measurements are coming from the surface rather than the contour. In addition, note that (34d) will be equal to (46a) for the following choice of parameters, $\mu_s = 1$ and $\sigma_s^2 = 0$.

9 Results

In this section, we evaluate the performance of the proposed method and compare it with relevant extended target models in the literature. The evaluation is performed through both simulations and real data experiments, and the corresponding results are reported in Section 9.2 and Section 9.3, respectively. For the scenarios in which the measurements originate from the target contour, the model described in Section 5 has been used. For the scenarios in which the measurements originate from the target surface, the extended model introduced in Section 8 has been used.

9.1 Alternative Models

Random Matrix Model

The proposed algorithm is compared with a standard random matrix-based extended target tracker (Feldmann et al., 2011), denoted here as RM. RM approximates the posterior distribution of the kinematic state $\bar{\mathbf{x}}_k \in \mathbb{R}^{n_x}$ and the target extent by using the normal inverse Wishart distribution as follows:

$$\begin{aligned} p(\bar{\mathbf{x}}_k, X_k|\mathbf{y}_{1:k}) &\approx p(\bar{\mathbf{x}}_k|\mathbf{y}_{1:k})p(X_k|\mathbf{y}_{1:k}) \\ &= \mathcal{N}(\bar{\mathbf{x}}_k; \bar{\mathbf{m}}_{k|k}, \bar{P}_{k|k}) \times \mathcal{IW}(X_k; \nu_{k|k}, V_{k|k}), \end{aligned} \tag{48}$$

where $\bar{\mathbf{m}}_{k|k}, \bar{P}_{k|k}$ are the mean vector and the covariance matrix of the estimated kinematic state at time index k. X_k is the symmetric positive definite matrix, which represents the elliptical extent, and $\mathcal{IW}(X; \nu, V)$ denotes an inverse Wishart distribution defined over the matrix $X \in \mathbb{S}_{++}$ with scalar degrees of freedom ν and parameter matrix $V \in \mathbb{S}_{++}$; see (Gupta and Nagar, 1999, Definition 3.4.1). The corresponding measurement likelihood for the measurements $\mathbf{y}_{k,l} \in \mathbb{R}^{n_y}$ for a linear model is

$$p(\mathbf{y}_{k,l}|\bar{\mathbf{x}}_k, X_k) = \mathcal{N}(\mathbf{y}_{k,l}; C_k\bar{\mathbf{x}}_k, zX_k + R), \tag{49}$$

where R is the measurement noise covariance matrix and C_k is the $n_x \times n_y$ measurement matrix. The scaling factor z is used in this work to account for the difference between the assumed normal distribution of the measurement sources and the actual uniform distribution over the target region or the target contour.

In the experiments, we employ an exponential forgetting factor in the time update of the sufficient statistics $\nu_{k|k}$ and $V_{k|k}$, which is shown to provide the maximum entropy distribution for the prediction when the transition density for the target extent is unknown but the change in the prediction density is upper bounded by a Kullback Leibler distance (see (Özkan et al., 2013, Theorem 1)). This will help the elliptical model to adapt itself for possible orientation changes in the examples. The time update for the sufficient statistics of the target extent is performed as follows.

$$\nu_{k|k-1} = \lambda \nu_{k-1|k-1}, \tag{50a}$$

$$V_{k|k-1} = \lambda V_{k-1|k-1}, \tag{50b}$$

where λ is the forgetting factor.

Random Hypersurface Model using Fourier Series

The proposed model is also compared with a star-convex model based on a Fourier series expansion of the radial function (Baum and Hanebeck, 2009), which was previously described in Section 2. Here, the model will be denoted as RHF. An implementation that is available online[1] has been used in the simulations.

9.2 Simulations

Several simulations have been performed to evaluate the performance of the proposed model in comparison with the RM and RHF models introduced above. In Section 9.2, moving targets or different shapes are considered, where we compare the proposed model with the RM model, and in Section 9.2, stationary targets of different sizes are considered, where the proposed model is compared with the RHF model. All simulation experiments were performed 100 times with different realizations of the measurement noise and measurement origin at each simulation. The presented numbers are the average of these 100 Monte Carlo runs.

Moving Target

We test the algorithm on moving objects with different shapes. A rectangular (S1), a cross-shaped (S2) and a triangular (S3) object are simulated to generate measurements. A trajectory, which is a combination of linear paths and turns, is generated. The objects first move on a linear path then make a turn and again follow a linear path, always with a velocity of 0.1 m s^{-1}. Note that the constant velocity assumption is no longer valid during the turns, but the algorithm is required to be robust enough to handle such model mismatches, which are encountered in

[1] www.cloudrunner.eu/algorithm/12/random-hypersurface-model/version/2/

most tracking applications. Furthermore, the objects make a rotation during the turns, which has to be tracked by the algorithm. The same set of parameters is used for all different objects to illustrate the robustness and flexibility of the GP model.

A constant velocity model is used for both the position and the orientation of moving targets. However, other dynamics, also non-linear, could be considered as well. By using the well-known time-discrete version of the constant velocity model, the state dynamics equation (35a) will be

$$\bar{F} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \otimes I_3, \quad \bar{Q} = \begin{bmatrix} \frac{T^3}{3} & \frac{T^2}{2} \\ \frac{T^2}{2} & T \end{bmatrix} \otimes \begin{bmatrix} \sigma_q^2 & 0 & 0 \\ 0 & \sigma_q^2 & 0 \\ 0 & 0 & \sigma_{q^\psi}^2 \end{bmatrix},$$

where σ_q^2 and $\sigma_{q^\psi}^2$ are the process noise variances for position and angle, respectively. The process noise standard deviations $\sigma_q = 0.01$ and $\sigma_{q^\psi} = 0.001$ have been used for position and angle, respectively, and $\alpha = 0.0001$ has been used for the target extent dynamics. For the proposed model, the hyper-parameters of the Gaussian process have been set to $\sigma_r = 2$, $\sigma_f = 2$ and $l = \pi/4$; the measurement noise variance has been set to $R = 0.1^2 I_2$; and the sampling time has been set to $T = 1$.

The performance of the target extent estimation is evaluated based on the Intersection-Over-Union (IOU) measure. This is a similarity measure used in, for example, computer vision to compare object shapes for similarity (Alexe et al., 2012). The IOU measure has also been used for evaluation in an extended target tracking context (Granström et al., 2011). If the true target covers region R_0 and the estimated target covers region $\hat{R}$, then the IOU measure is defined as the ratio of the areas for the intersection and the union of these two regions

$$\text{IOU}(\hat{R}, R_0) = \frac{\text{area}(\hat{R} \cap R_0)}{\text{area}(\hat{R} \cup R_0)}. \tag{51}$$

By construction, $\text{IOU}(\hat{R}, R_0) \in [0, 1]$, where the value 1 corresponds to a perfect match between these two regions and the value 0 corresponds to the worst possible match, i.e., the regions are not even overlapping.

The performance of the target position is evaluated using the root-mean-square error

$$\text{RMSE}(\hat{\mathbf{r}}, \mathbf{x}^0) = \sqrt{\frac{1}{N} \sum_{k=1}^{N} (\hat{\mathbf{r}}_k - \mathbf{r}_{0,k})^2}, \tag{52}$$

where $\mathbf{r}_{0,k}$ is the true position at time instance k.

For the RM model, we use the scaling factor $z = 1/4$ in scenarios where the measurements originate from the target surface and $z = 1/2$ if they originate from the target contour to obtain an unbiased estimate of the target extent; see Baum et al. (2010a) for further details.

Measurement origin	Target contour			Target surface		
Target shape	S1	S2	S3	S1	S2	S3
Example result (Fig.)	4a	4c	4e	4b	4d	4f
Proposed model	0.13	0.12	0.71	0.76	0.88	1.40
Random matrix (RM)	0.33	0.31	0.76	0.25	0.28	0.28

Table 1: *Root-mean-square error [*m*] of the target position in the simulated scenario using different shapes and target extent models. The numbers are averaged over 100 MC runs.*

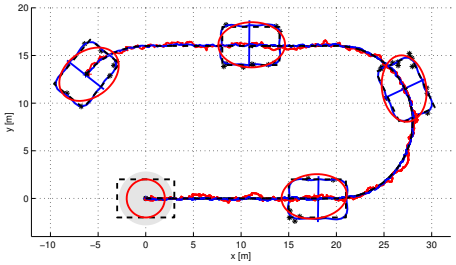
Measurement origin	Target contour			Target surface		
Target shape	S1	S2	S3	S1	S2	S3
Example result (Fig.)	4a	4c	4e	4b	4d	4f
Proposed model	0.93	0.86	0.86	0.84	0.73	0.64
Random matrix (RM)	0.79	0.57	0.62	0.80	0.61	0.63

Table 2: *The mean value of Intersection-Over-Union (IOU) between the true and the estimated target regions in the simulated scenario using different shapes and target extent models. The numbers are averaged over 100 MC runs.*

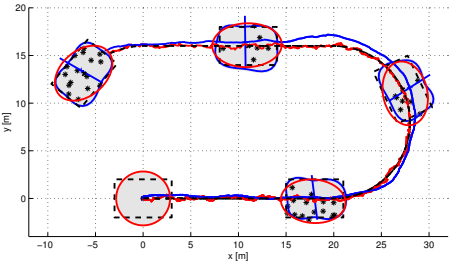
The number of measurements at each scan is Poisson distributed. An average of 10 measurements are generated in each scan. In the first set of simulations, the measurements are uniformly distributed over the target contour, and in the second set, the measurements are uniformly distributed over the target surface. The RMSE values for the position estimates are presented in Table 1, and the IOU values of the extent estimates are presented in Table 2. In Figure 4, an example of one MC run for each of the setups is presented. The measurements from a few scans are also illustrated.

As shown in Table 2, the proposed model is better at estimating the target extent than the RM model. This result is expected because the RM model is restricted to elliptical shapes, whereas the proposed model can handle any star-convex shape. If the measurements originate from the target surface, the improvement in IOU is rather small. This is particularly the case for shapes S1 and S3, which can be fairly well approximated with an ellipse. However, for shape S2, the improvement is more significant.

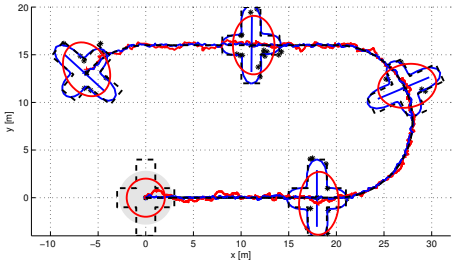
If the measurements originate from the target contour, the proposed model outperforms the RM model in estimating both the target extent and target position. The structure of the measurements in this setting can be utilized by the proposed model, which is not the case for the RM model. However, if the measurements originate from the target surface (see right part of Table 2), the RM model performs better in position estimation. This can also be observed in Figure 4b, 4d and 4f, where the position estimates of the proposed method do not match the true trajectory during the turns. The main reason for the differences in the performances is that there is less structured information that can be exploited



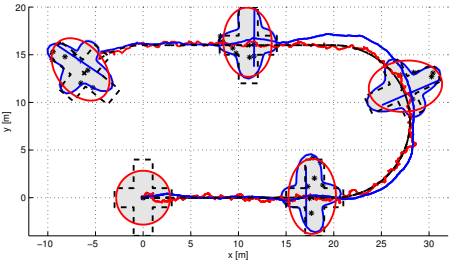
(a) *Target shape: S1. Measurement origin: Target contour*



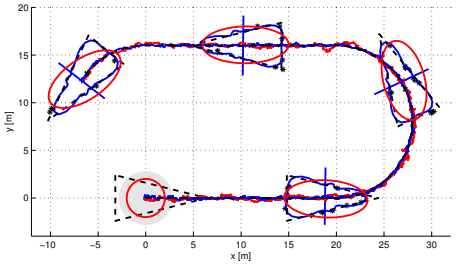
(b) *Target shape: S1. Measurement origin: Target surface*



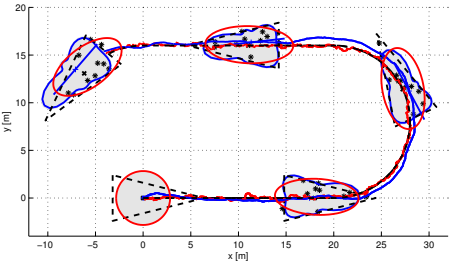
(c) *Target shape: S2. Measurement origin: Target contour*



(d) *Target shape: S2. Measurement origin: Target surface*



(e) *Target shape: S3. Measurement origin: Target contour*



(f) *Target shape: S3. Measurement origin: Target surface*

Figure 4: *Example results for tracking different moving targets where the measurements originate from the target contour in the left column ((a),(c),(e)) and from the target surface in the right column ((b),(d),(f)). True target (dashed black line) is compared with the proposed method (blue contour and with confidence region in gray and blue trajectory) and the RM model (red ellipse and trajectory). The blue line co-centered with the target represents the estimated orientation, and the measurements are depicted as black stars. Five snapshots out of the 500 scans are presented.*

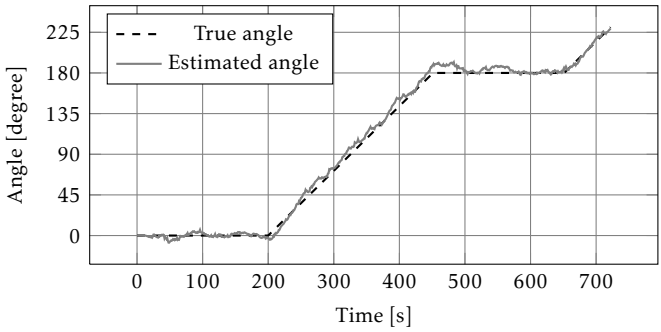


Figure 5: *Target orientation estimate for the scenario illustrated in Figure 4f.*

by the GP model in the surface measurements than in the contour measurements. The RM model uses the mean of the measurements when updating the target position estimates. In scenarios where the measurement mean contains information about the target centroid, the RM method will have an advantage. Another factor is that the target center is not uniquely defined for a star-convex shape, and this results in a bias in the centroid estimates of the GP model. Therefore, the IOU is a more relevant performance measure because it is invariant to the target center. However, if the target is symmetric, the covariance function presented in Section 4.3 can be used.

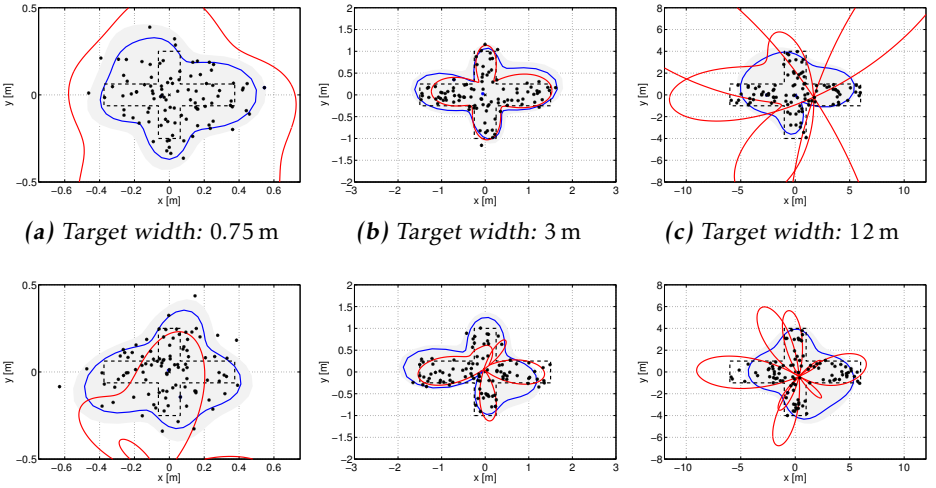
In Figure 5, the orientation estimate is shown for the target in Figure 4f. The proposed model is able to estimate not only the target position and extent, but also the target orientation accurately. Such an estimate is not provided by the RM model, and therefore, the proposed model provides a more detailed description of the target trajectory than the RM model.

Stationary Target

In total, 100 measurements from the surface of a stationary cross-shaped target have been generated. The target shape is the same as the one implemented in the code for the RHF model; see Section 9.1. To evaluate the ability of the proposed method to estimate targets of different sizes, three different sizes of the target shape have been considered. The tuning parameters for the RHF model, as given in the code, have been tailored for medium-sized targets. For the proposed model, the same hyper-parameters of the Gaussian process have been chosen as in the previous simulation. In Figure 6a-6c, the results for the three target sizes are presented.

Table 3 presents the IOU averaged over 100 MC runs. The results indicate that the RHF model performs slightly better for the medium-sized target presented in Figure 6b. However, the low initial covariance of the Fourier coefficients makes the RHF model unable to estimate the shapes of other target sizes, as shown in Figure 6a and 6c.

In Figure 6d-6f, a larger initial covariance has been used to decrease the im-



(a) Target width: 0.75 m *(b) Target width:* 3 m *(c) Target width:* 12 m

(d) Target width: 0.75 m *(e) Target width:* 3 m *(f) Target width:* 12 m

Figure 6: *Example results for estimating the extent of stationary targets of different sizes. True target (dashed black line) is compared with the proposed method (blue line and with confidence region in gray) and the RHF model (red line). The measurements are indicated with black stars. In Figures (a)-(c), the same initial standard deviation for the Fourier coefficients in the RHF model has been chosen, as provided in the code available online, whereas in Figures (d)-(f), a four-fold greater initial standard deviation has been chosen.*

pact of the prior on the performance for the RHF model, denoted here as RHFb. However, no stable estimates are achieved, as shown in Figure 6d-6f. Consequently, the authors were not able to find a set of tuning parameters for the RHF model that worked well for data from all these three targets, whereas the proposed model estimates reasonable target shapes in all three of these cases.

In fact, the estimated radius for the RHF model is occasionally negative. The same behavior has also been observed and reported in Sun et al. (2012), where inequality constraints on these parameters were suggested. Although this can also theoretically occur for the proposed model, the simulations indicate that this occurs considerably less frequently for the proposed model than for the RHF model.

Computation Time

Because the proposed solution is implemented with a standard extended Kalman filter, the computational demand is fairly low. The state dimension for 50 basis points is $\dim(\mathbf{x}_k) = \dim(\mathbf{r}_k) + \dim(\psi_k) + \dim(\mathbf{x}_k^*) + \dim(\mathbf{x}^\mathbf{f}) = 2 + 1 + 3 + 50 = 56$, and the analytical expressions for the derivatives $\frac{\partial \mathbf{h}(\mathbf{x})}{\partial \mathbf{x}}$ are available. The com-

Target size (width)	0.75 m	3 m	12 m
Example result	Figure 6a/6d	Figure 6b/6e	Figure 6c/6f
Proposed model	0.39	0.66	0.52
Fourier coef. model (RHF)	0.09	0.68	0.25
Fourier coef. model (RHFb)	0.15	0.54	0.36

Table 3: *The Intersection-Over-Union (IOU) between the true and the estimated target regions in the simulated scenario using different targets sizes and extended target models. The numbers are averaged over 100 MC runs.*

putational demand does not increase over time because the algorithm is fully recursive. All simulations are run in Matlab(R) R2013b on a standard laptop with an Intel(R) Core(TM) i5-2520M 2.50 GHz platform with 8 GB of RAM. One measurement update and one time update require 2.3 ms for the proposed method, 0.5 ms for RM, and 3.5 ms for the RHF model on average.

9.3 Real Data Experiments

In this section, we aim to illustrate the capabilities of the algorithm on real data. We will use a data set presented in Granström and Orguner (2012); Granström et al. (2012), where only a single type of object is considered. In this data set, a laser range sensor is used to collect data from multiple objects of different sizes and shapes. The sensor measured the range every 0.5° over a 180° surveillance region. The sensor is assumed to be located at the origin of a hemisphere, and every detection closer than 15 m is converted to Cartesian positions. The sampling time for a full 180° scan is 0.2 s. From the data set, three scenarios have been extracted, and the results are presented in Sections 9.3 and 9.3. In all scenarios, the hyper-parameters of the Gaussian process have been set to $\sigma_r = 1$, $\sigma_f = 1$ and $l = \pi/8$, and the measurement noise variance has been set to $R = 0.1I_2$. The process noise standard deviations $\sigma_q = 2.5$ and $\sigma_{q^\psi} = 0.5$ have been used for position and angle, respectively, as has $\alpha = 0.001$ for the target extent dynamics.

Miscellaneous Object Tracking

In Scenarios 1 and 2, we consider multiple maneuvering objects of different shapes (cars, bicycles and pedestrians), which enter and exit a surveillance region. The objects of interest are significantly different from each other. A snapshot of measurements originating from the objects are shown in Figure 7. Note that the problem addressed in this work is tracking a single extended target given the associated measurements. Here, we combined the proposed filter with a simple multi-target tracking approach for illustrating the output of the algorithm for different targets within the same scene without requiring any extra target-dependent parameter tuning.

We use a simple target tracking algorithm, where every measurement is associated with the nearest target if the measurement falls into the gate of the target. We used the gating strategy described in Section 7, where the gate threshold is

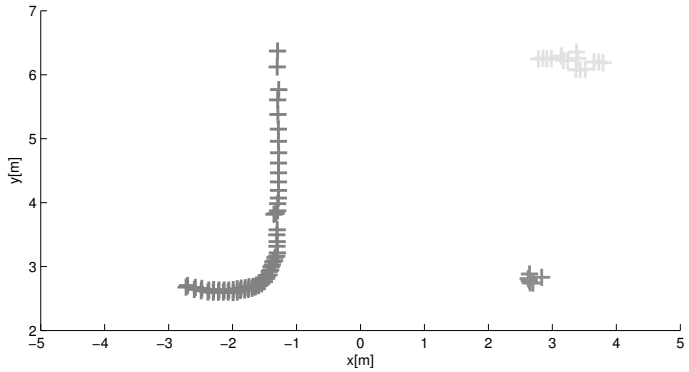
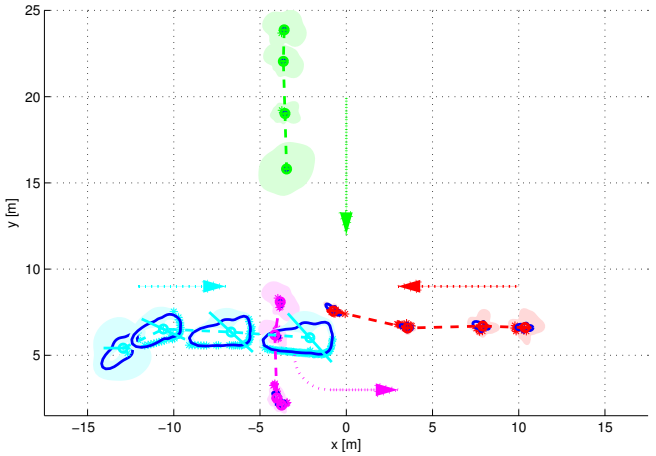


Figure 7: *Laser range sensor measurements corresponding to a car (on the left), a bicycle (top-right corner) and a pedestrian.*

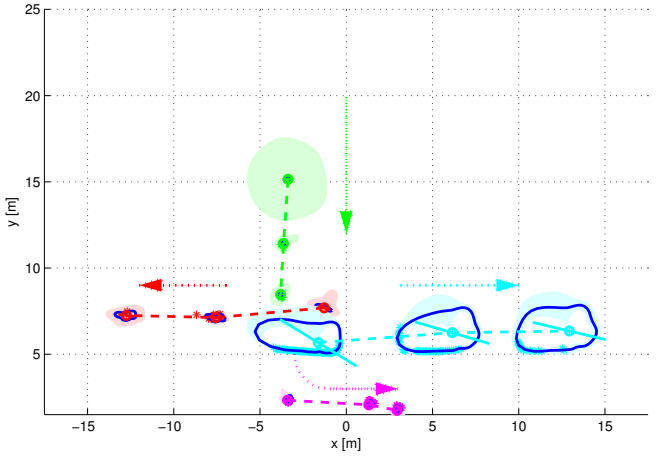
chosen to ensure that 99 percent of the target-originated measurements fall into the target's gate. The remaining measurements (which are not associated with any of the existing tracks) are simply clustered according to their mutual distances, and a new track is generated from each cluster.

