Karlsruhe Series on
Intelligent Sensor-Actuator-Systems

7

Marco Huber

Probabilistic Framework for Sensor Management

universitätsverlag karlsruhe

Marco Huber

Probabilistic Framework for Sensor Management

Karlsruhe Series on Intelligent Sensor-Actuator-Systems

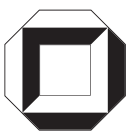
Volume 7

ISAS | Universität Karlsruhe (TH)
Intelligent Sensor-Actuator-Systems Laboratory

Edited by Prof. Dr.-Ing. Uwe D. Hanebeck

Probabilistic Framework for Sensor Management

by
Marco Huber



universitätsverlag karlsruhe

Dissertation, Universität Karlsruhe (TH)
Fakultät für Informatik, 2009

Impressum

Universitätsverlag Karlsruhe
c/o Universitätsbibliothek
Straße am Forum 2
D-76131 Karlsruhe
www.uvka.de

Dieses Werk ist unter folgender Creative Commons-Lizenz
lizenziert: http://creativecommons.org/licenses/by-nc-nd/3.0/de/

Universitätsverlag Karlsruhe 2009
Print on Demand

ISSN: 1867-3813
ISBN: 978-3-86644-405-8

Probabilistic Framework for Sensor Management

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

der Fakultät für Informatik
der Universität Fridericiana zu Karlsruhe (TH)

genehmigte

Dissertation

von

Marco Huber

aus Kehl

Tag der mündlichen Prüfung:	30.04.2009
Erster Gutachter:	Prof. Dr.-Ing. Uwe D. Hanebeck
Zweiter Gutachter:	Dr. rer. nat. Wolfgang Koch

Acknowledgement

A work like this greatly benefits from the support of many people. I would like to spend the following lines for thanking especially those mentioned below.

This thesis was conducted at the Intelligent Sensor-Actuator-Systems Laboratory (ISAS) of the Universität Karlsruhe (TH) within the DFG Research Training Group 1194 "Self-organizing Sensor-Actuator-Networks". I am grateful to *Uwe D. Hanebeck*, director of the ISAS, for valuable discussions, advising me and giving me the opportunity to work in such a rewarding research environment. I would also like to thank *Wolfgang Koch*, my co-advisor, for reviewing the thesis, for his suggestions and encouragement.

Thanks to the people at the ISAS, who have helped to make things entertaining and stimulating, and who have often provided a welcome respite from research. In particular, I must thank *Florian Weißel* and *Frederik Beutler* for their valued friendship. To Florian, thanks for your intelligence, for helping with words and deeds, and for the memorable nights we spent in the lab with paper writing. To Fred, thanks for your ever-friendly character, your unlimited willingness to help people, and for maintaining the uninterrupted supply with beer and movies. I would also like to thank *Oliver Schrempf* and *Peter Krauthausen* for letting me hanging around in their office and just for listening, *Felix Sawo* as the other "sensor network guy" at the ISAS for many valuable collaborations and his artistic talent, as well as *Marc Deisenroth* for many fruitful discussions, for being an (remote) inspiration, and for the recreative Pub and College Hall visits. Many thanks go to the students working together with me at the ISAS on several research projects, especially to *Christof Chlebek*, *Patrick Ding*, *Achim Kuwertz*, *Johannes Meyer*, and *Eric Stiegeler*. Their interest and engagement added a great deal to the success of this thesis.

I would also like to thank *Carl Rasmussen* and *Hugh Durrant-Whyte* for showing their interest in my work and for inviting me to work in their groups in Cambridge and Sydney. Special thanks go to *Tim Bailey* for his bright ideas and for proof-reading my thesis, and to *Thierry Peynot* and *Jairo Escobar*, for providing entertaining diversion in so many ping pong and table football sessions.

To my parents *Walter* and *Anita*, my brother *Stefan* and all my friends, thank you for a lifetime of care and support.

Last, but by no means least, I would like to thank *Karin*. You were and are by far my most valuable support. I am grateful for your constancy in love and understanding. This thesis is dedicated to you.

Karlsruhe, June 2009 Marco Huber

To Karin

Contents

Notation

Conventions

x	Scalar
$\underline{x}$	Column vector
$\boldsymbol{x}, \underline{\boldsymbol{x}}$	Random variable/vector
$\hat{x}, \underline{\hat{x}}$	Mean value/vector of random variable/vector
$x^*, \underline{x}^*$	Optimal value/vector
n_x	Dimension of the variable $\underline{x}$
$\underline{x}_k$	Vector at time step k
$\underline{x}_{0:k}$	Time sequence of vectors $(\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_k)$
$\underline{x}_{k,n}$	Vector at time step n within the control time horizon
$\underline{x}_k^{\underline{u}}$	Shorthand term for $\underline{x}_k(\underline{u})$, i.e., vector $\underline{x}_k$ depends on the configuration vector $\underline{u}$
$\mathbf{A}$	Matrices are denoted by bold upper case letters
$\succeq$	Positive semi-definite ordering relation between matrices, i.e., if $\mathbf{A} \succeq \mathbf{B}$ then $\mathbf{A} - \mathbf{B}$ is positive semi-definite
$\mathcal{A}$	Sets are denoted by calligraphic upper case letters
$\mathbb{R}$	Real numbers
$\sim$	Distribution operator, e.g., $\boldsymbol{x} \sim f(x)$ indicates that $\boldsymbol{x}$ is distributed according to the probability density function $f(x)$
■	End of example
□	End of proof

Symbols for Probability Density Functions

$f(\cdot)$	General probability density function
$\tilde{f}(\cdot)$	True probability density function (to be approximated by $f(\cdot)$)
$F(\cdot)$	Cumulative distribution functions are denoted by the upper case letter corresponding to the density function symbol
$f(\cdot\,;\underline{\eta})$	Parameterized density function
$f(\underline{x}\vert\underline{z})$	Conditional density function, i.e., conditioning of the random vector $\underline{\boldsymbol{x}}$ on the vector $\underline{z}$
$f_k^x(\underline{x})$	Density function of random vector $\underline{\boldsymbol{x}}_k$ at time step k
$f_k^T(\underline{x}_{k+1}\vert\underline{x}_k)$	Transition density
$f_k^L(\underline{\hat{z}}_k\vert\underline{x}_k)$	Likelihood function
$f_{k+1}^p(\underline{x}_{k+1})$	Predicted density at time $k+1$
$f_k^e(\underline{x}_k)$	Posterior density after measurement update at time k
$\mathcal{N}(\underline{x};\underline{\hat{x}},\mathbf{C}_x)$	Multivariate Gaussian density function
$\mathrm{erf}(x)$	Gaussian error function
$\delta(\underline{x})$	Dirac delta distribution
$H(x)$	Heaviside step function

Further Function Symbols

$\mathrm{diag}(\underline{x})$	Diagonal matrix, whose main diagonal elements are $\underline{x}$
$\mathbf{A}^{\mathrm{T}}$	Matrix transpose
$\mathbf{A}^{-1}$	Inverse of a matrix
$\mathrm{trace}(\mathbf{A})$	Trace of the matrix $\mathbf{A}$
$\lvert\mathbf{A}\rvert$	Determinant of the matrix $\mathbf{A}$
$\lvert\mathcal{A}\rvert$	Cardinality of the set $\mathcal{A}$
$\mathrm{E}\{\cdot\}$	Expectation operator
$H(\underline{\boldsymbol{x}})$	Entropy of the random vector $\underline{\boldsymbol{x}}$
$I(\underline{\boldsymbol{x}};\underline{\boldsymbol{z}})$	Mutual information between the random vector $\underline{\boldsymbol{x}}$ and $\underline{\boldsymbol{z}}$
$D\big(\tilde{f}(\underline{x})\lvert\lvert f(\underline{x})\big)$	Kullback-Leibler divergence between the probability density functions $\tilde{f}(\underline{x})$ and $f(\underline{x})$
$g_k(\underline{\boldsymbol{x}}_k, \underline{u}_k)$	Step objective function for time step k
$J_k(\underline{\boldsymbol{x}}_k)$	Cumulative objective function for time step k
$V_k(\underline{\boldsymbol{x}}_k, \underline{u}_k)$	Configuration dependent cumulative objective function for time step k
$O(g)$	Big-O in Landau notation, e.g., $f(x) \in O(g(x))$ indicates that the function $f(x)$ has the order of $O(g(x))$