In Figures 8 and 9, we present two different scenarios. In these figures, multiple snapshots of the measurements and the output of the GP tracker are superimposed together. The centroid of the object, its orientation, the estimated target extent and the confidence region of the extent are plotted as the output of the tracker.

In Scenario 1, four targets move in the scene. Multiple snapshots from the scenario are split into two sub-figures and plotted in Figure 8. In this scenario, one car is moving from left to right, one bicycle is moving in the opposite direction, one pedestrian (at the top) is walking in the top to bottom direction, and another pedestrian makes an L-shaped move close to the origin of the scene. The snapshots corresponding to frames {1, 6, 16, 27} are plotted in Figure 8a, and those corresponding to frames {28, 42, 52} are plotted in Figure 8b. The snapshots are chosen for the sake of clearer illustration, where the initial and final stages of the tracks are shown and the overlaps between the snapshots are kept to a minimum. In the scenario, the car and the bicycle pass each other while moving in opposite directions. During the transition, the bicycle is fully occluded by the car in multiple consecutive scans, as it blocks the line of sight of the range sensor, thereby preventing any possible detection. The pedestrian moving toward the bottom of the figure is also occluded by the car between scans 20 and 30. The prior for the target extent is the same for all objects at the initialization, which is chosen to be a circle with a radius of 2 m. Note that at the beginning of the scenario, the uncertainty in the target extent is large for all targets, and it decreases in time as more measurements are collected from the targets. Furthermore, one can also observe that the uncertainty region of the car's extent decreases around the observable section of the car where the reflections occur, and the uncertainty for the unobservable section is maintained due to a lack of observations. During the occlu-



(a) *Scenario 1: Part I*



(b) *Scenario 1: Part II*

Figure 8: *Scenario 1: Four targets moving in the scene. One car, moving from left to right, is shown in blue; one bicycle, moving in the opposite direction, is shown in red; one pedestrian (at the top), walking toward the bottom of the figure, is shown in green; and another pedestrian, who makes an L-shaped move at the bottom of the figure, is shown in magenta. The arrows indicate the direction of the targets' movements. For each target, the estimated contour is plotted with a solid line, the confidence region is plotted as a shaded area, and the measurements are shown as star-shaped markers. Figure ((a)) and ((b)) show the first and the second half of the scenario, respectively.*

sions, the uncertainty regions of the occluded objects increase because the filters are not updated by measurements; only the time update is performed. Once these objects can be detected again, the uncertainty decreases. The GP tracker is successful in tracking all different objects with their extent and orientation. Note that all filters are using the same prior and parameters. No additional tuning of hyper-parameters is required.

In Scenario 2, one car is moving from left to right and two pedestrians walk in the top to bottom direction, as shown in Figure 9. The pedestrians leave the scene during the scenario. Note that at the end of the scenario, only a few measurements are detected from only one edge of the car, and the algorithm successfully associates those detections with the back of the car and updates its extent and centroid accordingly. A typical approach that would discard the extent structure and attempt to represent the target with an approximate shape would fail to correctly estimate the centroid, which is presented explicitly in the next scenario.

Partially Observable Objects

In Scenario 3, we illustrate the performance of the model in a maneuvering object scenario; see Figure 10. We compare the output of the proposed method with that from the RM and the RHF models. The hyper-parameters of the GP are kept the same as in the previous subsection. The forgetting factor for the RM model, λ, is selected to be 0.99. The standard deviation of the Fourier coefficients is chosen as before for the RHF model. Manual tuning is attempted for this model, but no significant performance gain is achieved. This is because the errors arise from the fact that the model does not account for the rotation of the objects. In the scenario, one vehicle makes a right turn starting from the top of the Figure 10 while moving in the top to bottom direction. Throughout the scenario, the full target extent is never observable; however, the GP model is able to estimate the observable extent and predicts a higher uncertainty around the unobserved section of the target. The RM model tracks the centroid of the measurements rather than the centroid of the target, as expected. The performances of the RM and RHF models are limited because neither of these algorithms is capable of tracking the orientation of a target. The orientation of the partially observed target, before and after the turn, is successfully tracked by the GP model.

Finally, in Figure 11, the performance of the proposed method is evaluated on Scenario 3 using the symmetric assumption induced by the covariance function proposed in (26). This covariance function assumes that the radial distance has a period of π, $f(\theta) = f(\theta + \pi)$. This allows us to obtain a low uncertainty of sections of the target contour that has not yet been seen because of the symmetry that we impose. If such assumptions are valid, even better overall tracking performance could be expected. An investigation of such extensions should be addressed in future work.

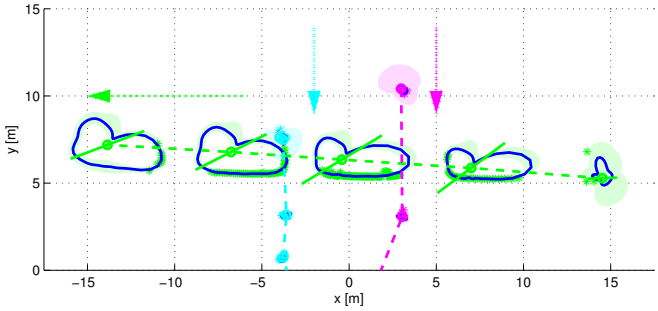


Figure 9: *Scenario 2: Three targets moving in the scene. One car, moving from right to left, is shown in green; one pedestrian (at the top) walking toward the bottom of the figure is shown in blue; and another pedestrian is shown in magenta.*

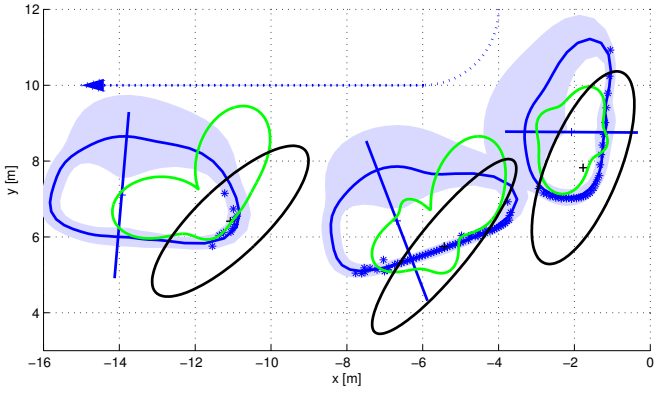


Figure 10: *Scenario 3: A vehicle making a right turn starting from the top of the figure. Outputs of the proposed (blue), the RM (black) and the RHF (green) models are plotted together.*

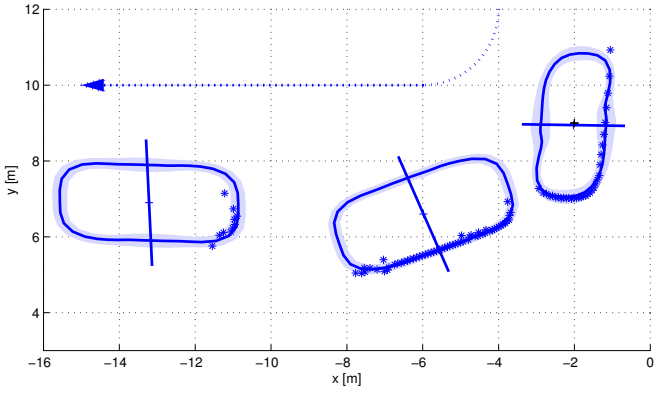


Figure 11: *Scenario 3 (symmetric covariance function): Same data as in Figure 10, where the symmetric covariance function* (26) *has been used.*

10 Conclusion

In this work, we propose a new approach for extended target tracking. The proposed method uses GPs to model the unknown target extent while simultaneously estimating the kinematic state of the target. We provide an efficient algorithm in which the filter updates are fully recursive and do not suffer from an increase in dimension with each available measurement, unlike the standard GP formulation. The performance and capabilities of the algorithm are demonstrated through simulations and real data experiments. The algorithm provides an analytical representation of the unknown target extent, which can be used for high-accuracy gating and object classification in future works.

Appendix

A Extended Kalman Filter Update

After initializing the estimate $\hat{\mathbf{x}}_{0|-1} = \boldsymbol{\mu}_0$ and $P_{0|-1} = P_0$, measurement and time update are applied sequentially for each scan.

Measurement Update

The standard EKF measurement update equations for (39) are

$$\hat{\mathbf{y}}_{k|k-1} = \mathbf{h}_k(\hat{\mathbf{x}}_{k|k-1}), \tag{53a}$$

$$H_k = \frac{d}{d\mathbf{x}_k}\mathbf{h}_k(\mathbf{x}_k)|_{\mathbf{x}_k=\hat{\mathbf{x}}_{k|k-1}}, \tag{53b}$$

$$S_k = H_k P_{k|k-1} H_k^\mathsf{T} + R_k, \tag{53c}$$

$$K_k = P_{k|k-1} H_k^\mathsf{T} S_k^{-1}, \tag{53d}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + K_k(\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1}), \tag{53e}$$

$$P_{k|k} = P_{k|k-1} + K_k H_k P_{k|k-1}. \tag{53f}$$

These recursions require a gradient of the measurement function $\frac{d\mathbf{h}(\mathbf{x}_k)}{d\mathbf{x}_k}$, which can be computed analytically; see Appendix B. Note that we compute the full derivatives of (34) with respect to $\mathbf{r}_k$ and ψ_k, requiring analytical expressions of $\frac{d\theta^G(\mathbf{r})}{d\mathbf{r}}$ and $\frac{dk(\theta,\theta')}{d\theta}$ to be combined with the chain rule. This results in a precise and correct state update, enabling estimation of the target orientation, which otherwise would not have been possible.

Time Update

Furthermore, in accordance with the EKF recursions, the time update equations are as follows:

$$\hat{\mathbf{x}}_{k+1|k} = F\hat{\mathbf{x}}_{k|k-1}, \tag{54a}$$
$$P_{k+1|k} = FP_{k|k-1}F^{\mathsf{T}} + Q. \tag{54b}$$

B Partial Derivatives

The derivative in (53b) can be divided into the following submatrices

$$H_k = \frac{d\mathbf{h}_k(\mathbf{x}_k)}{d\mathbf{x}_k} = \frac{d}{d\mathbf{x}_k}\Big[\mathbf{h}_{k,1}(\mathbf{x}_k)^{\mathsf{T}}, \ \dots \ , \mathbf{h}_{k,n_x}(\mathbf{x}_k)^{\mathsf{T}}\Big]^{\mathsf{T}}, \tag{55a}$$
$$\frac{d}{d\mathbf{x}_k}\mathbf{h}_{k,l}(\mathbf{x}_k) = \begin{bmatrix} \frac{d\mathbf{h}_{k,l}(\mathbf{x}_k)}{d\mathbf{r}_k} & \frac{d\mathbf{h}_{k,l}(\mathbf{x}_k)}{d\psi_k} & \frac{d\mathbf{h}_{k,l}(\mathbf{x}_k)}{d\mathbf{x}_k^{\mathbf{f}}} \end{bmatrix}, \tag{55b}$$

where each of them is given by

$$\begin{aligned}\frac{d\mathbf{h}_{k,l}(\mathbf{x}_k)}{d\mathbf{r}_k} &= I + \frac{\partial \mathbf{p}_{k,l}(\mathbf{w})}{\partial \mathbf{w}}\Big|_{\mathbf{w}=\mathbf{r}_k} H^{\mathbf{f}}\Big(\theta_{k,l}^L(\mathbf{r}_k, \psi_k)\Big)\mathbf{x}_k^{\mathbf{f}} \\ &+ \mathbf{p}_{k,l}(\mathbf{r}_k)\frac{\partial H^{\mathbf{f}}(u)}{\partial u}\Big|_{u=\theta_{k,l}^L(\mathbf{r}_k,\psi_k)} \frac{\partial \theta_{k,l}^G(\mathbf{w})}{\partial \mathbf{w}}\Big|_{\mathbf{w}=\mathbf{r}_k}\mathbf{x}_k^{\mathbf{f}},\end{aligned} \tag{56a}$$
$$\frac{d\mathbf{h}_{k,l}(\mathbf{x}_k)}{d\psi_k} = -\mathbf{p}_{k,l}(\mathbf{r}_k)\frac{\partial H^{\mathbf{f}}(u)}{\partial u}\Big|_{u=\theta_{k,l}^L(\mathbf{r}_k,\psi_k)}\mathbf{x}_k^{\mathbf{f}}, \tag{56b}$$
$$\frac{d\mathbf{h}_{k,l}(\mathbf{x}_k)}{d\mathbf{x}_k^{\mathbf{f}}} = \mathbf{p}_{k,l}(\mathbf{r}_k)H^{\mathbf{f}}\Big(\theta_{k,l}^L(\mathbf{r}_k, \psi_k)\Big), \tag{56c}$$

where

$$\frac{\partial \theta_{k,l}^G(\mathbf{w})}{\partial \mathbf{w}} = \frac{1}{\|\mathbf{y}_{k,l}-\mathbf{w}\|^2}\Big[y_{k,l}^{\mathsf{y}}-w^{\mathsf{y}}, \ -(y_{k,l}^{\mathsf{x}}-w^{\mathsf{x}})\Big], \tag{57a}$$
$$\frac{\partial \mathbf{p}_{k,l}(\mathbf{w})}{\partial \mathbf{w}} = \frac{(\mathbf{y}_{k,l}-\mathbf{w})(\mathbf{y}_{k,l}-\mathbf{w})^{\mathsf{T}}}{\|\mathbf{y}_{k,l}-\mathbf{w}\|^3} - \frac{1}{\|\mathbf{y}_{k,l}-\mathbf{w}\|}I, \tag{57b}$$
$$\frac{\partial H^{\mathbf{f}}(u)}{\partial u} = \frac{\partial K(u,\mathbf{u}^{\mathbf{f}})}{\partial u}[K(\mathbf{u}^{\mathbf{f}},\mathbf{u}^{\mathbf{f}})]^{-1}, \tag{57c}$$
$$\frac{\partial K(u,\mathbf{u}^{\mathbf{f}})}{\partial u} = \frac{\partial}{\partial u}\begin{bmatrix} k_{\mathrm{tot}}(u,u_1^{\mathbf{f}}) & \dots & k_{\mathrm{tot}}(u,u_{N^{\mathbf{f}}}^{\mathbf{f}}) \end{bmatrix}, \tag{57d}$$
$$\frac{\partial k_{\mathrm{tot}}(u,u_i^{\mathbf{f}})}{\partial u} = -\frac{1}{l^2}\sin(u-u_i^{\mathbf{f}})k(u,u_i^{\mathbf{f}}). \tag{57e}$$

Bibliography

B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of image windows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2189–2202, November 2012.

Y. Bar-Shalom and T. Fortmann. *Tracking and data association*. Mathematics in Science and Engineering, Academic Press Professional, CA, US, 1987.

M. Baum. *Simultaneous Tracking and Shape Estimation of Extended Objects*. PhD thesis, Karlsruhe Institute of Technology, 2013.

M. Baum and U. D. Hanebeck. Random hypersurface models for extended object tracking. In *Proceedings of 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 14th Conference on Information Fusion (FUSION)*, pages 338–345, Chicago, US, July 2011.

M. Baum and U. D. Hanebeck. Extended object tracking with random hypersurface models. *IEEE Transactions on Aerospace and Electronic Systems*, 50(1): 149–159, January 2014.

M. Baum, M. Feldmann, D. Fränken, U. D. Hanebeck, and 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 13th Conference on Information Fusion (FUSION)*, Edinburgh, UK, July 2010b.

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

A. Blake, R. Curwen, and A. Zisserma. A framework for spatio-temporal control in the tracking of visual contours. *International Journal of Computer Vision*, 11(2):127–145, 1993.

A. Blake, M. Isard, and D. Reynard. Learning to track the visual motion of contours. *Artificial Intelligence*, 78(1):179–212, 1995.

M. Feldmann, D. Franken, and W. Koch. Tracking of extended objects and group targets using random matrices. *IEEE Transactions on Signal Processing*, 59(4): 1409–1420, April 2011. doi: 10.1109/TSP.2010.2101064.

K. Granström and U. Orguner. A PHD filter for tracking multiple extended targets using random matrices. *IEEE Transactions on Signal Processing*, 60(11): 5657–5671, November 2012.

K. Granström and U. Orguner. New prediction for extended targets with random matrices. *IEEE Transactions on Aerospace and Electronic Systems*, 50(2):1577–1589, April 2014.

K. Granström, C. Lundquist, and U. Orguner. Tracking rectangular and elliptical extended targets using laser measurements. In *Proceedings of the 14th International Conference on Information Fusion (FUSION)*, pages 592–599, Chicago, US, 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*, 48(4):3268–3286, October 2012.

A. Gupta and D. Nagar. *Matrix Variate Distributions*. Monographs and Surveys in Pure and Applied Mathematics. Taylor & Francis, 1999.

J. Hartikainen and S. Särkkä. Kalman filtering and smoothing solutions to temporal Gaussian process regression models. In *Proceedings of the International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 379–384, Kittila, Finland, August 2010.

M. F. Huber. Recursive Gaussian process regression. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3362–3366, Vancouver, Canada, May 2013.

M. F. Huber. Recursive Gaussian process: On-line regression and learning. *Pattern Recognition Letters*, 45:85–91, August 2014.

M. Isard and A. Blake. Condensation - Conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

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. Lan and X. R. Li. Tracking of extended object or target group using random matrix — Part I: New model and approach. In *Proceedings of the 15th International Conference on Information Fusion (FUSION)*, pages 2177–2184, Singapore, July 2012a.

J. Lan and X. R. Li. Tracking of extended object or target group using random matrix — Part II: Irregular object. In *Proceedings of the 15th International Conference on Information Fusion (FUSION)*, pages 2185–2192, Singapore, July 2012b.

P. Li, T. Zhang, and A. E. C. Pece. Visual contour tracking based on particle filters. *Image and Vision Computing*, 21(1):111–123, January 2003.

P. Li, T. Zhang, and B. Ma. Unscented Kalman filter for visual curve tracking. *Image and Vision Computing*, 22(2):157–164, February 2004.

D. J. C. MacKay. Introduction to Gaussian processes. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, pages 133–165. Springer Verlag, 1998.

U. Orguner. A variational measurement update for extended target tracking with random matrices. *IEEE Transactions on Signal Processing*, 60(7):3827–3834, July 2012.

M. Osborne. *Bayesian Gaussian processes for sequential prediction, optimisation and quadrature*. PhD thesis, University of Oxford, UK, 2010.

E. Özkan, V. Smidl, S. Saha, C. Lundquist, and F. Gustafsson. Marginalized adaptive particle filtering for nonlinear models with unknown time-varying noise parameters. *Automatica*, 49(6):1566–1575, June 2013.

C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.

L. Sun, J. Lan, and X. R. Li. Extended target tracking using star-convex model with nonlinear inequality constraints. In *Proceedings of the 31st Chinese Control Conference (CCC)*, pages 3869–3874, Hefei, China, July 2012. IEEE.

N. Wahlström and E. Özkan. Extended target tracking using Gaussian processes. *IEEE Transactions on Signal Processing*, 63(16):4165–4178, 2015.

Paper E

Learning Deep Dynamical Models From Image Pixels

Authors: Niklas Wahlström, Thomas B. Schön, Marc P. Desienroth

Edited version of the paper:

N. Wahlström, T. B. Schön, and M. P. Deisenroth. Learning deep dynamical models from image pixels. In *Proceedings of the 17th IFAC Symposium on System Identification (SYSID)*, Bejing, China, October 2015a.

Learning Deep Dynamical Models From Image Pixels

Niklas Wahlström*, Thomas B. Schön†, Marc P. Desienroth‡

*Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
nikwa@isy.liu.se

†Dept. of Information Technology,
Uppsala University, Sweden
thomas.schon@it.uu.se

‡Dept. of Computing, Imperial College
London, United Kingdom
m.deisenroth@imperial.ac.uk

Abstract

Modeling dynamical systems is important in many disciplines, such as control, robotics, or neurotechnology. Commonly the state of these systems is not directly observed, but only available through noisy and potentially high-dimensional observations. In these cases, system identification, i.e., finding the measurement mapping and the transition mapping (system dynamics) in latent space can be challenging. For linear system dynamics and measurement mappings efficient solutions for system identification are available. However, in practical applications, the linearity assumption does not hold, requiring nonlinear system identification techniques. If additionally the observations are high-dimensional (e.g., images), nonlinear system identification is inherently hard. To address the problem of nonlinear system identification from high-dimensional observations, we combine recent advances in deep learning and system identification. In particular, we jointly learn a low-dimensional embedding of the observation by means of deep auto-encoders and a predictive transition model in this low-dimensional space. We demonstrate that our model enables learning good predictive models of dynamical systems from pixel information only.

High-dimensional time series include video streams, electroencephalography (EEG) and sensor network data. Dynamical models describing such data are desired for forecasting (prediction) and controller design, both of which play an important role, e.g., in autonomous systems, machine translation, robotics and surveillance applications. A key challenge is system identification, i.e., finding a mathematical model of the dynamical system based on the information provided by measurements from the underlying system. In the context of state-space models this includes finding two functional relationships between (i) the states at different time steps (prediction/transition model) and (ii) states and corresponding

measurements (observation/measurement model). In the linear case, this problem is well studied, and many standard techniques exist, e.g., subspace methods (Van Overschee and De Moor, 1996), expectation maximization (Shumway and Stoffer, 1982; Ghahramani, 1998; Gibson and Ninness, 2005) and prediction-error methods (Ljung, 1999). However, in realistic and practical scenarios we require nonlinear system identification techniques.

Learning nonlinear dynamical models is an inherently difficult problem, and it has been one of the most active areas in system identification for the last decades (Ljung, 2010; Sjöberg et al., 1995). In recent years, sequential Monte Carlo (SMC) methods have received attention for identifying nonlinear state-space models (Schön et al., 2011), see also the recent survey (Kantas et al., 2014). While methods based on SMC are powerful, they are also computationally expensive. Learning nonlinear dynamical models from very high-dimensional sensor data is even more challenging. First, finding (nonlinear) functional relationships in very high dimensions is hard (un-identifiability, local optima, overfitting, etc.); second, the amount of data required to find a good function approximator is enormous. Fortunately, high-dimensional data often possesses an intrinsic lower dimensionality. We will exploit this property for system identification by finding a low-dimensional representation of high-dimensional data and learning predictive models in this low-dimensional space. For this purpose, we need an automatic procedure to find compact low-dimensional representations/features. Doretto et al. (2003) implemented a subspace identification routine to model dynamical textures from high-dimensional pixel data. Whereas that method relies on a linear dimensionality reduction method, we will consider nonlinear mappings from the high-dimensional data to the low-dimensional features.

The state of the art in learning parsimonious representations of high-dimensional data is currently defined by deep learning architectures, such as deep neural networks (Hinton and Salakhutdinov, 2006), stacked/deep auto-encoders (Vincent et al., 2008) and convolutional neural networks (LeCun et al., 1998), all of which have been successfully applied to image, text, speech and audio data in commercial products, e.g., by Google, Amazon and Facebook. Typically, these feature learning methods are applied to static data sets, e.g., for image classification. The auto-encoder gives explicit expressions of two generative mappings: (i) an encoder $\mathbf{g}^{-1}$ mapping the high-dimensional data to the features, and (ii) a decoder $\mathbf{g}$ mapping the features to high-dimensional reconstructions.

In this paper, we combine feature/representation learning and dynamical systems modeling to obtain good predictive models for high-dimensional time series, e.g., videos. In particular, we use deep auto-encoder neural networks for automatically finding a compact low-dimensional representation of an image. In this low-dimensional feature space, we use a neural network for modeling the nonlinear system dynamics. An simplified illustration of our approach is shown in Figure 1. An encoder $\mathbf{g}^{-1}$ maps an image $\mathbf{y}_k$ at time step k to a low-dimensional feature $\mathbf{z}_k$. In this feature space, a prediction model $\mathbf{f}$ maps the feature forward in time to $\mathbf{z}_{k+1}$. The decoder $\mathbf{g}$ can generate a predicted image $\mathbf{y}_{k+1}$ at the next time step. This framework needs access to both the encoder $\mathbf{g}^{-1}$ and the decoder $\mathbf{g}$, which motivates our use of the auto-encoder as dimensionality reduction tech-

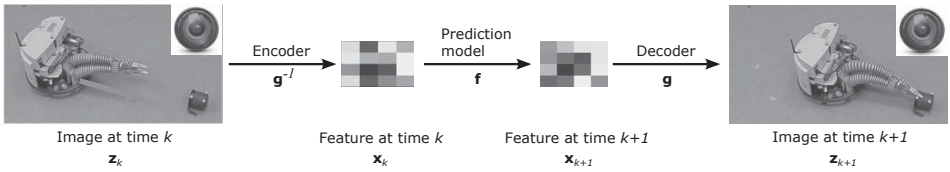


Figure 1: *Combination of deep learning architectures for feature learning and prediction models in feature space. A camera observes a robot approaching an object. A good low-dimensional feature representation of an image is important for learning a predictive model if the camera is the only sensor available.*

nique. Crucially, the embedding and the predictive model in feature space are learned *jointly*.

The contributions of this paper are (i) a model for learning a low-dimensional dynamical representation of high-dimensional data, which can be used for long-term predictions; (ii) experimental evidence demonstrating that joint learning of the parameters in the latent embedding and in the predictive model in latent space can increase the performance compared to separate training.

1 Model

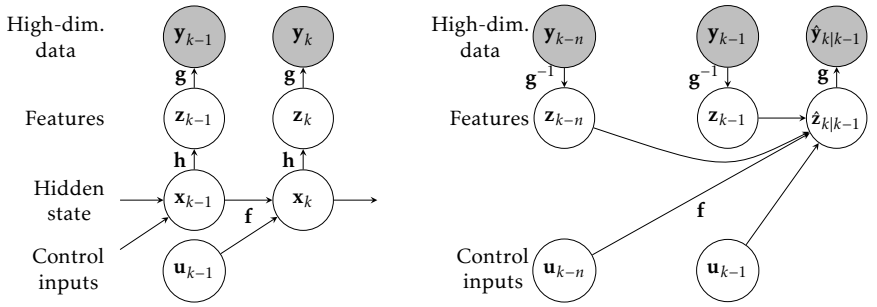
We consider a dynamical system where control inputs are denoted by u and observations are denoted by $\mathbf{y}$. In this paper, the observations are pixel information from images. We assume that a low-dimensional latent variable $\mathbf{z}$ exists that compactly represents the relevant properties of $\mathbf{y}$. Since we consider dynamical systems, a low-dimensional representation $\mathbf{z}$ of a (static) image $\mathbf{y}$ is insufficient to capture important dynamic information, such as velocities. Thus, we introduce an additional latent variable x, the *state*. In our case, the state $\mathbf{x}_k$ contains features from multiple time steps (e.g., $k-1$ and k) to capture velocity (or higher-order) information. Therefore, our transition model does not map features at time $k-1$ to time k (as illustrated in Figure 1), but the transition function $\mathbf{f}$ maps states $\mathbf{x}_k$ and control inputs $\mathbf{u}_k$ to states $\mathbf{x}_{k+1}$. The full dynamical system is given as the state-space model

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

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k; \boldsymbol{\theta}) + \mathbf{v}_k(\boldsymbol{\theta}), \tag{1b}$$

$$\mathbf{y}_k = \mathbf{g}(\mathbf{z}_k; \boldsymbol{\theta}) + \mathbf{e}_k(\boldsymbol{\theta}), \tag{1c}$$

where each measurement $\mathbf{y}_k$ can be described by a low-dimensional feature representation $\mathbf{z}_k$ (1c). These features are in turn modeled with a low-dimensional state-space model in (1a) and (1b), where the state $\mathbf{x}_k$ contains the full information about the state of the system at time instant k, see also Figure 2 (left). Here $\mathbf{w}_k(\boldsymbol{\theta})$, $\mathbf{v}_k(\boldsymbol{\theta})$ and $\mathbf{e}_k(\boldsymbol{\theta})$ are sequences of independent random variables and $\boldsymbol{\theta}$



(a) General graphical model *(b) Approximate prediction model*

Figure 2: *(a): The general graphical model: Each data point $\mathbf{y}_k$ has a low-dimensional representation $\mathbf{z}_k$, which is modeled using a state-space model with hidden state $\mathbf{x}_k$ and control input $\mathbf{u}_k$. (b): The approximate prediction model: The predicted feature $\hat{\mathbf{z}}_{k|k-1}$ is a function of the n past features $\mathbf{z}_{k-n}$ to $\mathbf{z}_{k-1}$ and n past control inputs $\mathbf{u}_{k-n}$ to $\mathbf{u}_{k-1}$. Each of the features $\mathbf{z}_{k-n}$ to $\mathbf{z}_{k-1}$ is computed from high-dimensional data $\mathbf{y}_{k-n}$ to $\mathbf{y}_{k-1}$ via the encoder $\mathbf{g}^{-1}$. The predicted feature $\hat{\mathbf{z}}_{k|k-1}$ is mapped to predicted high-dimensional data via the decoder $\mathbf{g}$.*

are the model parameters. The control inputs $\mathbf{u}_k$ will be important in controller design, which is further elaborated upon in Wahlström et al. (2015).

1.1 Approximate Prediction Model

To identify parameters in dynamical systems, the prediction-error method will be used, which requires a prediction model. In general, it is difficult to derive a prediction model based on the nonlinear state-space model (1), and a closed-form expression for the prediction is only available in a few special cases (Ljung, 1999). However, by approximating the optimal solution, a nonlinear autoregressive exogenous model (NARX) (Ljung, 1999) can be used

$$\hat{\mathbf{z}}_{k|k-1}(\boldsymbol{\theta}_{\mathrm{M}}) = \mathbf{f}(\mathbf{z}_{k-1}, \mathbf{u}_{k-1}, \ldots, \mathbf{z}_{k-n}, \mathbf{u}_{k-n}; \boldsymbol{\theta}_{\mathrm{M}}), \tag{2}$$

where $\mathbf{f}$ is a nonlinear function, in our case a neural network and $\boldsymbol{\theta}_{\mathrm{M}}$ is the corresponding model parameters. The model parameters in the nonlinear function are normally estimated by minimizing the sum of the prediction errors $\|\mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1}(\boldsymbol{\theta}_{\mathrm{M}})\|$. However, as we are interested in a good predictive performance for the high-dimensional data $\mathbf{y}$ rather than for the features $\mathbf{z}$, we transform the predictions back to the high-dimensional space and obtain a prediction $\hat{\mathbf{y}}_{k|k-1} = \mathbf{g}(\hat{\mathbf{z}}_{k|k-1}; \boldsymbol{\theta}_{\mathrm{D}})$, which we use in our error measure.

An additional complication is that we do not have access to the features $\mathbf{z}_k$. Therefore, before training, the past values of the time series have to be replaced with their feature representation $\mathbf{z} = \mathbf{g}^{-1}(\mathbf{y}; \boldsymbol{\theta}_{\mathrm{E}})$, which we compute from the pixel information $\mathbf{y}$. Here, $\mathbf{g}^{-1}$ is an approximate inverse of $\mathbf{g}$, which will be described

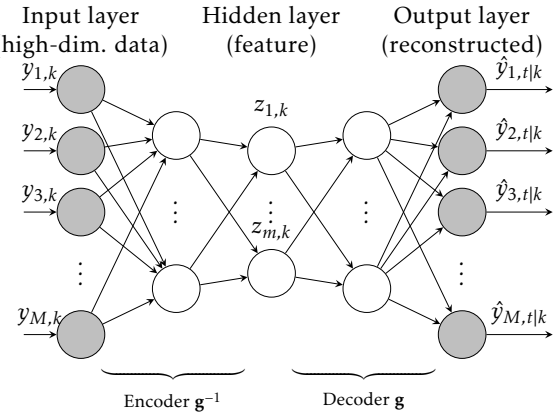


Figure 3: *An auto-encoder consisting of an encoder $\mathbf{g}^{-1}$ and a decoder $\mathbf{g}$. The original image $\mathbf{y}_k = [y_{1,k}, \cdots, y_{M,k}]^\mathsf{T}$ is mapped into its low-dimensional representation $\mathbf{z}_k = [z_{1,k}, \cdots, z_{m,k}]^\mathsf{T} = \mathbf{g}^{-1}(\mathbf{y}_k)$ with the encoder, and then back to a high-dimensional representation $\hat{\mathbf{y}}_{k|k-1} = \mathbf{g}(\hat{\mathbf{z}}_{k|k-1})$ by the decoder $\mathbf{g}$, where $M \gg m$.*

in more detail the next section. This gives the final prediction model

$$\hat{\mathbf{y}}_{k|k-1}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}, \boldsymbol{\theta}_\mathrm{M}) = \mathbf{g}(\hat{\mathbf{z}}_{k|k-1}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{M}); \boldsymbol{\theta}_\mathrm{D}), \tag{3a}$$

$$\hat{\mathbf{z}}_{k|k-1}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{M}) = \mathbf{f}(\mathbf{z}_{k-1}(\boldsymbol{\theta}_\mathrm{E}), \mathbf{u}_{k-1}, \ldots, \mathbf{z}_{k-n}(\boldsymbol{\theta}_\mathrm{E}), \mathbf{u}_{k-n}; \boldsymbol{\theta}_\mathrm{M}),$$

$$\mathbf{z}_k(\boldsymbol{\theta}_\mathrm{E}) = \mathbf{g}^{-1}(\mathbf{y}_k; \boldsymbol{\theta}_\mathrm{E}), \tag{3b}$$

which is also illustrated in Figure 2 (right). The corresponding prediction error will be

$$\boldsymbol{\varepsilon}^\mathrm{P}_k(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}, \boldsymbol{\theta}_\mathrm{M}) = \mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}, \boldsymbol{\theta}_\mathrm{M}). \tag{4}$$

1.2 Auto-Encoder

We use a deep auto-encoder neural network to parameterize the feature mapping and its inverse. It consists of a deep encoder network $\mathbf{g}^{-1}$ and a deep decoder network $\mathbf{g}$. Each layer i of the encoder neural network $\mathbf{g}^{-1}$ computes $\mathbf{y}^{(i+1)}_k = \sigma(A_i \mathbf{y}^{(i)}_k + \mathbf{b}_i)$, where σ is an activation function and A_i and $\mathbf{b}_i$ are free parameters. The control input to the first layer is the image, i.e., $\mathbf{y}^{(1)}_k = \mathbf{y}_k$. The last layer is the low-dimensional feature representation of the image $\mathbf{z}_k(\boldsymbol{\theta}_\mathrm{E}) = \mathbf{g}^{-1}(\mathbf{y}_k; \boldsymbol{\theta}_\mathrm{E})$, where $\boldsymbol{\theta}_\mathrm{E} = [\ldots, A_i, \mathbf{b}_i, \ldots]$ are the parameters of all neural network layers. The decoder $\mathbf{g}$ consists of the same number of layers in reverse order, see Figure 3, and can be considered an approximate inverse of the encoder $\mathbf{g}$, such that $\hat{\mathbf{y}}_{k|k}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}) \approx \mathbf{y}_k$, where

$$\hat{\mathbf{y}}_{k|k}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}) = \mathbf{g}(\mathbf{g}^{-1}(\mathbf{y}_k; \boldsymbol{\theta}_\mathrm{E}); \boldsymbol{\theta}_\mathrm{D}) \tag{5}$$

is the reconstructed version of $\mathbf{y}_k$. The encoder and decoder are trained jointly to minimize the reconstruction error

$$\varepsilon_k^{\mathrm{R}}(\boldsymbol{\theta}_{\mathrm{E}}, \boldsymbol{\theta}_{\mathrm{D}}) = \mathbf{y}_k - \hat{\mathbf{y}}_{k|k}(\boldsymbol{\theta}_{\mathrm{E}}, \boldsymbol{\theta}_{\mathrm{D}}), \tag{6}$$

and the parameters $\boldsymbol{\theta}_{\mathrm{D}}$, $\boldsymbol{\theta}_{\mathrm{E}}$ of $\mathbf{g}$ and $\mathbf{g}^{-1}$, respectively, can be coupled to constrain the solution to some degree (Vincent et al., 2008).

The auto-encoder suits our system identification problem well, since it provides an explicit expression of both the mapping $\mathbf{g}$ as well as its approximate inverse $\mathbf{g}^{-1}$, which we need for the predictions in (3a).

2 Training

To summarize, our model contains the following free parameters: the parameters for the encoder $\boldsymbol{\theta}_{\mathrm{E}}$, the parameters for the decoder $\boldsymbol{\theta}_{\mathrm{D}}$ and the parameters for the prediction model $\boldsymbol{\theta}_{\mathrm{M}}$. To train the model, we employ two cost functions, the sum of the prediction errors (4),

$$V_{\mathrm{P}}(\boldsymbol{\theta}_{\mathrm{E}}, \boldsymbol{\theta}_{\mathrm{D}}, \boldsymbol{\theta}_{\mathrm{M}}) = \sum_{k=1}^{N} \|\varepsilon_k^{\mathrm{P}}(\boldsymbol{\theta}_{\mathrm{E}}, \boldsymbol{\theta}_{\mathrm{D}}, \boldsymbol{\theta}_{\mathrm{M}})\|^2, \tag{7a}$$

and the sum of the reconstruction errors (6),

$$V_{\mathrm{R}}(\boldsymbol{\theta}_{\mathrm{E}}, \boldsymbol{\theta}_{\mathrm{D}}) = \sum_{k=1}^{N} \|\varepsilon_k^{\mathrm{R}}(\boldsymbol{\theta}_{\mathrm{E}}, \boldsymbol{\theta}_{\mathrm{D}})\|^2. \tag{7b}$$

Generally, there are two ways of finding the model parameters: (i) separate training and (ii) joint training of the auto-encoder and the prediction model, both of which are explained below.

2.1 Separate Training

Normally when features are used for learning dynamical models, they are first extracted from the data in a pre-processing step. In a second step the prediction model is estimated based on these features. In our setting, this corresponds to sequentially training the model using two cost functions (7a)–(7b): We first learn a compact feature representation by minimizing the reconstruction error

$$\left(\hat{\boldsymbol{\theta}}_{\mathrm{E}}, \hat{\boldsymbol{\theta}}_{\mathrm{D}}\right) \in \operatorname*{arg\,min}_{\boldsymbol{\theta}_{\mathrm{E}}, \boldsymbol{\theta}_{\mathrm{D}}} V_{\mathrm{R}}(\boldsymbol{\theta}_{\mathrm{E}}, \boldsymbol{\theta}_{\mathrm{D}}), \tag{8a}$$

and, subsequently, train the prediction model by minimizing the prediction error

$$\hat{\boldsymbol{\theta}}_{\mathrm{M}} = \operatorname*{arg\,min}_{\boldsymbol{\theta}_{\mathrm{M}}} V_{\mathrm{P}}(\hat{\boldsymbol{\theta}}_{\mathrm{E}}, \hat{\boldsymbol{\theta}}_{\mathrm{D}}, \boldsymbol{\theta}_{\mathrm{M}}), \tag{8b}$$

with fixed auto-encoder parameters $\hat{\boldsymbol{\theta}}_{\mathrm{E}}, \hat{\boldsymbol{\theta}}_{\mathrm{D}}$. The gradients of these cost functions with respect to the model parameters can be computed efficiently by back-propagation. The cost functions are then minimized by the BFGS algorithm (Nocedal and Wright, 2006).

2.2 Joint Training

An alternative to separate training is to minimize the reconstruction error and the prediction error jointly by considering the optimization problem

$$\left(\hat{\theta}_{\mathrm{E}}, \hat{\theta}_{\mathrm{D}}, \hat{\theta}_{\mathrm{M}}\right) = \underset{\theta_{\mathrm{E}}, \theta_{\mathrm{D}}, \theta_{\mathrm{M}}}{\arg\min} \; \left(V_{\mathrm{R}}(\theta_{\mathrm{E}}, \theta_{\mathrm{D}}) + V_{\mathrm{P}}(\theta_{\mathrm{E}}, \theta_{\mathrm{D}}, \theta_{\mathrm{M}})\right), \tag{9}$$

where we jointly optimize the free parameters in both the auto-encoder θ_{E}, θ_{D} and the prediction model θ_{M}. Again, back-propagation is used for computing the gradients of this cost function. Note that in (9) it is crucial to include not only the prediction error V_{P}, but also the reconstruction error V_{R}. Without this term the multi-step ahead prediction performance will decrease because predicted features are not consistent with features achieved from the encoder. The multi-step ahead predictive performance is crucial to design a controller for this system (Wahlström et al., 2015).

2.3 Initialization

With a linear activation function the auto-encoder and principal component analysis (PCA) are identical Bourlard and Kamp (1988), which we exploit to initialize the parameters of the auto-encoder: The auto-encoder network is unfolded, each pair of layers in the encoder and the decoder are combined, and the corresponding PCA solution is computed for each of these pairs. We start with high-dimensional image data at the top layer and use the principal components from that pair of layers as input to the next pair of layers. Thereby, we recursively compute a good initialization for all parameters of the auto-encoder. Similar pre-training routines are found in Hinton and Salakhutdinov (2006), in which a restricted Boltzmann machine is used instead of PCA.

3 Results

We report results on identification of the nonlinear dynamics of a planar pendulum (1-link robot arm) and the torque as control input. In this example, we learn the dynamics solely based on pixel information. Each pixel $y_k^{(i)}$ is a component of the measurement $\mathbf{y}_k = [y_k^{(1)}, \ldots, y_k^{(M)}]^{\mathsf{T}}$ and assumes a continuous gray-value in $[0, 1]$. In Wahlström et al. (2015) an model predictive controller is used to compute the control inputs, whereas we in this work use random control inputs.

We simulated 500 frames of a pendulum moving in a plane with $51 \times 51 = 2\,601$ pixels in each frame. To speed up training, the image input has been reduced to $\dim(\mathbf{y}_k) = 50$ prior to model learning (system identification) using PCA. With these 50 dimensional inputs, four layers have been used for the encoder $\mathbf{g}^{-1}$ as well as the decoder $\mathbf{g}$ with dimension 50-25-12-6-2. Hence, the features have dimension $\dim(\mathbf{x}_k) = 2$. The order of the dynamics was chosen as $n = 2$ to capture velocity information. For the prediction model $\mathbf{f}$ we used a two-layer neural network with a 6-4-2 architecture.