Glossary

BB	Branch-and-bound
EKF	Extended Kalman filter
GE	Gaussian estimator
GME	Gaussian mixture estimator
HDF	Hybrid density filter
IBP	Information-based pruning
KF	Kalman filter
KLD	Kullback-Leibler divergence
LRKF	Linear regression Kalman filter
MDP	Markov decision process
ODE	Ordinary differential equation
OLFC	Open-loop feedback control
PBAB	Probabilistic branch-and-bound
PF	Particle filter
PGMR	Progressive Gaussian mixture reduction
POMDP	Partially observable Markov decision process
UKF	Unscented Kalman filter

Zusammenfassung

Die Sensoreinsatzplanung (Sensor Management) stellt eine Erweiterung der stochastischen Zustandsschätzung dar, bei der durch gezielte Beeinflussung der Messparameter der verwendeten Sensoren eine signifikante Verbesserung der Schätzung interner Systemzustände erzielt werden kann. Für diese Maximierung des Nutzwerts von Messungen wird das zukünftige Verhalten des beobachteten Systems durch eine vorausschauende Planung, d. h. durch eine Prädiktion des Systemverhaltens über einen endlich langen Zeithorizont, miteinbezogen. Auf diese Weise lassen sich auch weitergehende Aufgaben, etwa die Reduzierung des Verbrauchs begrenzter Energie- und Rechenressourcen, behandeln.

Im Gegensatz zu vielen verfügbaren Verfahren zur Sensoreinsatzplanung, welche auf spezielle Problemklassen zugeschnitten sind, gestattet das in dieser Arbeit vorgestellte Framework eine möglichst allgemeine Anwendbarkeit. Hierzu wird von einer nichtlinearen Modellierung der Systeme und Sensoren sowie von einem kontinuierlichen Zustandsraum ausgegangen. Während schon die alleinige stochastische Zustandsschätzung unter diesen Vorgaben im Allgemeinen nicht geschlossen lösbar ist, verschärft sich diese Problematik durch die Hinzunahme der Einsatzplanungskomponente. Aus diesem Grund werden in dieser Arbeit für die beiden zentralen Bausteine des Frameworks, also Zustandsschätzer und Einsatzplaner, approximative Verfahren vorgestellt.

Eine zentrale Herausforderung bei der Einsatzplanung stellt das Antizipieren von Messungen innerhalb des Zeithorizonts dar. Die beiden in dieser Arbeit vorgestellten Einsatzplaner verfolgen hierfür unterschiedliche Strategien. Beim *quasi-linearen Planungsverfahren* werden keine zukünftigen Messwerte antizipiert. Dies entspricht einer Open-Loop-Regelung bei der keine Zustandsrückführung erfolgt. Stattdessen wird das nichtlineare Einsatzplanungsproblem durch prädiktive statistische Linearisierung über den betrachteten Horizont auf ein lineares Sensoreinsatzplanungsproblem abgebildet. Dadurch lässt sich die optimale Sequenz von Messparametern unabhängig von zukünftigen Messwerten mittels einer Baumsuche bestimmen, wobei allerdings der erforderliche Aufwand exponentiell mit der Länge des Horizontes wächst. Zur Aufwandsreduktion wird ein Baumbeschneidungsverfahren vorgestellt, welches auf der Grundlage einer Abschätzung des Informationsbeitrags bzw. des Nutzens von Messparametern das frühe Entfernen kompletter Teilbäume erlaubt.

Im Falle starker Nichtlinearitäten und multimodaler Wahrscheinlichkeitsdichten ist die Abbildung auf das lineare Einsatzplanungsproblem nur eingeschränkt zulässig. Beim *informationstheoretischen Einsatzplanungsverfahren* erfolgt stattdessen das Antizipieren zukünftiger Messwerte und somit eine Closed-Loop-Regelung. Zu diesem Zwecke werden repräsentative zukünftige Messwerte auf Grundlage der aktuellen Zustandsschätzung erzeugt. Durch die Verwendung informationstheoretischer Maße zur

Bewertung des Nutzwertes der Messparameter kann sämtliche in den Wahrscheinlichkeitsdichten kodierte Information quantifiziert und miteinbezogen werden. Allerdings ist die geschlossene Auswertung informationstheoretischer Maße, welche auf Shannons Entropie basieren, für die verwendeten Gaußmischdichten zur Repräsentierung der auftretenden Wahrscheinlichkeitsdichten nicht möglich. Um den Einsatz aufwändiger numerischer Näherungslösungen zu vermeiden, werden enge, leicht zu berechnende Schranken für die tatsächliche Entropie hergeleitet und eingesetzt.

Den zweiten zentralen Baustein des Frameworks stellt die effiziente Zustandsschätzung dar. Neben der Inferenz des Zustands nach vorliegendem Messwert, erfolgen auch im Einsatzplaner selbst, aufgrund der vorausschauenden Planung, wiederholt Zustandsschätzungen. Das entwickelte *Gaußfilter* kommt unmittelbar in der quasi-linearen Sensoreinsatzplanung zwecks statistischer Linearisierung zum Einsatz. Hierbei wird neben der gängigen Berücksichtigung der ersten beiden Momente einer Schätzung auch die Form der Verteilungsfunktion zur Linearisierung miteinbezogen. Für die hochqualitative Zustandsschätzung, insbesondere zur Inferenz des Zustands aber auch als Schätzerkomponente für die informationstheoretische Planung, dient das sogenannte *Hybrid Density Filter*. Dieser Bayes'sche Schätzer verfolgt den Ansatz einer Approximation der Transitionsdichten, welche eine probabilistische Repräsentation der nichtlinearen System- und Sensormodelle darstellen. Durch die Verwendung der namensgebenden hybriden Dichte zur Approximation, welche aus Gaußdichten und Delta-Distributionen besteht, kann das Schätzproblem geschlossen gelöst werden, wobei sich eine Gaußmischdichte als Repräsentation des Schätzergebnisses ergibt.

Die rekursive Verarbeitung der auftretenden Gaußmischdichten hat ein unbegrenztes Wachstum der Anzahl an Gaußkomponenten zur Folge. Dieser Zuwachs lässt sich durch ein neuartiges *progressives Reduktionsverfahren* begrenzen. Im Gegensatz zu gängigen Reduktionsverfahren erfolgt die Verkleinerung der Komponentenanzahl nicht durch schrittweises Entfernen von Komponenten. Stattdessen wird mit einer einzelne Gaußdichte begonnen und diese an die über eine Homotopie sukzessive eingeführte komplexe Mischdichte angepasst. Wann immer das Einhalten einer vordefinierten Approximationsgüte nicht länger möglich ist, werden zusätzliche Komponenten hinzugefügt. Aufgrund der besseren Ausnutzung von Redundanzen resultiert dieses konstruktive Vorgehen in einer deutlich stärkeren Reduktion bei zugleich sehr hoher Approximationsgüte.

Das Zusammenwirken der entwickelten Verfahren zur Sensoreinsatzplanung und nichtlinearen Zustandsschätzung wird anhand diverser Objektverfolgungsaufgaben evaluiert. Unter anderem werden dabei die Sensorauswahl in Sensornetzwerken sowie die Bewegungsplanung mobiler Sensoren betrachtet.

Abstract

Sensor management extends classical state estimation in such a way that significantly more accurate estimates are provided by reason of systematically exploiting the modalities of the employed sensors. To achieve this utility maximization of measurements, the future behavior of the observed system is incorporated via predictive planning, i.e., by predicting the system behavior over a finite time horizon. In doing so, dealing with advanced requirements and applications, e.g., reducing the consumption of limited energy, computation, and communication resources as it is the case in sensor networks, is possible.

In contrast to the variety of existing sensor management approaches that are fitted to specific classes of problems, the framework proposed in this thesis is versatilely applicable. Besides nonlinear models for the observed system and the sensors, a continuous state space and continuous-valued measurements are assumed. With this setting, even solving the state estimation problem in closed form is not possible. By adding the sensor management aspect, this issue gets even worse. On this account, for both central components of the framework, i.e., state estimator and sensor manager, approximative approaches are proposed for a feasible sensor management.

The anticipation of future measurements within the time horizon possesses a major challenge of sensor management. For coping with this challenge, both sensor managers proposed in this thesis pursue different strategies. The *quasi-linear sensor management* approach employs an open-loop control strategy where no state feedback is employed, i.e., no future measurements are anticipated. Instead, predictive statistical linearization over the considered time horizon is used for converting the nonlinear non-Gaussian sensor management problem into a linear Gaussian one. To significantly reduce the complexity of determining the optimal configuration sequence, which depends exponentially on the length of the time horizon, a novel optimal pruning method is introduced.