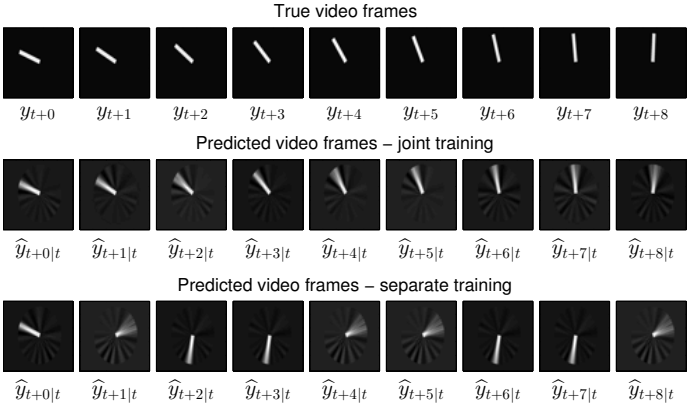


Figure 4: *A typical image sequence and corresponding prediction results (validation data), computed according to* (10)*. The top rows show nine consecutive ground truth image frames from time instant k to $k+8$. The second and the third rows display the corresponding long-term ahead predictions based on measured images up to time k for both joint (center) and separate training (bottom) of the model parameters.*

We evaluate the performance in terms of long-term predictions where we assumed that a sequence of open-loop torques was given. These predictions are constructed by concatenating multiple 1-step ahead predictions. More precisely, the p-step ahead prediction $\hat{\mathbf{y}}_{k+p|k} = \mathbf{g}(\hat{\mathbf{z}}_{k+p|k})$ is computed iteratively as

$$\hat{\mathbf{z}}_{k+1|k} = \mathbf{f}(\hat{\mathbf{z}}_{k|k}, \mathbf{u}_k, \dots), \tag{10a}$$

$$\dots$$

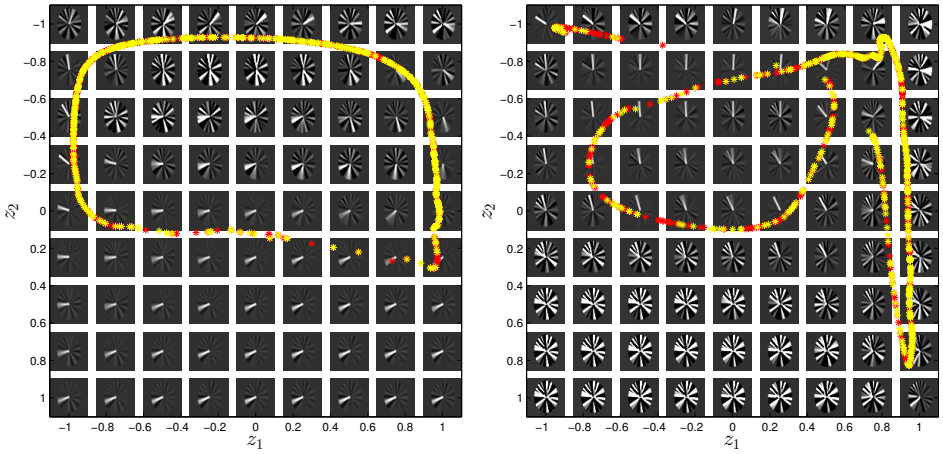
$$\hat{\mathbf{z}}_{k+p|k} = \mathbf{f}(\hat{\mathbf{z}}_{k+p-1|k}, \mathbf{u}_{k+p-1|k}, \dots), \tag{10b}$$

where $\hat{\mathbf{z}}_{k|k} = \mathbf{g}^{-1}(\mathbf{y}_k)$ are the image features at time k.

The predictive performance on an exemplary image sequence of the validation data of our system identification models is illustrated in Figure 4. The top row shows the ground truth images, the center row shows the predictions based on a model using joint training (9), the bottom row shows the corresponding predictions of a model where the auto-encoder and the predictive model were trained sequentially according to (8). The model that jointly learns all parameters yields a good predictive performance for both one-step ahead prediction and multiple-step ahead prediction. Compared to this, the predictive performance of the model that learns features and the dynamics separately is worse. Although the auto-encoder does a perfect job (left-most frame, 0-step ahead prediction), already the (reconstructed) one-step ahead prediction is dissimilar to the ground-truth image. This is also shown in Table 1 where the reconstruction error is equally good for both models, but for the prediction error we manage to get a better value using joint training than using separate training. Let us have a closer look at the model based on separate training: As the auto-encoder

Table 1: *Prediction error V_P and reconstruction error V_R for separate and joint training.*

Training	V_P	V_R
Joint training (9)	0.0011	0.0011
Separate training (8)	0.0051	0.0011



(a) *Joint learning* ***(b)*** *Separate learning*

Figure 5: *The feature space $\mathbf{z} \in [-1, 1] \times [-1, 1]$ is divided into 9×9 grid points. For each grid point the decoded high-dimensional image is displayed. The features corresponding to the training (red) and validation (yellow) data are displayed. Feature spaces found by joint (left) and separate (right) parameter learning.*

performs well, the learned transition model is the cause of bad predictive performance. We believe that the auto-encoder found a good feature representation for reconstruction, but this representation was not ideal for learning a transition model.

Figure 5 displays the "decoded" images corresponding to the latent representations using joint and separate training, respectively. After joint training the relevant features line up in a circular shape, such that a relatively simple prediction model is sufficient to describe the dynamics. However, for the separate training such an advantageous structure of the feature values are not obtained. Separate training extracts the low-dimensional features without context, i.e., the knowledge that these features constitute a time series.

In this particular data set, the data points clearly reside on one-dimensional manifold, encoded by the pendulum angle. However, a one-dimensional feature space would be insufficient since this one-dimensional manifold is cyclic, see

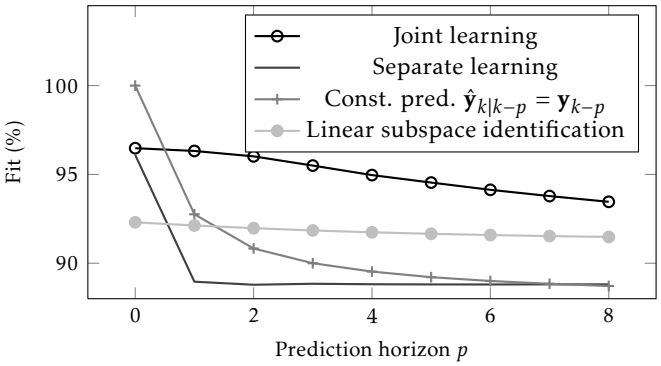


Figure 6: *Fitting quality* (11) *for joint and separate learning of features and dynamics for different prediction horizons p. The fit is compared with the naive prediction $\hat{\mathbf{y}}_{k|k-p} = \mathbf{y}_{k-p}$, where the most recent image is used and a linear subspace-ID method.*

Figure 5, compare also with the 2π period of an angle. Therefore, we have used a two-dimensional latent space.

To analyze the long-term predictive performance of both training methods, we define the fitting quality as

$$\mathrm{FIT}_p = 1 - \sqrt{\frac{1}{NM} \sum_{k=1}^{N} \|\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-p}\|^2}. \tag{11}$$

As a reference, the predictive performance is compared with a baseline prediction using the previous frame at time step $k - p$ as the prediction at k as $\hat{\mathbf{y}}_{k|k-p} = \mathbf{y}_{k-p}$.

The result for a prediction horizon ranging from $p = 0$ to $p = 8$ is displayed in Figure 6. Clearly, joint learning (blue) outperforms separate learning in terms of predictive performance for prediction horizons greater than 0. Even by using the last available image frame for prediction (const. pred., brown), we obtain a better fit than the model that learns its parameter sequentially (red). This is due to the fact that the dynamical model often predicts frames, which do not correspond to any real pendulum, see Figure 4, leading to a poor fit. Furthermore, joint training gives better predictions than the naive constant prediction. The predictive performance slightly degrades when the prediction horizon p increases, which is to be expected. Finally we also compare with the subspace identification method (Van Overschee and De Moor, 1996) (black, starred), which is restricted to linear models. Such a restriction does not capture the nonlinear, embedded features and, hence, the predictive performance is sub-optimal.

4 Discussion

From a system identification point of view, the prediction-error method, where we minimize the one-step ahead prediction error, is fairly standard. However,

in a control or reinforcement learning setting (Wahlström et al., 2015), we are primarily interested in good predictive performance on a longer horizon to do planning. Thus, we have also investigated to include a multi-step ahead prediction error in the cost (4). These models achieved similar performance, but no significantly better prediction error could be observed either for one-step ahead predictions nor for longer prediction horizons.

Instead of computing the prediction errors in image space, see (4), we can compute errors directly in feature space. However, this will require an extra penalty term to avoid trivial solutions that map everything to zero, resulting in a more complicated and less intuitive cost function.

Although joint learning aims at finding a feature representation that is suitable for modeling the low-dimensional dynamical behavior, the pre-training initialization as described in Section 2.3 does not. If this pre-training yields feature values far from "useful" ones for modeling the dynamics, joint training might not find a good model.

The autoencoder structure has to be chosen before the actual training starts. Especially the dimension of the latent state and the order of the dynamics have to be chosen by the user, which requires some prior knowledge about the system to be identified. In our examples, we chose the latent dimensionality based on insights about the true dynamics of the problem. In general, a model selection procedure will be preferable to find both a good network structure and a good latent dimensionality.

5 Conclusions and Future Work

We have presented an approach to nonlinear system identification from high-dimensional time series data. Our model combines techniques from both the system identification and the machine learning community. In particular, we used a deep auto-encoder for finding low-dimensional features from high-dimensional data, and a nonlinear autoregressive exogenous model was used to describe the low-dimensional dynamics. The framework has been applied to a identifying the dynamics of a planar pendulum from image pixels. The proposed model exhibits good long-term predictive performance, and a major advantage has been identified by training the auto-encoder and the dynamical model jointly compared to training them sequentially.

Possible directions for future work include (i) robustify learning by using denoising autoencoders (Vincent et al., 2008) to deal with noisy real-world data; (ii) apply convolutional neural networks, which are often more suitable for images; (iii) continue the work in Wahlström et al. (2015) using the model for learning controllers purely based on pixel information; (iv) investigate Sequential Monte Carlo methods for systematic treatments of such nonlinear probabilistic models, which are required in a reinforcement learning setting.

In a setting where we make decisions based on predictions, such as optimal control or model-based reinforcement learning, a probabilistic model is often needed for robust decision making as we need to account for model er-

rors (Schneider, 1997; Deisenroth et al., 2015). An extension of our model to a probabilistic setting is non-trivial since random variables have to be transformed through the neural networks, and their exact probability density functions will be intractable to compute. Sampling-based approaches or deterministic approximate inference are two options that we will investigate in future.

Bibliography

H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics*, 59(4-5):291–294, 1988.

M. P. Deisenroth, D. Fox, and C. E. Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):408–423, 2015.

G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto. Dynamic textures. *International Journal of Computer Vision*, 51(2):91–109, 2003.

Z. Ghahramani. Learning dynamic bayesian networks. In *Adaptive Processing of Sequences and Data Structures*, volume 1387 of *Lecture Notes in Computer Science*, pages 168–197. Springer, 1998.

S. Gibson and B. Ninness. Robust maximum-likelihood estimation of multivariable dynamic systems. *Automatica*, 41(10):1667–1682, 2005.

G. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

N. Kantas, A. Doucet, S. S. Singh, J. Maciejowski, and N. Chopin. On particle methods for parameter estimation in state-space models. Pre-print arXiv:1412.8695, 2014.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

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

L. Ljung. Perspectives on system identification. *Annual Reviews in Control*, 34 (1):1–12, 2010.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2006.

J. G. Schneider. Exploiting model uncertainty estimates for safe dynamic control learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1047–1053, 1997.

T. B. Schön, A. Wills, and B. Ninness. System identification of nonlinear state-space models. *Automatica*, 47(1):39–49, 2011.

R. H. Shumway and D. S. Stoffer. An approach to time series smoothing and forecasting using the EM algorithm. *Journal of Time Series Analysis*, 3(4):253–264, 1982.

J. Sjöberg, Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P.-Y. Glorennec, H. Hjalmarsson, and A. Juditsky. Nonlinear black-box modeling in system identification: a unified overview. *Automatica*, 31(12):1691–1724, 1995.

P. Van Overschee and B. De Moor. *Subspace identification for linear systems - theory, implementation, applications*. Kluwer Academic Publishers, 1996.

P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, Helsinki, Fnland, July 2008.

N. Wahlström, T. B. Schön, and M. P. Deisenroth. Learning deep dynamical models from image pixels. In *Proceedings of the 17th IFAC Symposium on System Identification (SYSID)*, Bejing, China, October 2015.

N. Wahlström, T. B. Schön, and M. P. Deisenroth. From Pixels to Torques: Policy Learning with Deep Dynamical Models. *Pre-print arXiv:1502.02251*, February 2015.

Paper F

From Pixels to Torques: Policy Learning with Deep Dynamical Models

Authors: Niklas Wahlström, Thomas B. Schön, Marc P. Desienroth

Edited version of the paper:

N. Wahlström, T. B. Schön, and M. P. Deisenroth. From pixels to torques: Policy learning with deep dynamical models. In *Deep Learning Workshop at the International Conference on Machine Learning (ICML)*, Lille, France, July 2015b.

From Pixels to Torques: Policy Learning with Deep Dynamical Models

Niklas Wahlström*, Thomas B. Schön†, Marc P. Desienroth‡

*Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
nikwa@isy.liu.se

†Dept. of Information Technology,
Uppsala University, Sweden
thomas.schon@it.uu.se

‡Dept. of Computing, Imperial College
London, United Kingdom
m.deisenroth@imperial.ac.uk

Abstract

Data-efficient learning in continuous state-action spaces using very high-dimensional observations remains a key challenge in developing fully autonomous systems. In this paper, we consider one instance of this challenge, the pixels to torques problem, where an agent must learn a closed-loop control policy from pixel information only. We introduce a data-efficient, model-based reinforcement learning algorithm that learns such a closed-loop policy directly from pixel information. The key ingredient is a deep dynamical model that uses deep auto-encoders to learn a low-dimensional embedding of images jointly with a predictive model in this low-dimensional feature space. Joint learning ensures that not only static but also dynamic properties of the data are accounted for. This is crucial for long-term predictions, which lie at the core of the adaptive model predictive control strategy that we use for closed-loop control. Compared to state-of-the-art reinforcement learning methods for continuous states and actions, our approach learns quickly, scales to high-dimensional state spaces and is an important step toward fully autonomous learning from pixels to torques.

1 Introduction

The vision of fully autonomous and intelligent systems that learn by themselves has influenced AI and robotics research for many decades. To devise fully autonomous systems, it is necessary to (1) process perceptual data (e.g., images) to summarize knowledge about the surrounding environment and the system's behavior in this environment, (2) make decisions based on uncertain and incomplete information, (3) take new information into account for learning and adapta-

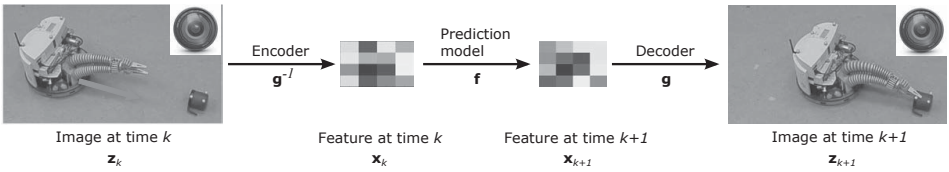


Figure 1: *Illustration of our idea of combining deep learning architectures for feature learning and prediction models in feature space. A camera observes a robot approaching an object. A good low-dimensional feature representation of an image is important for learning a predictive model if the camera is the only sensor available.*

tion. Effectively, any fully autonomous system has to close this perception-action-learning loop without relying on specific human expert knowledge. The *pixels to torques problem* (Brock, 2011) identifies key aspects of an autonomous system: autonomous thinking and decision making using sensor measurements only, intelligent exploration and learning from mistakes.

We consider the problem of learning closed-loop policies ("torques") from pixel information end-to-end. A possible scenario is a scene in which a robot is moving about. The only available sensor information is provided by a camera, i.e., no direct information of the robot's joint configuration is available. The objective is to learn a continuous-valued policy that allows the robotic agent to solve a task in this continuous environment in a data-efficient way, i.e., we want to keep the number of trials small. To date, there is no fully autonomous system that convincingly closes the perception-action-learning loop and solves the pixels to torques problem in continuous state-action spaces, the natural domains in robotics.

A promising approach toward solving the pixels to torques problem is Reinforcement Learning (RL) (Sutton and Barto, 1998), a principled mathematical framework that deals with fully autonomous learning from trial and error. However, one practical shortcoming of many existing RL algorithms is that they require many trials to learn good policies, which is prohibitive when working with real-world mechanical plants or robots.

One way of using data efficiently (and therefore keep the number of experiments small) is to learn forward models of the underlying dynamical system, which are then used for internal simulations and policy learning. These ideas have been successfully applied to RL, control and robotics by Schmidhuber (1990); Atkeson and Schaal (1997); Bagnell and Schneider (2001); Contardo et al. (2013); Pan and Theodorou (2014); Deisenroth et al. (2015); Pan and Theodorou (2014); van Hoof et al. (2015); Levine et al. (2015), for instance. However, these methods use heuristic or engineered low-dimensional features, and they do not easily scale to data-efficient RL using pixel information only because even "small" images possess thousands of dimensions.

A common way of dealing with high-dimensional data is to learn low-dimensional feature representations. Deep learning architectures, such as deep neural networks (Hinton and Salakhutdinov, 2006), stacked auto-encoders (Ben-

gio et al., 2007; Vincent et al., 2008), or convolutional neural networks (LeCun et al., 1998), are the current state of the art in learning parsimonious representations of high-dimensional data. Deep learning has been successfully applied to image, text and speech data in commercial products, e.g., by Google, Amazon and Facebook.

Deep learning has been used to produce first promising results in the context of model-free RL on images: For instance, Mnih et al. (2015) present an approach based on Deep-Q-learning, in which human-level game strategies are learned autonomously, purely based on pixel information. Moreover, Lange et al. (2012) presented an approach that learns good discrete actions to control a slot car based on raw images, employing deep architectures for finding compact low-dimensional representations. Other examples of deep learning in the context of RL on image data include Cuccu et al. (2011); Koutnik et al. (2013). These approaches have in common that they try to estimate the value function from which the policy is derived. However, neither of these algorithms learns a predictive model and are, therefore, prone to data inefficiency, either requiring data collection from millions of experiments or relying on discretization and very low-dimensional feature spaces, limiting their applicability to mechanical systems.

To increase data efficiency, we therefore introduce a model-based approach to learning from pixels to torques. In particular, exploit results from Wahlström et al. (2015a) and jointly learn a lower-dimensional embedding of images and a transition function in this lower-dimensional space that we can use for internal simulation of the dynamical system. For this purpose, we employ deep auto-encoders for the lower-dimensional embedding and a multi-layer feed-forward neural network for the transition function. We use this deep dynamical model to predict trajectories and apply an adaptive model-predictive-control (MPC) algorithm (Mayne, 2014) for online closed-loop control, which is practically based on pixel information only.

MPC has been well explored in the control community, However, adaptive MPC has so far not received much attention in the literature (Mayne, 2014). An exception is Sha (2008), where the authors advocate a neural network approach similar to ours. However, they do not consider high-dimensional data but assume that they have direct access to low-dimensional measurements.

Our approach benefits from the application of model-based optimal control principles within a machine learning framework. Along these lines, Deisenroth et al. (2009); Abramova et al. (2012); Boedecker et al. (2014); Pan and Theodorou (2014); Levine et al. (2015) suggested to first learn a transition model and then use optimal control methods to solve RL problems. Unlike these methods, our approach does not need to estimate value functions and scales to high-dimensional problems.

Similar to our approach, Boots et al. (2014); Levine et al. (2015); van Hoof et al. (2015) recently proposed model-based RL methods that learn policies directly from visual information. Unlike these methods, we exploit a low-dimensional feature representation that allows for fast predictions and online control learning via MPC.

Problem Set-up and Objective

We consider a classical N-step finite-horizon RL setting in which an agent attempts to solve a particular task by trial and error. In particular, our objective is to find a closed-loop policy π^* that minimizes the long-term cost $V^\pi = \sum_{k=0}^{N-1} \mathbf{f}_0(\mathbf{x}_k, \mathbf{u}_k)$, where $\mathbf{f}_0$ denotes an immediate cost, $\mathbf{x}_k \in \mathbb{R}^D$ is the continuous-valued system state and $\mathbf{u}_k \in \mathbb{R}^F$ are continuous control inputs.

The learning agent faces the following additional challenges: (i) The agent has no access to the true state, but perceives the environment only through high-dimensional pixel information (images), (ii) a good control policy is required in only a few trials. This setting is practically relevant, e.g., when the agent is a robot that is monitored by a video camera based on which the robot has to learn to solve tasks fully autonomously. Therefore, this setting is an instance of the pixels to torques problem.

2 Deep Dynamical Model

Our approach to solve the pixels-to-torques problem is based on a deep dynamical model (DDM), which jointly (i) embeds high-dimensional images in a low-dimensional feature space via deep auto-encoders and (ii) learns a predictive forward model in this feature space (Wahlström et al., 2015a). In particular, we consider a DDM with control inputs $\mathbf{u}$ and high-dimensional observations $\mathbf{y}$. We assume that the relevant properties of $\mathbf{y}$ can be compactly represented by a feature variable $\mathbf{z}$. The two components of the DDM, i.e., the low-dimensional embedding and the prediction model, which predicts future observations $\mathbf{y}_{k+1}$ based on past observations and control inputs, are detailed in the following. Throughout this paper, $\mathbf{y}_k$ denotes the high-dimensional measurements, $\mathbf{z}_k$ the corresponding low-dimensional encoded features and $\hat{\mathbf{y}}_k$ the reconstructed high-dimensional measurement. Further, $\hat{\mathbf{z}}_{k+1}$ and $\hat{\mathbf{y}}_{k+1}$ denote a predicted feature and measurement at time $k+1$, respectively.

2.1 Deep Auto-Encoder

We use a deep auto-encoder for embedding images in a low-dimensional feature space, where both the encoder $\mathbf{g}^{-1}$ and the decoder $\mathbf{g}$ are modeled with deep neural networks. Each layer i of the *encoder* neural network $\mathbf{g}^{-1}$ computes $\mathbf{y}_k^{(i+1)} = \sigma(A_i \mathbf{y}_k^{(i)} + \mathbf{b}_i)$, where σ is a sigmoidal activation function (we used tangent hyperbolicus) and A_i and $\mathbf{b}_i$ are free parameters. The input to the first layer is the image, i.e., $\mathbf{y}_k^{(1)} = \mathbf{y}_k$. The last layer is the low-dimensional feature representation of the image $\mathbf{z}_k(\boldsymbol{\theta}_\mathrm{E}) = \mathbf{g}^{-1}(\mathbf{y}_k; \boldsymbol{\theta}_\mathrm{E})$, where $\boldsymbol{\theta}_\mathrm{E} = [\dots, A_i, \mathbf{b}_i, \dots]$ are the parameters of all neural network layers. The *decoder* $\mathbf{g}$ consists of the same number of layers in reverse order, see Figure 2, and approximately inverts the encoder $\mathbf{g}$, such that $\hat{\mathbf{y}}_k(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}) = \mathbf{g}(\mathbf{g}^{-1}(\mathbf{y}_k; \boldsymbol{\theta}_\mathrm{E}); \boldsymbol{\theta}_\mathrm{D}) \approx \mathbf{y}_k$ is the reconstructed version of $\mathbf{y}_k$ with an associated reconstruction error

$$\boldsymbol{\varepsilon}_k^\mathrm{R}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}) = \mathbf{y}_k - \hat{\mathbf{y}}_k(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}). \tag{1}$$

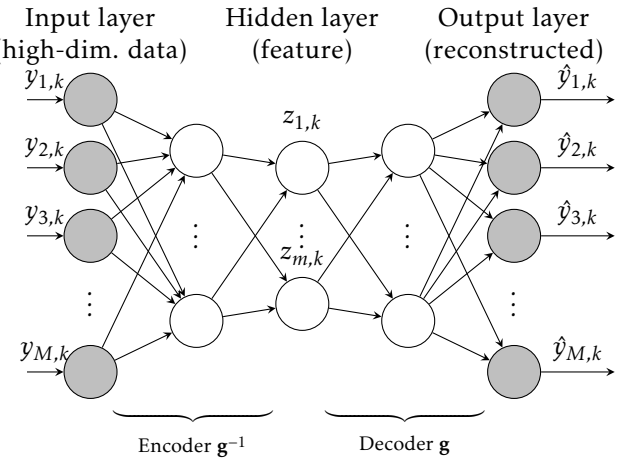


Figure 2: *Auto-encoder that consists of an encoder $\mathbf{g}^{-1}$ and a decoder $\mathbf{g}$. The encoder maps the original image $\mathbf{y}_k \in \mathbb{R}^M$ onto its low-dimensional representation $\mathbf{z}_k = \mathbf{g}^{-1}(\mathbf{y}_k) \in \mathbb{R}^m$, where $m \ll M$; the decoder maps this feature back to a high-dimensional representation $\hat{\mathbf{y}}_k = \mathbf{g}(\hat{\mathbf{z}}_k)$. The gray color represents high-dimensional observations.*

The main purpose of the deep auto-encoder is to keep this reconstruction error and the associated compression loss negligible, such that the features $\mathbf{z}_k$ are a compact representation of the images $\mathbf{y}_k$.

2.2 Prediction Model

We now turn the static auto-encoder into a dynamical model that can predict future features $\hat{\mathbf{z}}_{k+1}$ and images $\hat{\mathbf{y}}_{k+1}$. The encoder $\mathbf{g}^{-1}$ allows us to map high-dimensional observations $\mathbf{y}_k$ onto low-dimensional features $\mathbf{z}_k$. For predicting we assume that *future features* $\hat{\mathbf{z}}_{k+1|h_n}$ depend on an n-step history h_n of past features and control inputs, i.e.,

$$\hat{\mathbf{z}}_{k+1|h_n}(\boldsymbol{\theta}_\mathrm{P}) = \mathbf{f}(\mathbf{z}_k, \mathbf{u}_k, \ldots, \mathbf{z}_{k-n+1}, \mathbf{u}_{k-n+1}; \boldsymbol{\theta}_\mathrm{P}), \tag{2}$$

where $\mathbf{f}$ is a nonlinear transition function, in our case a feed-forward neural network, and $\boldsymbol{\theta}_\mathrm{P}$ are the corresponding model parameters. This is a nonlinear autoregressive exogenous model (NARX) (Ljung, 1999). The predictive performance of the model will be important for model predictive control (see Section 3) and for model learning based on the prediction error (Ljung, 1999).

To predict *future observations* $\hat{\mathbf{y}}_{k+1|h_n}$ we exploit the decoder, such that $\hat{\mathbf{y}}_{k+1|h_n} = \mathbf{g}(\hat{\mathbf{z}}_{k+1|h_n}; \boldsymbol{\theta}_\mathrm{D})$. The deep decoder $\mathbf{g}$ maps features $\mathbf{z}$ to high-dimensional observations $\mathbf{y}$ parameterized by $\boldsymbol{\theta}_\mathrm{D}$.

Now, we are ready to put the pieces together: With feature prediction model (2) and the deep auto-encoder, the DDM predicts future features and images ac-

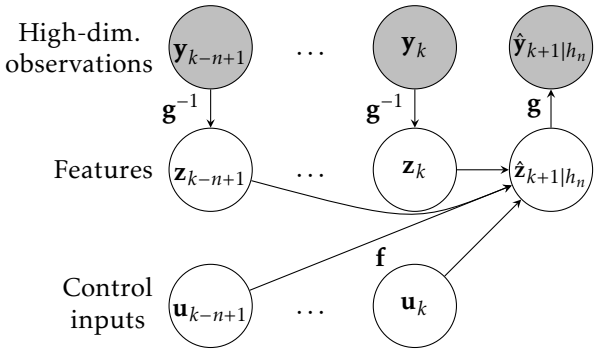


Figure 3: *Prediction model: Each feature $\mathbf{z}_i$ is computed from high-dimensional data $\mathbf{y}_i$ via the encoder $\mathbf{g}^{-1}$. The transition model predicts the feature $\hat{\mathbf{z}}_{k+1|h_n}$ at the next time step based on the n-step history of n past features $\mathbf{z}_{k-n+1}, \ldots, \mathbf{z}_k$ and control inputs $\mathbf{u}_{k-n+1}, \ldots, \mathbf{u}_k$. The predicted feature $\hat{\mathbf{z}}_{k+1|h_n}$ can be mapped to a high-dimensional prediction $\hat{\mathbf{y}}_{k+1}$ via the decoder $\mathbf{g}$. The gray color represents high-dimensional observations.*

cording to

$$\mathbf{z}_k(\boldsymbol{\theta}_\mathrm{E}) = \mathbf{g}^{-1}(\mathbf{y}_k; \boldsymbol{\theta}_\mathrm{E}), \tag{3a}$$

$$\hat{\mathbf{z}}_{k+1|h_n}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{P}) = \mathbf{f}(\mathbf{z}_k, \mathbf{u}_k, \ldots, \mathbf{z}_{k-n+1}, \mathbf{u}_{k-n+1}; \boldsymbol{\theta}_\mathrm{P}),$$

$$\hat{\mathbf{y}}_{k+1|h_n}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}, \boldsymbol{\theta}_\mathrm{P}) = \mathbf{g}(\hat{\mathbf{z}}_{k+1|h_n}; \boldsymbol{\theta}_\mathrm{D}), \tag{3b}$$

which is illustrated in Figure 3. With this prediction model we define the prediction error

$$\boldsymbol{\varepsilon}^\mathrm{P}_{k+1}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}, \boldsymbol{\theta}_\mathrm{P}) = \mathbf{y}_{k+1} - \hat{\mathbf{y}}_{k+1|h_n}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}, \boldsymbol{\theta}_\mathrm{P}), \tag{4}$$

where $\mathbf{y}_{k+1}$ is the observed image at time $k+1$.

2.3 Training

The DDM is parameterized by the encoder parameters $\boldsymbol{\theta}_\mathrm{E}$, the decoder parameters $\boldsymbol{\theta}_\mathrm{D}$ and the prediction model parameters $\boldsymbol{\theta}_\mathrm{P}$. In the DDM, we train both the prediction model and the deep auto-encoder jointly by finding parameters $\left(\hat{\boldsymbol{\theta}}_\mathrm{E}, \hat{\boldsymbol{\theta}}_\mathrm{D}, \hat{\boldsymbol{\theta}}_\mathrm{P}\right)$, such that

$$\left(\hat{\boldsymbol{\theta}}_\mathrm{E}, \hat{\boldsymbol{\theta}}_\mathrm{D}, \hat{\boldsymbol{\theta}}_\mathrm{P}\right) = \underset{\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}, \boldsymbol{\theta}_\mathrm{P}}{\arg\min}\ V_\mathrm{R}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}) + V_\mathrm{P}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}, \boldsymbol{\theta}_\mathrm{P}), \tag{5a}$$

$$V_\mathrm{P}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}, \boldsymbol{\theta}_\mathrm{P}) = \sum_{k=1}^{N} \|\boldsymbol{\varepsilon}^\mathrm{P}_k(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}, \boldsymbol{\theta}_\mathrm{P})\|^2, \tag{5b}$$

$$V_\mathrm{R}(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D}) = \sum_{k=1}^{N} \|\boldsymbol{\varepsilon}^\mathrm{R}_k(\boldsymbol{\theta}_\mathrm{E}, \boldsymbol{\theta}_\mathrm{D})\|^2, \tag{5c}$$

which minimizes the sums of squared reconstruction (1) and prediction (4) errors.

We learn all model parameters $\theta_E, \theta_D, \theta_P$ *jointly* by solving (5a). [1] The required gradients with respect to the parameters are computed efficiently by back-propagation, and the cost function is minimized by the BFGS algorithm (Nocedal and Wright, 2006). Note that in (5a) it is crucial to include not only the prediction error V_P, but also the reconstruction error V_R. Without this term the multi-step ahead prediction performance will decrease because predicted features are not consistent with features achieved from the encoder. Since we consider a control problem in this paper, multi-step ahead predictive performance is crucial.

Initialization. With a linear activation function the auto-encoder and PCA are identical (Bourlard and Kamp, 1988), which we exploit to initialize the parameters of the auto-encoder: The auto-encoder network is unfolded, each pair of layers in the encoder and the decoder are combined, and the corresponding PCA solution is computed for each of these pairs. We start with high-dimensional image data at the top layer and use the principal components from that pair of layers as input to the next pair of layers. Thereby, we recursively compute a good initialization for all parameters of the auto-encoder. Similar pre-training routines are found in Hinton and Salakhutdinov (2006), in which a restricted Boltzmann machine is used instead of PCA.

In this section, we have presented a DDM that facilitates fast predictions of high-dimensional observations via a low-dimensional embedded time series. The property of fast predictions will be exploited by the online feedback control strategy presented in the following. More details on the proposed model are given in Wahlström et al. (2015a).

3 Learning Closed-Loop Policies from Images

We use the DDM for learning a closed-loop policy by means of nonlinear model predictive control (MPC). We start off by an introduction to classical MPC, before moving on to MPC on images in Section 3.1. MPC finds an optimal sequence of control signals that minimizes a K-step loss function, where K is typically smaller than the full horizon. In general, MPC relies on (i) a reference trajectory $\mathbf{x}_{\text{ref}} = \mathbf{x}_1^*, \ldots, \mathbf{x}_K^*$ (which can be a constant reference signal) and (ii) a dynamics model

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), \tag{6}$$

which, assuming that the current state is denoted by $\mathbf{x}_0$, can be used to compute/predict a state trajectory $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_K$ for a given sequence $\mathbf{u}_0, \ldots, \mathbf{u}_{K-1}$ of control signals. Using the dynamics model MPC determines an optimal (open-loop)

[1] Normally when features are used for learning dynamical models, they are first extracted from the data in a pre-processing step by minimizing (5c) with respect to the auto-encoder parameters θ_E, θ_D. In a second step, the prediction model parameters θ_P are estimated based on these features by minimizing (5b) conditioned on the estimated $\hat{\theta}_E$ and $\hat{\theta}_D$. In our experience, a problem with this approach is that the learned features might have a small reconstruction error, but this representation will not be ideal for learning a transition model. Section 4.1 discusses this in more detail.

control sequence $\mathbf{u}_0^*, \ldots, \mathbf{u}_{K-1}^*$, such that the predicted trajectory $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_K$ gets as close to the reference trajectory $\mathbf{x}_{\text{ref}}$ as possible, such that

$$\mathbf{u}_0^*, \ldots, \mathbf{u}_{K-1}^* \in \arg\min_{\mathbf{u}_{0:K-1}} \sum_{k=0}^{K-1} \|\hat{\mathbf{x}}_k - \mathbf{x}_k^*\|^2 + \lambda \|\mathbf{u}_k\|^2, \tag{7}$$

where $\|\hat{\mathbf{x}}_k - \mathbf{x}_k^*\|^2$ is a cost associated with the deviation of the predicted state trajectory $\hat{\mathbf{x}}_{0:K-1}$ from the reference trajectory $\mathbf{x}_{\text{ref}}$, and $\|\mathbf{u}_k\|^2$ penalizes the amplitude of the control signals. Here, λ is a tuning parameter. When the control sequence $\mathbf{u}_0^*, \ldots, \mathbf{u}_{K-1}^*$ is determined, the first control $\mathbf{u}_0^*$ is applied to the system. After observing the next state, MPC repeats the entire optimization and turns the overall policy into a closed-loop (feedback) control strategy.

3.1 MPC on Images

We now turn the classical MPC procedure into MPC on images by exploiting some convenient properties of the DDM. The DDM allows us to predict *features* $\hat{\mathbf{z}}_1, \ldots, \hat{\mathbf{z}}_K$ based on a sequence of controls $\mathbf{u}_0, \ldots, \mathbf{u}_{K-1}$. By comparing (6) with (2), we define the *state* $\mathbf{x}_0$ as the present and past $n-1$ features and the past $n-1$ control inputs, such that

$$\mathbf{x}_0 = [\mathbf{z}_0, \ldots, \mathbf{z}_{-n+1}, \mathbf{u}_{-1}, \ldots, \mathbf{u}_{-n+1}]. \tag{8}$$

The DDM computes the present and past features with the encoder $\mathbf{z}_k = \mathbf{g}^{-1}(\mathbf{y}_k, \boldsymbol{\theta}_{\text{E}})$, such that $\mathbf{x}_0$ is known at the current time, which matches the MPC requirement. Our objective is to control the system towards a desired reference image frame $\mathbf{y}_{\text{ref}}$. This reference frame $\mathbf{y}_{\text{ref}}$ can also be encoded to a corresponding reference feature $\mathbf{z}_{\text{ref}} = \mathbf{g}^{-1}(\mathbf{y}_{\text{ref}}, \boldsymbol{\theta}_{\text{E}})$, which results in the MPC objective

$$\mathbf{u}_0^*, \ldots, \mathbf{u}_{K-1}^* \in \arg\min_{\mathbf{u}_{0:K-1}} \sum_{k=0}^{K-1} \|\hat{\mathbf{z}}_k - \mathbf{z}_{\text{ref}}\|^2 + \lambda \|\mathbf{u}_k\|^2. \tag{9}$$

The gradients of the cost function (9) with respect to the control signals $\mathbf{u}_0, \ldots, \mathbf{u}_{K-1}$ are computed in closed form, and we use BFGS to find the optimal sequence of control signals. Note that the objective function depends on $\mathbf{u}_0, \ldots, \mathbf{u}_{K-1}$ not only via the control penalty $\|\mathbf{u}_k\|^2$ but also via the feature predictions $\hat{\mathbf{z}}_k$ of the DDM via (2).

Overall, we now have an online MPC algorithm that, given a trained DDM, works indirectly on images by exploiting their feature representation. In the following, we will now turn this into an iterative algorithm that learns predictive models from images and good controllers from scratch.

3.2 Adaptive MPC for Learning from Scratch

We will now turn over to describe how (adaptive) MPC can be used together with our DDM to address the pixels to torques problem and to learn from scratch. At

the core of our MPC formulation lies the DDM, which is used to predict future states (8) from a sequence of control inputs. The quality of the MPC controller is inherently bound to the prediction quality of the dynamical model, which is typical in model-based RL (Schneider, 1997; Schaal, 1997; Deisenroth et al., 2015).

To learn models and controllers from scratch, we apply a control scheme that allows us to update the DDM as new data arrives. In particular, we use the MPC controller in an adaptive fashion to gradually improve the model by collecting data in the feedback loop without any specific prior knowledge of the system at hand. Data collection is performed in closed-loop (online MPC), and it is divided into multiple sequential trials. After each trial, we add the data of the most recent trajectory to the data set, and the model is re-trained using all data that has been collected so far.

Algorithm 1 Adaptive MPC in feature space

```
Follow a random control strategy and record data
loop
  Update DDM with all data collected so far
  for k = 0 to N − 1 do
    Get state x_k via auto-encoder
    u*_k ← ϵ-greedy MPC policy using DDM prediction
    Apply u*_k and record data
  end for
end loop
```

Simply applying the MPC controller based on a randomly initialized model would make the closed-loop system very likely to converge to a point, which is far away from the desired reference value, due to the poor model that cannot extrapolate well to unseen states. This would in turn imply that no data is collected in unexplored regions, including the region that we actually are interested in. There are two solutions to this problem: Either we use a probabilistic dynamics model as suggested in Schneider (1997); Deisenroth et al. (2015) to explicitly account for model uncertainty and the implied natural exploration or we follow an explicit exploration strategy to ensure proper excitation of the system. In this paper, we follow the latter approach. In particular, we choose an ϵ-greedy exploration strategy where the optimal feedback $\mathbf{u}_0^*$ at each time step is selected with a probability $1-\epsilon$, and a random action is selected with probability ϵ.

Algorithm 1 summarizes our adaptive online MPC scheme. We initialize the DDM with a random trial. We use the learned DDM to find an ϵ-greedy policy using predicted features within MPC. This happens online. The collected data is added to the data set and the DDM is updated after each trial.

4 Experimental Results

In the following, we empirically assess the components of our proposed methodology for autonomous learning from high-dimensional synthetic image data: (i)

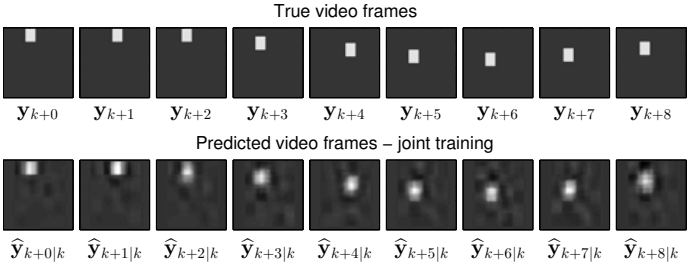


Figure 4: *Long-term (up to eight steps) predictive performance of the DDM: True (upper plot) and predicted (lower plot) video frames on test data.*

the quality of the learned DDM and (ii) the overall learning framework.

In both cases, we consider a sequence of images ($51 \times 51 = 2601$ pixels) and a control input associated with these images. Each pixel $y_k^{(i)}$ is a component of the measurement $\mathbf{y}_k \in \mathbb{R}^{2601}$ and assumes a continuous gray-value in the interval $[0, 1]$. No access to the underlying dynamics or the state (angle φ and angular velocity $\dot{\varphi}$) was available, i.e., we are dealing with a high-dimensional continuous state space. The challenge was to learn (i) a good dynamics model, and (ii) a good controller from pixel information only. We used a sampling frequency of 0.2 s and a time horizon of 25 s, which corresponds to 100 frames per trial.

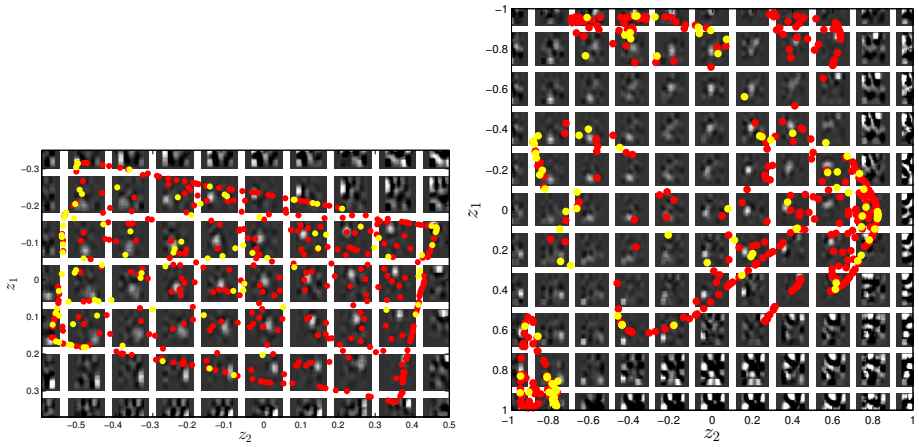
The input dimension has been reduced to $\dim(\mathbf{y}_k) = 50$ prior to model learning using PCA. With these 50-dimensional inputs, a four-layer auto-encoder network was used with dimension 50-25-12-6-2, such that the features were of dimension $\dim(\mathbf{z}_k) = 2$, which is optimal to model the periodic angle of the pendulum. The order of the dynamics was selected to be $n = 2$ (i.e., we consider two consecutive image frames) to capture velocity information, such that $\mathbf{z}_{k+1} = \mathbf{f}(\mathbf{z}_k, \mathbf{u}_k, \mathbf{z}_{k-1}, \mathbf{u}_{k-1})$. For the prediction model $\mathbf{f}$ we used a feedforward neural network with a 6-4-2 architecture. Note that the dimension of the first layer is given by $n(\dim(\mathbf{z}_k) + \dim(\mathbf{u}_k)) = 2(2 + 1) = 6$.

4.1 Learning Predictive Models from Pixels

To assess the predictive performance of the DDM, we took 601 screenshots of a moving tile, see Figure 4. The control inputs are the (random) increments in position in horizontal and vertical directions.

We evaluate the performance of the learned DDM in terms of long-term predictions, which play a central role in MPC for autonomous learning. Long-term predictions are obtained by concatenating multiple 1-step ahead predictions.

The performance of the DDM is illustrated in Figure 4 on a test data set. The top row shows the ground truth images and the bottom row shows the DDM's long-term predictions. The model predicts future frames of the tile with high accuracy both for 1-step ahead and multiple steps ahead. The model yields a good predictive performance for both one-step ahead prediction and multiple-step ahead prediction.



(a) *Autoencoder and prediction model* ***(b)*** *Only auto-encoder*

Figure 5: *Feature space for both joint (a) and sequential training (b) of auto-encoder and prediction model. The feature space is divided into grid points. For each grid point the decoded high-dimensional image is displayed and the feature values for the training data (red) and validation data (yellow) are overlain. For the joint training the feature values reside on a two-dimensional manifold that corresponds to the two-dimensional position of the tile. For the separate training the feature values are scattered without structure.*

In Figure 5a, the feature representation of the data is displayed. The features reside on a two-dimensional manifold that encodes the two-dimensional position of the moving tile. By inspecting the decoded images we can see that each corner of the manifold corresponds to a corner position of the tile. Due to this structure a relatively simple prediction model is sufficient to describe the dynamics. In case the auto-encoder and the prediction model would have been learned sequentially (first training the auto-encoder, and then based on these features values train the prediction model) such a structure would not have been enforced. In Figure 5b the corresponding feature representation is displayed where only the auto-encoder has been trained. Clearly, these features does not exhibit such a structure.

4.2 Closed-Loop Policy Learning from Pixels

In this section, we report results on learning a policy that moves a pendulum (1-link robot arm with length 1 m, weight 1 kg and friction coefficient 1 Nsm/rad) from a start position $\varphi = 0$ to a target position $\varphi = \pm\pi$. The reference signal was the screenshot of the pendulum in the target position. For the MPC controller, we used a planning horizon of $P = 15$ steps and a control penalty $\lambda = 0.01$. For the

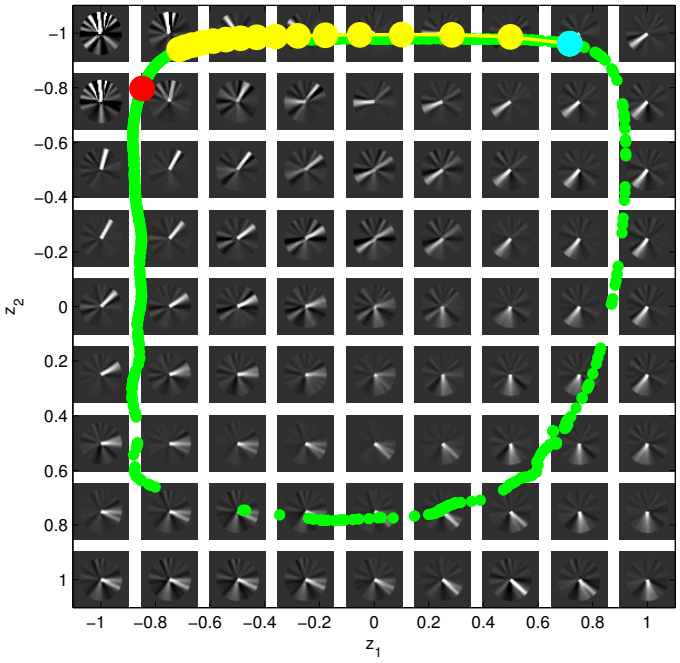


Figure 6: *The feature space* $\mathbf{z} \in [-1, 1] \times [-1, 1]$ *is divided into* 9×9 *grid points for illustration purposes. For each grid point the decoded high-dimensional image is displayed. Green: Feature values that correspond to collected experience in previous trials. Cyan: Feature value that corresponds to the current time step. Red: Desired reference value. Yellow: 15-steps-ahead prediction after optimizing for the optimal control inputs.*

ϵ-greedy exploration strategy we used $\epsilon = 0.2$. We conducted 50 independent experiments with different random initializations. The learning algorithm was run for 15 trials (plus an initial random trial). After each trial, we retrained the DDM using all collected data so far, where we also include the reference image while learning the auto-encoder.

Figure 6 displays the decoded images corresponding to learned latent representations in $[-1, 1]^2$. The learned feature values of the training data (green) line up in a circular shape, such that a relatively simple prediction model is sufficient to describe the dynamics. If we would not have optimized for both the prediction error and reconstruction error, such an advantageous structure of the feature values would not have been obtained. The DDM extracts features that can also model the *dynamic* behavior compactly. The figure also shows the predictions produced by the MPC controller (yellow), starting from the current time step (cyan) and targeting the reference feature (red) where the pendulum is in the target position.

To assess the controller performance after each trial, we applied a greedy pol-

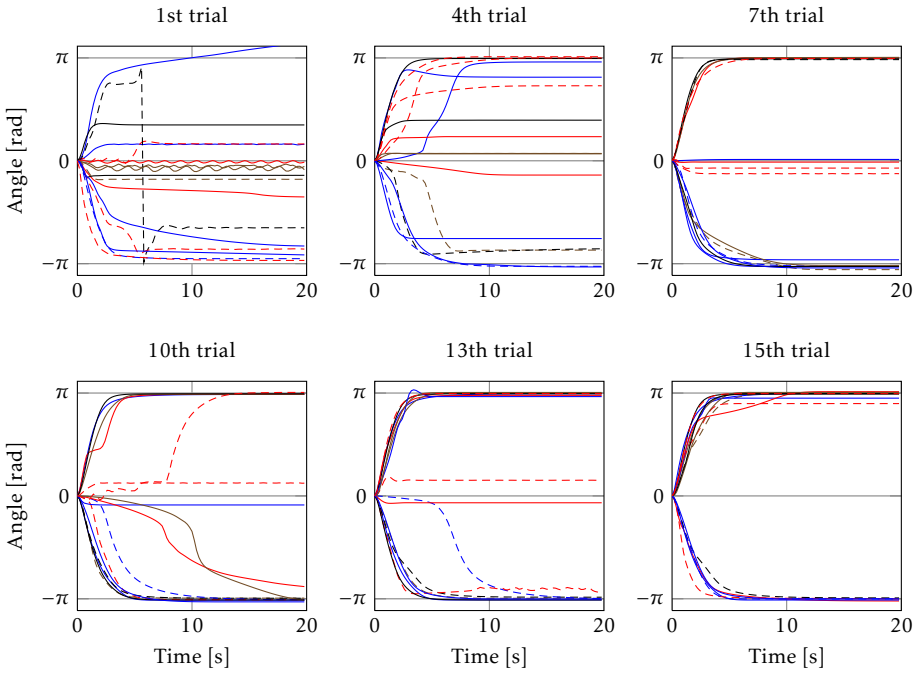


Figure 7: *Control performance after 1st to 15th trial evaluated with $\varepsilon = 0$ for 16 different experiments. The objective was to reach an angle of $\pm\pi$.*

icy ($\epsilon = 0$). In Figure 7, angle trajectories for 15 of the 50 experiments at different learning stages are displayed. In the first trial, the controller managed only in a few cases to drive the pendulum toward the reference value $\pm\pi$. The control performance increased gradually with the number of trials, and after the 15th trial, it manages in most cases to get it to an upright position.

To assess the data efficiency of our approach, we compared it with the PILCO RL framework (Deisenroth et al., 2015) to learning closed-loop control policies for the pendulum task above. PILCO is a current state-of-the art model-based RL algorithm for data-efficient learning of control policies in continuous state-control spaces. Using collected data PILCO learns a probabilistic model of the system dynamics, implemented as a Gaussian process (GP) (Rasmussen and Williams, 2006). Subsequently, this model is used to compute a distribution over trajectories and the corresponding expected cost, which is used for gradient-based optimization of the controller parameters.

Although PILCO uses data very efficiently, its computational demand makes its direct application impractical for many data points or high-dimensional ($\gg$ 20 D) problems, such that we had to make suitable adjustments to apply PILCO to the pixels-to-torques problem. In particular, we performed the following experiments: (1) PILCO applied to 20D PCA features, (2) PILCO applied to 2D features learned by deep auto-encoders, (3) An optimal baseline where we applied PILCO

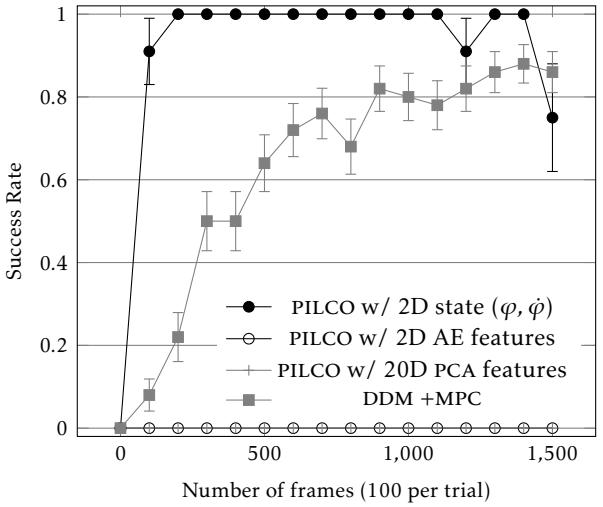


Figure 8: *Average learning success with standard errors. Black filled circle: PILCO ground-truth RL baseline using the true state $(\varphi, \dot{\varphi})$. Black circle: PILCO with learned auto-encoder features from image pixels. Gray cross: PILCO on 20D feature determined by PCA. Gray square: Our proposed MPC solution using the DDM.*

to the standard RL setting with access to the "true" state $(\varphi, \dot{\varphi})$ (Deisenroth et al., 2015).

Figure 8 displays the average success rate of PILCO (including standard error) and our proposed method using deep dynamical models together with a tailored MPC (DDM +MPC). We define "success" if the pendulum's angle is stabilized within 10° around the target state.[2] The baseline (PILCO trained on the ground-truth 2D state $(\varphi, \dot{\varphi})$) is shown in blue and solves the task very quickly. The graph shows that our proposed algorithm (black), which learns torques directly from pixels, is not too far behind the ground-truth RL solution, achieving an almost 90% success rate after 15 trials (1500 image frames). However, PILCO trained on the 2D auto-encoder features (red) and 20D PCA features fail consistently in all experiments We explain PILCO's failure by the fact that we trained the auto-encoder and the transition dynamics in feature space separately. The auto-encoder finds good features that minimize the reconstruction error. However, these features are not good for modeling the dynamic behavior of the system, and lead to bad long-term predictions.

Computation times of PILCO and our method are vastly different: While PILCO spends most time optimizing policy parameters, our model spends most of the time on learning the DDM. Computing the optimal nonparametric MPC policy happens online and does not require significant computational overhead. To put this into context, PILCO required a few days of learning time for 10 trials

[2] Since we consider a continuous setting, we have to define a target *region*.

(in a 20D feature space). In a 2D feature space, running PILCO for 10 trials and 1000 data points requires about 10 hours.

Overall, our DDM +MPC approach to learning closed-loop policies from high-dimensional observations exploits the learned Deep Dynamical Model to learn good policies fairly data efficiently.

5 Conclusion

We have proposed a data-efficient model-based RL algorithm that learns closed-loop policies in continuous state and action spaces directly from pixel information. The key components of our solution are (1) a deep dynamical model (DDM) that is used for long-term predictions in a compact feature space and (2) an MPC controller that uses the predictions of the DDM to determine optimal actions on the fly without the need for value function estimation. For the success of this RL algorithm it is crucial that the DDM learns the feature mapping and the predictive model in feature space jointly to capture dynamic behavior for high-quality long-term predictions. Compared to state-of-the-art RL our algorithm learns fairly quickly, scales to high-dimensional state spaces and facilitates learning from pixels to torques.

Bibliography

E. Abramova, L. Dickens, D. Kuhn, and A. A. Faisal. Hierarchical, heterogeneous control using reinforcement learning. In *European Workshop on Reinforcement Learning (EWRL)*, Edinburgh, UK, June 2012.

C. G. Atkeson and S. Schaal. Learning tasks from a single demonstration. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 1706–1712, Albuquerque, NM, USA, April 1997.

J. A. Bagnell and J. G. Schneider. Autonomous helicopter control using reinforcement learning policy search methods. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 1615–1620, Seoul, South Korea, May 2001.

Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

J. Boedecker, J. T. Springenberg, J. Wülfing, and M. Riedmiller. Approximate real-time optimal control based on sparse Gaussian process models. In *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, Orlando, FL, USA, December 2014.

B. Boots, A. Byravan, and D. Fox. Learning predictive models of a depth camera & manipulator from raw execution traces. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, May 2014.

H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics*, 59(4-5):291–294, 1988.

O. Brock. *Berlin Summit on Robotics: Conference Report*, chapter Is Robotics in Need of a Paradigm Shift?, pages 1–10. 2011.

G. Contardo, L. Denoyer, T. Artieres, and P. Gallinari. Learning states representations in POMDP. *Pre-print arXiv:1312.6042*, 2013.

G. Cuccu, M. Luciw, J. Schmidhuber, and F. Gomez. Intrinsically motivated neuroevolution for vision-based reinforcement learning. In *Proceedings of the International Conference on Development and Learning (ICDL)*, Frankfurt am Main, Germany, August 2011.

M. P. Deisenroth, C. E. Rasmussen, and J. Peters. Gaussian process dynamic programming. *Neurocomputing*, 72(7–9):1508–1524, 2009.

M. P. Deisenroth, D. Fox, and C. E. Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):408–423, 2015.

G. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

J. Koutnik, G. Cuccu, J. Schmidhuber, and F. Gomez. Evolving large-scale neural networks for vision-based reinforcement learning. In *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation (GECCO)*, pages 1061–1068, Amsterdam, The Neatherlands, July 2013.

S. Lange, M. Riedmiller, and A. Voigtländer. Autonomous reinforcement learning on raw visual input data in a real-world application. In *International Joint Conference on Neural Networks (IJCNN)*, Brisbane, Austrailia, 2012.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *Pre-print arXiv:1504.00702*, 2015.

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

D. Q. Mayne. Model predictive control: Recent developments and future promise. *Automatica*, 50(12):2967–2986, 2014.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, and et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2006.

Y. Pan and E. Theodorou. Probabilistic differential dynamic programming. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

S. Schaal. Learning from demonstration. In *Advances in Neural Information Processing Systems (NIPS)*, pages 633–639, Denver, CO, USA, December 1997.

J. Schmidhuber. An on-line algorithm for dynamic reinforcement learning and planning in reactive environments. In *International Joint Conference on Neural Networks (IJCNN)*, pages 253–258, San Diego, CA, USA, June 1990.

J. G. Schneider. Exploiting model uncertainty estimates for safe dynamic control learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1047–1053, 1997.

D. Sha. A new neural networks based adaptive model predictive control for unknown multiple variable non-linear systems. *International Journal of Advanced Mechatronic Systems (IJAMS)*, 1(2):146–155, 2008.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

H. van Hoof, J. Peters, and G. Neumann. Learning of non-parametric control policies with high-dimensional state features. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, San Diego, USA, May 2015.

P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, Helsinki, Fnland, July 2008.

N. Wahlström, T. B. Schön, and M. P. Deisenroth. Learning deep dynamical models from image pixels. In *Proceedings of the 17th IFAC Symposium on System Identification (SYSID)*, Bejing, China, October 2015a.

N. Wahlström, T. B. Schön, and M. P. Deisenroth. From pixels to torques: Policy learning with deep dynamical models. In *Deep Learning Workshop at the International Conference on Machine Learning (ICML)*, Lille, France, July 2015b.

Paper G

Discretizing Stochastic Dynamical Systems Using Lyapunov Equations

Authors: Niklas Wahlström, Patrik Axelsson, and Fredrik Gustafsson

Edited version of the paper:

N. Wahlström, P. Axelsson, and F. Gustafsson. Discretizing stochastic dynamical systems using Lyapunov equations. In *Proceedings of the The 19th World Congress of the International Federation of Automatic Control (IFAC)*, pages 3726–3731, Cape Town, South Africa, August 2014.

Discretizing Stochastic Dynamical Systems Using Lyapunov Equations

Niklas Wahlström, Patrik Axelsson, and Fredrik Gustafsson

Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
{nikwa,axelsson,fredrik}@isy.liu.se

Abstract

Stochastic dynamical systems are fundamental in state estimation, system identification and control. System models are often provided in continuous time, while a major part of the applied theory is developed for discrete-time systems. Discretization of continuous-time models is hence fundamental. We present a novel algorithm using a combination of Lyapunov equations and analytical solutions, enabling efficient implementation in software. The proposed method circumvents numerical problems exhibited by standard algorithms in the literature. Both theoretical and simulation results are provided.

1 Introduction

Dynamical processes in engineering and physics have for a long time successfully been modeled with continuous-time differential equations. In order to account for uncertainties, these equations are usually driven by an unknown stochastic process called process noise. This noise is ideally modeled as completely "white" in order to obtain the Markov property, which is required in recursive Bayesian inference, such as Kalman filtering. However, in order to implement such filtering, the continuous-time model has to be discretized. Such discretization includes solving an integral involving the matrix exponential on the form

$$Q = \int_0^T e^{A\tau} S e^{A^\mathsf{T}\tau} d\tau, \tag{1}$$

where $A, S, Q \in \mathbb{R}^{n\times n}$.

We propose an algorithm for solving (1) by decomposing the problem into subproblems and then solve these subproblems either analytically or using a combination of Lyapunov and Sylvester equations.

In many practical applications the discrete-time process noise covariance is modeled and tuned directly, rather than discretized from its continuous-time

counterpart. However, in certain scenarios the dependency between the discrete-time process noise covariance and the sampling time is important. If the filtering should work on different devices with different sampling frequencies, this dependency should be properly modeled to guarantee the same dynamical behavior of the filter. Further, in data with non-equidistant sampling the process noise covariance has to be rescaled at each time instant. This is often the case in Gaussian process regression which can be described with a state-space model and solved using Kalman filtering (Särkkä et al., 2013).

In the literature there exist different algorithms for computing the integral (1). The probably most well-cited one was presented by Van Loan (1978), which involves computing the matrix exponential for an augmented $2n \times 2n$ matrix followed by a matrix multiplication of two resulting submatrices. This method does not require any assumption on the model, however the resulting matrix becomes ill-conditioned if the sampling time is large or if the poles of the system are fast. For certain models, (1) can be solved analytically. Rome (1969) presented a direct solution under the assumption that A is diagonalizable. The method requires an eigenvalue decomposition which is not always numerical stable (Higham, 2008) and not all matrices are diagonalizable. Finally, the integral can always be solved numerically using the trapezoidal or the rectangular method.

In this work we present an alternative method for solving (1). This method is based on a Lyapunov equation which characterizes the solution of (1). However, since Lyapunov equations cannot be solved if the system contains integrators (Antoulas, 2005), the problem is decomposed into subproblems where the integrators are treated separately. As will be explained, one set of subproblems cannot be solved using Lyapunov equations, but they do have an analytical solution of (1). Conversely, the remaining set of subproblems do not have a closed form solution of (1), but then the method with Lyapunov equations works fine. The algorithm involves computing the matrix exponential of the $n \times n$ system matrix rather than an augmented $2n \times 2n$ matrix as required by the solution by Van Loan. Furthermore, the proposed algorithm circumvents some numerical problems in the method proposed by Van Loan. Our theoretical algebraic contributions include:

- A Lemma describing the relation between (1) and the aforementioned Lyapunov equation, see Lemma 2.
- A novel extension of this solution which also handles integrators, see Section 4.
- A complete algorithm which solves (1) with complementing numerical properties compared to existing solutions, see Algorithm 3.

The outline of the paper is as follows. In Section 2 the mathematical models are presented and the importance of the discretization method in use is motivated. In Section 3 the discretization using Lyapunov equations is presented together with the main theoretical contributions of the paper. In Section 4 the solutions from the previous two sections will be combined to solve for systems with

integrators. In Section 5 a numerical evaluation is performed and in Section 6 the conclusions are summarized and future directions pointed out.

2 Mathematical Preliminaries

Consider the following Itô stochastic differential equation

$$d\mathbf{x}(t) = A\mathbf{x}(t)dt + d\boldsymbol{\beta}(t), \tag{2a}$$

where $\boldsymbol{\beta}(t)$ is a Brownian motion with

$$\mathbb{E}[d\boldsymbol{\beta}(t)d\boldsymbol{\beta}(t)^{\mathsf{T}}] = S\,dt. \tag{2b}$$

The model (2a) is formally equivalent to the stochastic differential equation

$$\frac{d\mathbf{x}(t)}{dt} = A\mathbf{x}(t) + \mathbf{w}(t), \tag{3a}$$

where $\mathbf{w}(t)$ is a zero-mean white Gaussian process with

$$\mathbb{E}[\mathbf{w}(t)\mathbf{w}(\tau)^{\mathsf{T}}] = S\delta(t-\tau). \tag{3b}$$

Since $\mathbf{w}(t)$ is not square Riemann integrable, the model (3) does not have any mathematical meaning (Jazwinski, 1970). However, we can still intuitively think of it as a stochastic differential equation driven by white noise.

It is important to note that this is just a model of the physical process and cannot be found in reality. For example, white noise has a flat power spectral density requiring infinite power, which is not physically realizable. Nevertheless, using this continuous-time model will lead to sound properties for the equivalent discrete-time model as will be explained later.

By integrating (2a) over the time interval $[t_k, t_{k+1}]$ we can find its discrete-time equivalence as

$$\mathbf{x}(t_{k+1}) = \underbrace{e^{AT_k}}_{F_{T_k}} \underbrace{\mathbf{x}(t_k)}_{\mathbf{x}_k} + \underbrace{\int_{t_k}^{t_{k+1}} e^{A(t_{k+1}-\tau)} d\boldsymbol{\beta}(\tau)}_{\mathbf{w}_k}, \tag{4}$$

where $T_k = t_{k+1} - t_k$ is the sampling time. This can be stated as a discrete-time stochastic difference equation

$$\mathbf{x}_{k+1} = F_{T_k}\mathbf{x}_k + \mathbf{w}_k. \tag{5a}$$

By following for example Jazwinski (1970), the noise $\mathbf{w}_k$ will be zero-mean, white Gaussian

$$\mathbb{E}[\mathbf{w}_k\mathbf{w}_l^{\mathsf{T}}] = Q_{T_k}\delta_{kl}, \tag{5b}$$

where δ_{kl} is the Kronecker delta function and

$$Q_{T_k} = \int_0^{T_k} e^{A\tau} S e^{A^\mathsf{T}\tau} d\tau, \tag{6a}$$

which together with the discrete-time system matrix

$$F_{T_k} = e^{AT_k} \tag{6b}$$

completes the discretization procedure.

The integral expression (6a) can be found in multiple textbooks on Kalman filtering (e.g. Bar-Shalom et al. (2001); Grewal and Andrews (2008)) for modeling discrete-time dynamical processes. Nevertheless, the discretization of continuous-time differential equations for filtering applications is often misused. For example, the noise $\mathbf{w}(t)$ is commonly assumed to be constant during each sampling interval leading to the following discrete-time noise covariance

$$\bar{Q}^{\mathrm{A}}_{T_k} = \frac{1}{T_k} \underbrace{\left(\int_0^{T_k} e^{A\tau} d\tau \right)}_{\bar{G}_{T_k}} S \underbrace{\left(\int_0^{T_k} e^{A^\mathsf{T}\tau} d\tau \right)}_{\bar{G}^\mathsf{T}_{T_k}}, \tag{7a}$$

or just rescaling the continuous-time noise covariance with the sampling time

$$\bar{Q}^{\mathrm{B}}_{T_k} = T_k S. \tag{7b}$$

In contrast to the discretization in (6), the assumptions in (7) lead to a dynamical description of the process which depends on the sampling intervals, whereas the actual physical process do not. This can be seen by the property derived in the following example.

Example 1

Consider the three time instances t_1, t_2 and t_3. We then have

$$\begin{aligned}
&\mathrm{Cov}\Big[\mathbf{x}(t_3)\Big|\mathbf{x}(t_1)\Big] && \text{(8a)}\\
&\quad = \mathrm{Cov}\Big[F_{t_3-t_2}\mathbf{x}(t_2) + \mathbf{w}_2\Big|\mathbf{x}(t_1)\Big] && \text{(8b)}\\
&\quad = \mathrm{Cov}\Big[F_{t_3-t_2}\Big(F_{t_2-t_1}\mathbf{x}(t_1) + \mathbf{w}_1\Big) + \mathbf{w}_2\Big|\mathbf{x}(t_1)\Big] && \text{(8c)}\\
&\quad = \mathrm{Cov}\Big[F_{T_2}\mathbf{w}_1 + \mathbf{w}_2\Big|\mathbf{x}(t_1)\Big] && \text{(8d)}\\
&\quad = F_{T_2} Q_{T_1} F^\mathsf{T}_{T_2} + Q_{T_2}. && \text{(8e)}
\end{aligned}$$

We could also use only one time interval and go from t_1 directly to t_3 with the sampling time $t_3 - t_1 = T_1 + T_2$, which gives

$$\begin{aligned}
&\mathrm{Cov}\Big[\mathbf{x}(t_3)\Big|\mathbf{x}(t_1)\Big] && \text{(9a)}\\
&\quad = \mathrm{Cov}\Big[F_{t_3-t_1}\mathbf{x}(t_1) + \mathbf{w}_1\Big|\mathbf{x}(t_1)\Big] && \text{(9b)}\\
&\quad = \mathrm{Cov}\Big[\mathbf{w}_1\Big|\mathbf{x}(t_1)\Big] = Q_{t_3-t_1} = Q_{T_1+T_2}. && \text{(9c)}
\end{aligned}$$

This gives the relation

$$Q_{T_1+T_2} = F_{T_2} Q_{T_1} F_{T_2}^{\mathsf{T}} + Q_{T_2}. \tag{10}$$

Indeed, this property is fulfilled for the discretization presented in (6).

Lemma 1. *If F_{T_k} and Q_{T_k} are computed as described in* (6)*, then*

$$Q_{T_1+T_2} = F_{T_2} Q_{T_1} F_{T_2}^{\mathsf{T}} + Q_{T_2}.$$

Proof:

$$\begin{aligned}
Q_{T_1+T_2} &= \int_0^{T_1+T_2} e^{A\tau} S e^{A^{\mathsf{T}}\tau} d\tau && \text{(11a)}\\
&= \int_0^{T_2} e^{A\tau} S e^{A^{\mathsf{T}}\tau} d\tau + \int_{T_2}^{T_1+T_2} e^{A\tau} S e^{A^{\mathsf{T}}\tau} d\tau && \text{(11b)}\\
&= \underbrace{\int_0^{T_2} e^{A\tau} S e^{A^{\mathsf{T}}\tau} d\tau}_{Q_2} + \underbrace{e^{AT_2}}_{F_{T_2}} \underbrace{\int_0^{T_1} e^{A^{\mathsf{T}}\tau} S e^{A^{\mathsf{T}}\tau} d\tau}_{Q_{T_1}} \underbrace{e^{A^{\mathsf{T}}T_2}}_{F_{T_2}^{\mathsf{T}}} && \text{(11c)}\\
&= Q_{T_2} + F_{T_2} Q_{T_1} F_{T_2}^{\mathsf{T}}. && \text{(11d)}
\end{aligned}$$

□

With similar calculations we can easily derive the equivalent results for the covariance matrices in (7) and conclude that they do not share this property since

$$\begin{aligned}
\bar{Q}^{\mathrm{A}}_{T_1+T_2} &= \bar{Q}^{\mathrm{A}}_{T_2} + F_{T_2}\bar{Q}^{\mathrm{A}}_{T_1}F_{T_2}^{\mathsf{T}} + F_{T_2}\bar{G}_{T_1} S \bar{G}_{T_2}^{\mathsf{T}} + \bar{G}_{T_2} S \bar{G}_{T_1}^{\mathsf{T}} F_{T_2}^{\mathsf{T}}\\
&\neq \bar{Q}^{\mathrm{A}}_{T_2} + F_{T_2}\bar{Q}^{\mathrm{A}}_{T_1}F_{T_2}^{\mathsf{T}}, && \text{(12a)}\\
\bar{Q}^{\mathrm{B}}_{T_1+T_2} &= \bar{Q}^{\mathrm{B}}_{T_2} + \bar{Q}^{\mathrm{B}}_{T_1}\\
&\neq \bar{Q}^{\mathrm{B}}_{T_2} + F_{T_2}\bar{Q}^{\mathrm{B}}_{T_1}F_{T_2}^{\mathsf{T}}. && \text{(12b)}
\end{aligned}$$

Hence by assuming that the underlying continuous-time model is driven by a continuous-time white process the corresponding discrete-time model has the property that the dynamical description does not depend on the sampling intervals, in contrast to other common discretization procedures. We can therefore see (5) and (6) as algebraic relations between A, S, T_k, F_{T_k} and Q_{T_k} fulfilling the property in (10) without deriving it from its continuous-time counterpart.

The main advantage with the alternative expressions in (7) in comparison to (6a) is their ease of calculation (especially true for (7b)). The remaining part of this work will therefore describe how the integral (6a) can be solved in an efficient manner with good numerical properties.

3 Discretization Using Lyapunov Equations

A method for computing the integral (6a) will now be presented. The method will be proposed by requiring the system to be asymptotically stable. Later in this section we will prove that this requirements actually can be relaxed.

3.1 Proposal of Solution

If the system is asymptotically stable, i.e. if all eigenvalues of A have negative real part, a stationary covariance will exist and we denote it as

$$\mathrm{Cov}[\mathbf{x}(t)] = \mathrm{Cov}[\mathbf{x}_k] = P. \tag{13}$$

This covariance satisfies the following two Lyapunov equations for the continuous-time model (2a) and the discrete-time model (5a), respectively

$$0 = AP + PA^{\mathsf{T}} + S, \tag{14a}$$

$$P = F_{T_k} P F_{T_k}^{\mathsf{T}} + Q_{T_k}. \tag{14b}$$

which gives a structured way of computing Q_{T_k}, as presented in Algorithm 1.

Algorithm 1 Solution using Lyapunov equation for P

The matrices A and S and the scalar T_k are given. The matrices F_{T_k} and Q_{T_k} in (6) can then be computed as

$$F_{T_k} = e^{AT_k}, \tag{15a}$$

$$Q_{T_k} = P - F_{T_k} P F_{T_k}^{\mathsf{T}}, \tag{15b}$$

where P is the solution to the Lyapunov equation

$$AP + PA^{\mathsf{T}} = -S. \tag{15c}$$

This algorithm can also be reformulated such that we do not need to compute P in an intermediate step. By multiplying (15b) with A from the left and with A^{T}

from the right, respectively, we get

$$AQ_{T_k} = AP - F_{T_k} APF_{T_k}^{\mathsf{T}}, \tag{16a}$$

$$Q_{T_k} A^{\mathsf{T}} = PA^{\mathsf{T}} - F_{T_k} PA^{\mathsf{T}} F_{T_k}^{\mathsf{T}}, \tag{16b}$$

where the fact that F_{T_k} and A commute has been used since

$$AF_{T_k} = A(I + A + \frac{1}{2}A^2 \dots) = (A + A^2 + \frac{1}{2}A^3 \dots) \tag{17a}$$

$$= (I + A + \frac{1}{2}A^2 \dots)A = F_{T_k} A. \tag{17b}$$

By adding (16a) and (16b), we get

$$AQ_{T_k} + Q_{T_k} A^{\mathsf{T}} = AP - F_{T_k} APF_{T_k}^{\mathsf{T}} + PA^{\mathsf{T}} - F_{T_k} PA^{\mathsf{T}} F_{T_k}^{\mathsf{T}} \tag{18a}$$

$$= \underbrace{AP + PA^{\mathsf{T}}}_{-S} - F_{T_k} (\underbrace{AP + PA^{\mathsf{T}}}_{-S}) F_{T_k}^{\mathsf{T}} \tag{18b}$$

$$= -S + F_{T_k} SF_{T_k}^{\mathsf{T}}. \tag{18c}$$

This gives the following algorithm as presented in Algorithm 2. This algorithm

Algorithm 2 Solution using Lyapunov equation for Q_{T_k}

The matrices A and S and the scalar T_k are given. The matrices F_{T_k} and Q_{T_k} in (6) can then be computed as

$$F_{T_k} = e^{AT_k} \tag{19a}$$

and Q_{T_k} is the solution to the Lyapunov equation

$$AQ_{T_k} + Q_{T_k} A^{\mathsf{T}} = -V_{T_k}, \tag{19b}$$

where

$$V_{T_k} = S - F_{T_k} SF_{T_k}^{\mathsf{T}}. \tag{19c}$$

is similar to the solution presented by Axelsson and Gustafsson (2012) derived from a continuous-time differential Lyapunov equation.

From here on we will proceed with Algorithm 2. However, all results (including the final algorithm) can be reformulated to suit Algorithm 1 as well.

3.2 Theoretical Result

It can now be proven that Algorithm 2 (and consequently also Algorithm 1) gives a solution to (6), provided that the solution of the Lyapunov equation exists and is unique.

Lemma 2. *The solution to the integral*

$$Q = \int_0^T e^{A\tau} S e^{B\tau} d\tau \tag{20a}$$

satisfies the Sylvester equation

$$AQ + QB = -S + e^{AT} S e^{BT}. \tag{20b}$$

Proof: Start with (20b) and replace Q with the integral (20a). This gives

$$AQ + QB = \int_0^T A e^{A\tau} S e^{B\tau} d\tau + \int_0^T e^{A\tau} S e^{B\tau} B d\tau \tag{21a}$$

$$= \int_0^T \frac{d}{d\tau}[e^{A\tau} S e^{B\tau}] d\tau \tag{21b}$$

$$= e^{A\tau} S e^{B\tau}\Big|_0^T = e^{AT} S e^{BT} - S. \tag{21c}$$

□

Remark 1. A similar result was presented by Gawronski (2004) in the context of time-limited grammians. However, that result requires $B = A^\mathsf{T}$ and that all eigenvalues of A should have negative real part.

Remark 2. Note that Lemma 2 does not require anything about the matrices A and B. In particular, they do not need to be stable as assumed in (13) and (14). Indeed, the requirements for the Lyapunov equation (19b) to have a unique solution are milder. This is answered by the following proposition, which is given for the more general Sylvester equation.

Proposition 1. *The Sylvester equation*

$$AP + PB = C \tag{22a}$$

has a unique solution P if and only if

$$\lambda_i(A) + \lambda_j(B) \neq 0 \quad \forall i, j. \tag{22b}$$

For proof, see for example Antoulas (2005).

For the case where $B = A^\mathsf{T}$ and with the requirement that A is stable, the condition (22b) is always fulfilled. By using that observation together with Lemma 2 where $T \to \infty$, we get the following well known results relating the controllability grammian to a Lyapunov equation, which can be found in most textbooks on linear systems, e.g. Rugh (1996).

Corollary 1. *If all eigenvalues of A have negative real parts, then for each symmetric matrix S there exists a unique solution of*

$$AQ + QA^{\mathsf{T}} = -S \tag{23a}$$

given by

$$Q = \int_0^\infty e^{A\tau} S e^{A^{\mathsf{T}}\tau} d\tau. \tag{23b}$$

According to Proposition 1 the integral (6a) cannot be computed using the Lyapunov equation (19b) if A and $-A$ have any common eigenvalues. This is especially the case if the system has integrators, which indeed is common in models intended for Kalman filtering. In the next section we will therefore present a solution which handles such systems as well. With this extension almost all systems of interest will be covered, except for the systems which have at least one pair of non-zero poles mirrored in the imaginary axis.

This extension will be performed by decomposing the problem into subproblems where some of these subproblems still can be solved using parts of the Lyapunov equation (19b), whereas the remaining subproblem can be solved analytically using the integral (6a).

4 Solution for Systems with Integrators

Consider the case when A is block triangular consisting of three blocks

$$A = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \tag{24}$$

where

$$\lambda_i(A_{11}) + \lambda_j(A_{11}) \neq 0 \quad \forall i, j, \tag{25a}$$

$$\lambda_i(A_{11}) + \lambda_j(A_{22}) \neq 0 \quad \forall i, j, \tag{25b}$$

$$\lambda_j(A_{22}) = 0 \quad \forall i, j. \tag{25c}$$

In this manner we have partitioned A such that all zero eigenvalues have been placed in A_{22} and all remaining non-zero eigenvalues in A_{11}. Many systems do have such block triangular structure, for example if an observer canonical form has been used, see Example 3. If the system does not have that form, an orthogonal transformation can be applied. This transformation can be computed using Schur decomposition and reordering of the eigenvalues (Golub and Van Loan, 1996). This will also affect the covariance matrix S as well as V_{T_k} by considering this transformation as a state transformation, see Appendix A.

4.1 Solution using Lyapunov and Sylvester Equations

According to Lemma 2, the solution of the integral (6a) for the block triangular matrix (24) shall obey the following Lyapunov equation

$$\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12}^\mathsf{T} & Q_{22} \end{bmatrix} + \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12}^\mathsf{T} & Q_{22} \end{bmatrix} \begin{bmatrix} A_{11}^\mathsf{T} & 0 \\ A_{12}^\mathsf{T} & A_{22}^\mathsf{T} \end{bmatrix} = - \begin{bmatrix} V_{11} & V_{12} \\ V_{12}^\mathsf{T} & V_{22} \end{bmatrix}, \tag{26}$$

where Q_{T_k} and V_{T_k} have been partitioned in a similar manner as A. Note that the subscript T_k has been omitted from the submatrices in order to make the notation less cluttered. This gives the following set of Lyapunov and Sylvester equations

$$A_{11}Q_{11} + Q_{11}A_{11}^\mathsf{T} = -V_{11} - A_{12}Q_{12}^\mathsf{T} - Q_{12}A_{12}^\mathsf{T}, \tag{27a}$$

$$A_{11}Q_{12} + Q_{12}A_{22}^\mathsf{T} = -V_{12} - A_{12}Q_{22}, \tag{27b}$$

$$A_{22}Q_{12}^\mathsf{T} + Q_{12}^\mathsf{T}A_{11}^\mathsf{T} = -V_{12}^\mathsf{T} - Q_{22}A_{12}^\mathsf{T}, \tag{27c}$$

$$A_{22}Q_{22} + Q_{22}A_{22}^\mathsf{T} = -V_{22}. \tag{27d}$$

Based on the requirements in (25a) and (25b), Proposition 1 guarantees that Q_{11} and Q_{12} can be solved uniquely using (27a) and (27b) if Q_{22} is known. In contrast, (27d) does not have a unique solution for Q_{22}. Instead, Q_{22} can be solved analytically using the integral (6a). Note that (27c) is just a transposed version of (27b) and does not bring any extra information.

4.2 Analytical Solution for the Nilpotent Part

Due to the block triangular structure of A, the submatrix Q_{22} will only depend on A_{22} and S_{22} via a similar expression as in (6a). By starting from (6a) we have

$$Q_{22} = \begin{bmatrix} 0 & I \end{bmatrix} Q \begin{bmatrix} 0 \\ I \end{bmatrix} \tag{28a}$$

$$= \begin{bmatrix} 0 & I \end{bmatrix} \int_0^{T_k} e^{A\tau} S e^{A^\mathsf{T}\tau} d\tau \begin{bmatrix} 0 \\ I \end{bmatrix} \tag{28b}$$

$$= \int_0^{T_k} \begin{bmatrix} 0 & e^{A_{22}\tau} \end{bmatrix} S \begin{bmatrix} 0 \\ e^{A_{22}^\mathsf{T}\tau} \end{bmatrix} d\tau \tag{28c}$$

$$= \int_0^{T_k} e^{A_{22}\tau} S_{22} e^{A_{22}^\mathsf{T}\tau} d\tau. \tag{28d}$$

Further, since all eigenvalues of A_{22} are zero, the submatrix A_{22} will also be nilpotent (Lancaster and Tismenetsky, 1985) leading to

$$e^{A_{22}\tau} = \sum_{i=0}^{p-1} A_{22}^i \frac{\tau^i}{i!}, \tag{29}$$

where p is the dimension of A_{22}, i.e. the number of integrators in the system. Expression (28d) can then be computed analytically as

$$Q_{22} = \int_0^{T_k} \left(\sum_{i=0}^{p-1} \frac{1}{i!} A_{22}^i \tau^i \right) S_{22} \left(\sum_{j=0}^{p-1} \frac{1}{j!} A_{22}^j{}^{\mathsf{T}} \tau^j \right) d\tau \tag{30a}$$

$$= \sum_{i=0}^{p-1} \sum_{j=0}^{p-1} \frac{1}{i!j!} A_{22}^i S_{22} A_{22}^j{}^{\mathsf{T}} \int_0^{T_k} \tau^{i+j} d\tau \tag{30b}$$

$$= \sum_{i=0}^{p-1} \sum_{j=0}^{p-1} \frac{T_k^{i+j+1}}{i!j!(i+j+1)} A_{22}^i S_{22} A_{22}^j{}^{\mathsf{T}}. \tag{30c}$$

This is illustrated with the following example.

Example 2

Consider a constant velocity model, which formally can be described on the form

$$\dot{\mathbf{x}}(t) = \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}}_{A} \mathbf{x}(t) + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{B} q(t), \quad \mathbb{E}[q(t)q(\tau)] = \delta(t-\tau). \tag{31}$$

This system has only zero eigenvalues which gives

$$A = A_{22} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \tag{32a}$$

$$S = S_{22} = \mathbb{E}[Bq(Bq)^{\mathsf{T}}] = \begin{bmatrix} 0 \\ 1 \end{bmatrix} 1 \begin{bmatrix} 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}. \tag{32b}$$

By using this in (28) we get

$$Q_{T_k} = S T_k + S A_{22}^{\mathsf{T}} \frac{T_k^2}{2} + A_{22} S \frac{T_k^2}{2} + A_{22} S A_{22}^{\mathsf{T}} \frac{T_k^3}{3} \tag{32c}$$

$$= \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} T_k + \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \frac{T_k^2}{2} + \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \frac{T_k^2}{2} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \frac{T_k^3}{3} \tag{32d}$$

$$= \begin{bmatrix} \frac{T_k^3}{3} & \frac{T_k^2}{2} \\ \frac{T_k^2}{2} & T_k \end{bmatrix}, \tag{32e}$$

which is the same result as given by Grewal and Andrews (2008), but derived in a different way.

Algorithm 3 Proposed algorithm (for systems with arbitrary number of integrators)

The matrices A and S and the scalar T_k are given. The matrices F_{T_k} and Q_{T_k} in (6) can then be computed as

1. Transform A and S to $\tilde{A}$ and $\tilde{S}$ such that $\tilde{A}$ becomes block triangular

$$U^{-1}AU = \tilde{A} = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ 0 & \tilde{A}_{22} \end{bmatrix}, \quad U^{-1}SU^{-\mathsf{T}} = \tilde{S} = \begin{bmatrix} \tilde{S}_{11} & \tilde{S}_{12} \\ \tilde{S}_{12}^{\mathsf{T}} & \tilde{S}_{22} \end{bmatrix},$$

   and with all integrators collected in $\tilde{A}_{22}$. This can be done with an orthogonal transformation computed using Schur decomposition and reordering of the eigenvalues.

2. Compute $\tilde{F}_{T_k} = e^{\tilde{A}T_k}$.

3. Compute $\tilde{V}_{T_k} = \tilde{S} - \tilde{F}_{T_k}\tilde{S}\tilde{F}_{T_k}^{\mathsf{T}}$.

4. Compute

$$\tilde{Q}_{T_k} = \begin{bmatrix} \tilde{Q}_{11} & \tilde{Q}_{12} \\ \tilde{Q}_{12}^{\mathsf{T}} & \tilde{Q}_{22} \end{bmatrix},$$

   using the following steps:

   (a) Compute $\tilde{Q}_{22}$ by evaluating

$$\tilde{Q}_{22} = \sum_{i=0}^{p-1}\sum_{j=0}^{p-1} \frac{T_k^{i+j+1}}{i!j!(i+j+1)} \tilde{A}_{22}^i \tilde{S}_{22} (\tilde{A}_{22}^i)^{\mathsf{T}}, \tag{33a}$$

   where p is the number of integrators.

   (b) Compute $\tilde{Q}_{12}$ by solving the Sylvester equation

$$\tilde{A}_{11}\tilde{Q}_{12} + \tilde{Q}_{12}\tilde{A}_{22}^{\mathsf{T}} = -\tilde{V}_{12} - \tilde{A}_{12}\tilde{Q}_{22}. \tag{33b}$$

   (c) Compute $\tilde{Q}_{11}$ by solving the Lyapunov equation

$$\tilde{A}_{11}\tilde{Q}_{11} + \tilde{Q}_{11}\tilde{A}_{11}^{\mathsf{T}} = -\tilde{V}_{11} - \tilde{A}_{12}\tilde{Q}_{12}^{\mathsf{T}} - \tilde{Q}_{12}\tilde{A}_{12}^{\mathsf{T}}. \tag{33c}$$

5. Transform $\tilde{F}_{T_k}$ and $\tilde{Q}_{T_k}$ back to F_{T_k} and Q_{T_k}

$$F_{T_k} = U\tilde{F}_{T_k}U^{-1}, \tag{34a}$$
$$Q_{T_k} = U\tilde{Q}_{T_k}U^{\mathsf{T}}. \tag{34b}$$

4.3 General Algorithm

Based on the results in the last section, we can now propose an algorithm for computing the integral (6a), also in the case where A consists of integrators, see Algorithm 3.

Remark 3. If A does not have any integrators, Algorithm 3 will collapse to the simpler version in Algorithm 2.

Remark 4. In theory, $U^{-1} = U^{\mathsf{T}}$ since U is orthogonal. However, numerical algorithms for computing the Schur decomposition do not make U completely orthogonal. From a numerical point of view it is therefor a benefit to distinguish between U^{-1} and U^{T}.

Remark 5. If $\tilde{A}_{12} = 0$ the coupling in (33b) and (33c) via $\tilde{Q}_{12}$ and $\tilde{Q}_{22}$ will disappear and they can be solved independently from each other. If this is desired, the transformation in Step 1 can be extended to eliminate $\tilde{A}_{12}$ by solving an addition Sylvester equation (Bavely and Stewart, 1979). However, such transformation is no longer orthogonal and can be arbitrary ill-conditioned if the non-zero eigenvalues are close to zero.

Remark 6. If the system already has a block triangular structure, Step 1 and Step 5 in Algorithm 3 can be omitted. This is the case for the observer canonical form as seen in the following short example.

Example 3

Consider a SISO-system of order $n = m+p$ with m non-zero poles and p additional integrators described with a transfer function

$$G(s) = \frac{b_1 s^{m-1} + b_2 s^{m-2} + \cdots + b_{m-1} s + b_m}{s^n + a_1 s^{m-1} + \cdots + a_{m-1} s + a_m} \cdot \frac{1}{s^p}. \tag{35}$$

This system can be described with the observer canonical form (Glad and Ljung, 2000)

$$\dot{\mathbf{x}} = \left[\begin{array}{cccc|cccc} -a_1 & 1 & \dots & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ -a_{m-1} & 0 & \dots & 1 & 0 & 0 & \dots & 0 \\ -a_m & 0 & \dots & 0 & 1 & 0 & \dots & 0 \\ \hline 0 & 0 & \dots & 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 1 \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 \end{array}\right] \mathbf{x} + \left[\begin{array}{c} 0 \\ \vdots \\ 0 \\ b_1 \\ \vdots \\ \vdots \\ b_m \end{array}\right] \mathbf{w}, \tag{36a}$$

$$\mathbf{y} = \left[\begin{array}{cccc} 1 & 0 & \dots & 0 \end{array}\right] \mathbf{x}, \tag{36b}$$

which can be written more compactly as

$$\dot{\mathbf{x}} = \left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline 0 & A_{22} \end{array}\right] \mathbf{x} + B\mathbf{w}, \tag{37a}$$

$$\mathbf{y} = \left[\begin{array}{cccc} 1 & 0 & \dots & 0 \end{array}\right] \mathbf{x}. \tag{37b}$$

This system has by construction the desired block triangular structure.

5 Numerical Evaluation

In this section the numerical properties of the proposed solution will be compared with a standard solution presented by Van Loan (1978) given in Algorithm 4.

Algorithm 4 Van Loan's method

The matrices A and S and the scalar T_k are given. The matrices F_{T_k} and Q_{T_k} in (6) can then be computed as

1. Compute the matrix exponential of an augmented $2n \times 2n$ matrix

$$\begin{bmatrix} M_{11} & M_{12} \\ 0 & M_{22} \end{bmatrix} = e^{HT_k}, \text{where} \quad H = \begin{bmatrix} A & S \\ 0 & -A^\mathsf{T} \end{bmatrix}. \tag{38a}$$

2. The matrices F_{T_k} and Q_{T_k} are given as

$$F_{T_k} = M_{11}, \qquad Q_{T_k} = M_{12}M_{11}^\mathsf{T}. \tag{38b}$$

5.1 Implementation Aspects

In both methods MATLAB's built-in function `expm` has been used for computing the matrix exponential. In Step 1 of Algorithm 3 the functions `schur` and `ordschur` have been used for computing the Schur decomposition and the reordering of the eigenvalues. Finally, the Lyapunov and Sylvester equations have been solved using `lyap`.

5.2 Simulation Results

In total 100 systems of order $n = 6$ with $m = 4$ stable poles and $p = 2$ additional integrators are randomly generated. Each system is normalized such that the fastest pole is at distance 1 from the imaginary axis, i.e. $\max(|\mathrm{Re}(\lambda_i)|) = 1$. An estimate $\hat{Q}_{T_k}$ is computed using both Algorithm 3 and Algorithm 4 with single precision for different values of the sampling time T_k. Finally, the error

$$\varepsilon = \|\hat{Q}_{T_k} - Q_{T_k}\|_2 / \|Q_{T_k}\|_2$$

is evaluated, where Q_{T_k} is computed using numerical integration of (6a) with double precision, here considered as the true value. The result is presented in Figure 1.

According to the result the proposed method outperforms the standard method for large T_k. The reason will become clear if we investigate the two methods further. In Algorithm 4, both AT_k and $-A^\mathsf{T}T_k$ are present in the augmented matrix HT_k and the task to compute its matrix exponential (38a) will become

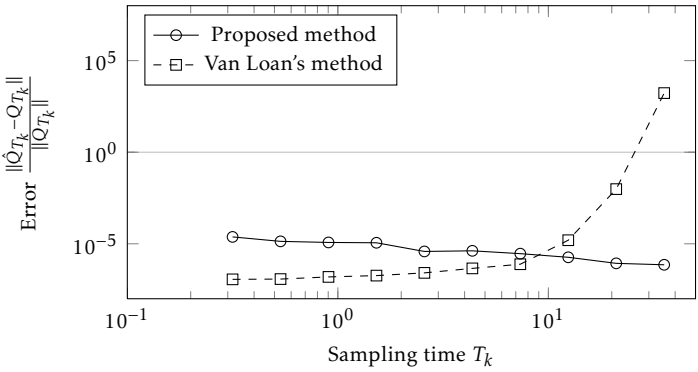


Figure 1: *The performance of the proposed method (Algorithm 3) and Van Loan's method (Algorithm 4).*

ill-conditioned if T_k or $\max(|\mathrm{Re}(\lambda_i)|)$ is large. In fact, the error will grow exponentially with T_k, or the magnitude of work will grow linearly with T_k to keep a certain tolerance (Van Loan, 1978). This issue is not present in the proposed method, which can be seen in its simplified version in Algorithm 1. If T_k is large we have $F_{T_k} = e^{AT_k} \approx 0$ and Q_{T_k} will approach the stationary covariance P according to (15b).

However, for fast sampling the proposed method performers slightly worse. This is especially the case if the system has integrators as well as non-zero poles close to the origin leading to that the Sylvester equation (33b) will become ill-conditioned. Consequently, van Loan's method has numerical issues when the fastest pole is fast and the sampling is slow, whereas the proposed method has nummerical problems when the slowest (non-zero) pole is slow and the sampling is fast. The proposed method also has computational complexity advantages since it only needs to compute the matrix exponential of an $n \times n$ matrix rather than of an augmented $2n \times 2n$ matrix as required by van Loan's method.

6 Conclusions and Future Work

An algorithm for computing an integral involving the matrix exponential common in optimal sampling was proposed. The algorithm is based on a Lyapunov equation and is justified with a novel lemma. An extension to systems with integrators was presented. Numerical evaluations showed that the proposed algorithm has advantageous numerical properties for slow sampling and fast dynamics in comparison with a standard method in the literature.

Further work includes extending the algorithm further to handle arbitrary matrices, i.e. also matrices with non-zero eigenvalues mirrored in the imaginary axis. Also the numerical properties should be analyzed further and strategies for improving the numerical properties for slow poles should be investigated.

Appendix

A State Transformation

Consider the following state transformation

$$\mathbf{x} = U\tilde{\mathbf{x}}. \tag{39}$$

By applying (39) to the dynamical equation (3a) we get

$$\dot{\mathbf{x}} = A\mathbf{x} + \mathbf{w} \quad \Rightarrow \tag{40a}$$
$$U\dot{\tilde{\mathbf{x}}} = AU\tilde{\mathbf{x}} + \mathbf{w} \quad \Rightarrow \tag{40b}$$
$$\dot{\tilde{\mathbf{x}}} = U^{-1}AU\tilde{\mathbf{x}} + U^{-1}\mathbf{w} \quad \Rightarrow \tag{40c}$$
$$\dot{\tilde{\mathbf{x}}} = \tilde{A}\tilde{\mathbf{x}} + \tilde{\mathbf{w}}. \tag{40d}$$

which gives the following transformation of A, S and V

$$\tilde{A} = U^{-1}AU, \tag{41a}$$
$$\begin{aligned}\tilde{S} &= \mathbb{E}[\tilde{\mathbf{w}}\tilde{\mathbf{w}}^\mathsf{T}] = \mathbb{E}[U^{-1}\mathbf{w}(U^{-1}\mathbf{w})^\mathsf{T}] = U^{-1}\mathbb{E}[\mathbf{w}\mathbf{w}^\mathsf{T}]U^{-\mathsf{T}} \\ &= U^{-1}SU^{-\mathsf{T}}.\end{aligned} \tag{41b}$$

These matrices will then be used to compute $\tilde{F}_{T_k}$ and $\tilde{Q}_{T_k}$ by following Step 2-4 in Algorithm 3. We then have

$$\tilde{\mathbf{x}}_{k+1} = \tilde{F}_{T_k}\tilde{\mathbf{x}}_k + \tilde{\mathbf{w}}_k \quad \Rightarrow \tag{42a}$$
$$U^{-1}\mathbf{x}_{k+1} = \tilde{F}_{T_k}U^{-1}\mathbf{x}_k + \tilde{\mathbf{w}}_k \quad \Rightarrow \tag{42b}$$
$$\mathbf{x}_{k+1} = U\tilde{F}_{T_k}U^{-1}\mathbf{x}_k + U\tilde{\mathbf{w}}_k \quad \Rightarrow \tag{42c}$$
$$\mathbf{x}_{k+1} = F_{T_k}\mathbf{x}_k + \mathbf{w}_k \tag{42d}$$

which gives the transformations

$$F_{T_k} = U\tilde{F}_{T_k}U^{-1}, \tag{43a}$$
$$\begin{aligned}Q_{T_k} &= \mathbb{E}[\mathbf{w}_k\mathbf{w}_k^\mathsf{T}] = \mathbb{E}[U\tilde{\mathbf{w}}_k(U\tilde{\mathbf{w}}_k)^\mathsf{T}] = U\mathbb{E}[\tilde{\mathbf{w}}_k\tilde{\mathbf{w}}_k^\mathsf{T}]U^\mathsf{T} \\ &= U\tilde{Q}_{T_k}U^\mathsf{T}.\end{aligned} \tag{43b}$$

Note that if U is orthogonal, we have $U^{-1} = U^\mathsf{T}$.

Bibliography

A. C. Antoulas. *Approximation of large-scale dynamical systems*. SIAM, Philadelphia, PA, USA, 2005.

P. Axelsson and F. Gustafsson. Discrete-time solutions to the continuous-time differential Lyapunov equation with applications to Kalman filtering. Technical report, Linköping University, Sweden, 2012.

Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software*. John Wiley & Sons, New York, NY, USA, 2001.

C. A. Bavely and G. W. Stewart. An algorithm for computing reducing subspaces by block diagonalization. *SIAM Journal on Numerical Analysis*, 16(2):359–367, 1979.

W. Gawronski. *Advanced structural dynamics and active control of structures*. Springer-Verlag, Berlin, Heidelberg, Germany, 2004.

T. Glad and L. Ljung. *Control theory*. Taylor Francis, New York, NY, USA, 2000.

G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. The Johns Hopkins University Press, Baltimore, MD, USA, 1996.

M. S. Grewal and A. P. Andrews. *Kalman Filtering. Theory and Practice Using MATLAB*. John Wiley & Sons, Hoboken, NJ, USA, third edition, 2008.

N. J. Higham. *Functions of Matrices – Theory and Computation*. SIAM, Philadelphia, PA, USA, 2008.

A. H. Jazwinski. *Stochastic Processes and Filtering Theory*, volume 64 of *Mathematics in Science and Engineering*. Academic Press, New York, NY, USA, 1970.

P. Lancaster and M. Tismenetsky. *The theory of matrices*, volume 2. Academic Press, New York, NY, USA, 1985.

H. J. Rome. A direct solution to the linear variance equation of a time-invariant linear system. *IEEE Transactions on Automatic Control*, 14(5):592–593, October 1969.

W. J. Rugh. *Linear System Theory*. Information and System Sciences Series. Prentice Hall Inc., Upper Saddle River, NJ, USA, second edition, 1996.

S. Särkkä, A. Solin, and J. Hartikainen. Spatio-temporal learning via infinite-dimensional Bayesian filtering and smoothing. *IEEE Signal Processing Magazine*, 30(4):51–61, 2013.

C. F. Van Loan. Computing integrals involving the matrix exponential. *IEEE Transactions on Automatic Control*, 23(3):395–404, June 1978.

N. Wahlström, P. Axelsson, and F. Gustafsson. Discretizing stochastic dynamical systems using Lyapunov equations. In *Proceedings of the The 19th World Congress of the International Federation of Automatic Control (IFAC)*, pages 3726–3731, Cape Town, South Africa, August 2014.

PhD Dissertations
Division of Automatic Control
Linköping University

M. Millnert: Identification and control of systems subject to abrupt changes. Thesis No. 82, 1982. ISBN 91-7372-542-0.

A. J. M. van Overbeek: On-line structure selection for the identification of multivariable systems. Thesis No. 86, 1982. ISBN 91-7372-586-2.

B. Bengtsson: On some control problems for queues. Thesis No. 87, 1982. ISBN 91-7372-593-5.

S. Ljung: Fast algorithms for integral equations and least squares identification problems. Thesis No. 93, 1983. ISBN 91-7372-641-9.

H. Jonson: A Newton method for solving non-linear optimal control problems with general constraints. Thesis No. 104, 1983. ISBN 91-7372-718-0.

E. Trulsson: Adaptive control based on explicit criterion minimization. Thesis No. 106, 1983. ISBN 91-7372-728-8.

K. Nordström: Uncertainty, robustness and sensitivity reduction in the design of single input control systems. Thesis No. 162, 1987. ISBN 91-7870-170-8.

B. Wahlberg: On the identification and approximation of linear systems. Thesis No. 163, 1987. ISBN 91-7870-175-9.

S. Gunnarsson: Frequency domain aspects of modeling and control in adaptive systems. Thesis No. 194, 1988. ISBN 91-7870-380-8.

A. Isaksson: On system identification in one and two dimensions with signal processing applications. Thesis No. 196, 1988. ISBN 91-7870-383-2.

M. Viberg: Subspace fitting concepts in sensor array processing. Thesis No. 217, 1989. ISBN 91-7870-529-0.

K. Forsman: Constructive commutative algebra in nonlinear control theory. Thesis No. 261, 1991. ISBN 91-7870-827-3.

F. Gustafsson: Estimation of discrete parameters in linear systems. Thesis No. 271, 1992. ISBN 91-7870-876-1.

P. Nagy: Tools for knowledge-based signal processing with applications to system identification. Thesis No. 280, 1992. ISBN 91-7870-962-8.

T. Svensson: Mathematical tools and software for analysis and design of nonlinear control systems. Thesis No. 285, 1992. ISBN 91-7870-989-X.

S. Andersson: On dimension reduction in sensor array signal processing. Thesis No. 290, 1992. ISBN 91-7871-015-4.

H. Hjalmarsson: Aspects on incomplete modeling in system identification. Thesis No. 298, 1993. ISBN 91-7871-070-7.

I. Klein: Automatic synthesis of sequential control schemes. Thesis No. 305, 1993. ISBN 91-7871-090-1.

J.-E. Strömberg: A mode switching modelling philosophy. Thesis No. 353, 1994. ISBN 91-7871-430-3.

K. Wang Chen: Transformation and symbolic calculations in filtering and control. Thesis No. 361, 1994. ISBN 91-7871-467-2.

T. McKelvey: Identification of state-space models from time and frequency data. Thesis No. 380, 1995. ISBN 91-7871-531-8.

J. Sjöberg: Non-linear system identification with neural networks. Thesis No. 381, 1995. ISBN 91-7871-534-2.

R. Germundsson: Symbolic systems – theory, computation and applications. Thesis No. 389, 1995. ISBN 91-7871-578-4.

P. Pucar: Modeling and segmentation using multiple models. Thesis No. 405, 1995. ISBN 91-7871-627-6.
H. Fortell: Algebraic approaches to normal forms and zero dynamics. Thesis No. 407, 1995. ISBN 91-7871-629-2.
A. Helmersson: Methods for robust gain scheduling. Thesis No. 406, 1995. ISBN 91-7871-628-4.
P. Lindskog: Methods, algorithms and tools for system identification based on prior knowledge. Thesis No. 436, 1996. ISBN 91-7871-424-8.
J. Gunnarsson: Symbolic methods and tools for discrete event dynamic systems. Thesis No. 477, 1997. ISBN 91-7871-917-8.
M. Jirstrand: Constructive methods for inequality constraints in control. Thesis No. 527, 1998. ISBN 91-7219-187-2.
U. Forssell: Closed-loop identification: Methods, theory, and applications. Thesis No. 566, 1999. ISBN 91-7219-432-4.
A. Stenman: Model on demand: Algorithms, analysis and applications. Thesis No. 571, 1999. ISBN 91-7219-450-2.
N. Bergman: Recursive Bayesian estimation: Navigation and tracking applications. Thesis No. 579, 1999. ISBN 91-7219-473-1.
K. Edström: Switched bond graphs: Simulation and analysis. Thesis No. 586, 1999. ISBN 91-7219-493-6.
M. Larsson: Behavioral and structural model based approaches to discrete diagnosis. Thesis No. 608, 1999. ISBN 91-7219-615-5.
F. Gunnarsson: Power control in cellular radio systems: Analysis, design and estimation. Thesis No. 623, 2000. ISBN 91-7219-689-0.
V. Einarsson: Model checking methods for mode switching systems. Thesis No. 652, 2000. ISBN 91-7219-836-2.
M. Norrlöf: Iterative learning control: Analysis, design, and experiments. Thesis No. 653, 2000. ISBN 91-7219-837-0.
F. Tjärnström: Variance expressions and model reduction in system identification. Thesis No. 730, 2002. ISBN 91-7373-253-2.
J. Löfberg: Minimax approaches to robust model predictive control. Thesis No. 812, 2003. ISBN 91-7373-622-8.
J. Roll: Local and piecewise affine approaches to system identification. Thesis No. 802, 2003. ISBN 91-7373-608-2.
J. Elbornsson: Analysis, estimation and compensation of mismatch effects in A/D converters. Thesis No. 811, 2003. ISBN 91-7373-621-X.
O. Härkegård: Backstepping and control allocation with applications to flight control. Thesis No. 820, 2003. ISBN 91-7373-647-3.
R. Wallin: Optimization algorithms for system analysis and identification. Thesis No. 919, 2004. ISBN 91-85297-19-4.
D. Lindgren: Projection methods for classification and identification. Thesis No. 915, 2005. ISBN 91-85297-06-2.
R. Karlsson: Particle Filtering for Positioning and Tracking Applications. Thesis No. 924, 2005. ISBN 91-85297-34-8.
J. Jansson: Collision Avoidance Theory with Applications to Automotive Collision Mitigation. Thesis No. 950, 2005. ISBN 91-85299-45-6.
E. Geijer Lundin: Uplink Load in CDMA Cellular Radio Systems. Thesis No. 977, 2005. ISBN 91-85457-49-3.
M. Enqvist: Linear Models of Nonlinear Systems. Thesis No. 985, 2005. ISBN 91-85457-64-7.
T. B. Schön: Estimation of Nonlinear Dynamic Systems — Theory and Applications. Thesis No. 998, 2006. ISBN 91-85497-03-7.

I. Lind: Regressor and Structure Selection — Uses of ANOVA in System Identification. Thesis No. 1012, 2006. ISBN 91-85523-98-4.

J. Gillberg: Frequency Domain Identification of Continuous-Time Systems Reconstruction and Robustness. Thesis No. 1031, 2006. ISBN 91-85523-34-8.

M. Gerdin: Identification and Estimation for Models Described by Differential-Algebraic Equations. Thesis No. 1046, 2006. ISBN 91-85643-87-4.

C. Grönwall: Ground Object Recognition using Laser Radar Data – Geometric Fitting, Performance Analysis, and Applications. Thesis No. 1055, 2006. ISBN 91-85643-53-X.

A. Eidehall: Tracking and threat assessment for automotive collision avoidance. Thesis No. 1066, 2007. ISBN 91-85643-10-6.

F. Eng: Non-Uniform Sampling in Statistical Signal Processing. Thesis No. 1082, 2007. ISBN 978-91-85715-49-7.

E. Wernholt: Multivariable Frequency-Domain Identification of Industrial Robots. Thesis No. 1138, 2007. ISBN 978-91-85895-72-4.

D. Axehill: Integer Quadratic Programming for Control and Communication. Thesis No. 1158, 2008. ISBN 978-91-85523-03-0.

G. Hendeby: Performance and Implementation Aspects of Nonlinear Filtering. Thesis No. 1161, 2008. ISBN 978-91-7393-979-9.

J. Sjöberg: Optimal Control and Model Reduction of Nonlinear DAE Models. Thesis No. 1166, 2008. ISBN 978-91-7393-964-5.

D. Törnqvist: Estimation and Detection with Applications to Navigation. Thesis No. 1216, 2008. ISBN 978-91-7393-785-6.

P-J. Nordlund: Efficient Estimation and Detection Methods for Airborne Applications. Thesis No. 1231, 2008. ISBN 978-91-7393-720-7.

H. Tidefelt: Differential-algebraic equations and matrix-valued singular perturbation. Thesis No. 1292, 2009. ISBN 978-91-7393-479-4.

H. Ohlsson: Regularization for Sparseness and Smoothness — Applications in System Identification and Signal Processing. Thesis No. 1351, 2010. ISBN 978-91-7393-287-5.

S. Moberg: Modeling and Control of Flexible Manipulators. Thesis No. 1349, 2010. ISBN 978-91-7393-289-9.

J. Wallén: Estimation-based iterative learning control. Thesis No. 1358, 2011. ISBN 978-91-7393-255-4.

J. Hol: Sensor Fusion and Calibration of Inertial Sensors, Vision, Ultra-Wideband and GPS. Thesis No. 1368, 2011. ISBN 978-91-7393-197-7.

D. Ankelhed: On the Design of Low Order H-infinity Controllers. Thesis No. 1371, 2011. ISBN 978-91-7393-157-1.

C. Lundquist: Sensor Fusion for Automotive Applications. Thesis No. 1409, 2011. ISBN 978-91-7393-023-9.

P. Skoglar: Tracking and Planning for Surveillance Applications. Thesis No. 1432, 2012. ISBN 978-91-7519-941-2.

K. Granström: Extended target tracking using PHD filters. Thesis No. 1476, 2012. ISBN 978-91-7519-796-8.

C. Lyzell: Structural Reformulations in System Identification. Thesis No. 1475, 2012. ISBN 978-91-7519-800-2.

J. Callmer: Autonomous Localization in Unknown Environments. Thesis No. 1520, 2013. ISBN 978-91-7519-620-6.

D. Petersson: A Nonlinear Optimization Approach to H2-Optimal Modeling and Control. Thesis No. 1528, 2013. ISBN 978-91-7519-567-4.

Z. Sjanic: Navigation and Mapping for Aerial Vehicles Based on Inertial and Imaging Sensors. Thesis No. 1533, 2013. ISBN 978-91-7519-553-7.

F. Lindsten: Particle Filters and Markov Chains for Learning of Dynamical Systems. Thesis No. 1530, 2013. ISBN 978-91-7519-559-9.

P. Axelsson: Sensor Fusion and Control Applied to Industrial Manipulators. Thesis No. 1585, 2014. ISBN 978-91-7519-368-7.

A. Carvalho Bittencourt: Modeling and Diagnosis of Friction and Wear in Industrial Robots. Thesis No. 1617, 2014. ISBN 978-91-7519-251-2.

M. Skoglund: Inertial Navigation and Mapping for Autonomous Vehicles. Thesis No. 1623, 2014. ISBN 978-91-7519-233-8.

S. Khoshfetrat Pakazad: Divide and Conquer: Distributed Optimization and Robustness Analysis. Thesis No. 1676, 2015. ISBN 978-91-7519-050-1.

T. Ardeshiri: Analytical Approximations for Bayesian Inference. Thesis No. 1710, 2015. ISBN 978-91-7685-930-8.