The *information theoretic sensor management* approach is aimed at scenarios, where the conversion into a linear Gaussian sensor management problem is of limited fidelity. It employs a closed-loop control scheme, where meaningful future measurement values are explicitly incorporated into the sensor management decisions. The utility of the sensor configurations is quantified by means of mutual information as objective function, while state estimation and density representation rely on Gaussian mixture densities for accurately encoding the effect of the measurement value anticipation.

Efficient state estimation represents the second central component of the framework. Besides inferring the system state given an actual measurement value, state estimation is also applied multiple times within the sensor managers. Here, the *Gaussian estimator* is employed directly within the quasi-linear sensor manager for statistical

linearization. In contrast to existing linearization approaches, not only the first two moments of the state estimate are considered but also the shape of the distribution function and thus, information on higher-order moments is incorporated. For high quality estimation, especially for state inference given an actual measurement as well as for state estimation within the information theoretic sensor manager, the *hybrid density filter* is proposed. This Bayesian estimator employs the approach of approximating transition densities, which form a probabilistic representation of the nonlinear system and sensor models. Due to the eponymous hybrid density functions consisting of Gaussians and Dirac delta distributions, the estimation problem can be solved analytically resulting in a Gaussian mixture presentation of the state estimate.

The recursive processing of the occurring Gaussian mixtures often results in an unbounded growth of the Gaussian mixture components. For bounding this growth, a novel *progressive Gaussian mixture reduction* algorithm is proposed for feasible state estimation and sensor management, respectively. Compared to existing reduction methods, reducing the number of components does not commence from the given complex mixture. Instead, a new Gaussian mixture is constructed in a bottom-up fashion. By employing homotopy continuation, the new mixture can be constantly adapted and its complexity needs only to be increased whenever the approximation capability of the mixture is not longer sufficient. This approach leads to an improved exploitation of redundancies and thus, to a more significant reduction with a high approximation quality at the same time.

The effectiveness and the interdependency of the proposed methods for state estimation and sensor management is evaluated by means of different target localization and tracking tasks. Here, sensor scheduling in sensor networks as well as motion control of a mobile sensor are considered in particular.

CHAPTER 1

Introduction

The recent advances in miniaturization, wireless communication, and sensor technology make it possible to build up and deploy sensor systems with a large variety of sensing modalities for a smart and persistent surveillance. For instance, sensor networks consisting of numerous inexpensive, possibly mobile sensors are a popular subject in research and practice. The main objective of *sensor management* in such sensor systems is to collect information about the observed phenomenon or object by utilizing the sensors and their sensing modalities in a *most informative way*, which requires making decisions involving multiple time steps. Each decision generates observations or measurements providing additional information about the internal state of the observed system. However, the internal state, e.g., position and velocity of a vehicle or temperature distributions in buildings, can typically only be observed indirectly and uncertainly, whereby the state information inferred from the observations and the outcome of any decision is not known for certain. Each subsequent decision is then made based on this uncertain knowledge.

Even if the sensor management problem is discussed in this thesis from a technical and scientific point of view, it is also present in everyday life. For instance, the card-game *Texas hold'em poker* (see e.g. [62] for game rules and complexity), which has become very popular due to the large media coverage, can convey an impression of the sensor management problem:

Example 1.1: Texas Hold'em Poker and Sensor Management
A person is playing Texas hold'em poker against two other players. In the current game, the person's two private cards are two aces, while the three face-up cards at the board (the so-called flop) are a four, a five, and a six, all of a different suit. The cards in the person's hand and the three cards of the flop are the only cards the person can see. All cards in the game form the internal state. The person could be beaten, if one of the opponents could complete the cards of the flop to a "three of a kind", "four of a kind" or a "straight". Each of these hands has a specific probability of occurrence. To obtain information about the unseen cards, the person can observe the reaction of the opponents when a bet is placed, called (matched) or even raised (increased) by himself or another player.

The person can evaluate different strategies considering the unseen cards and the potential win or loss of money. In this round, the person decides to approach carefully by placing the minimum amount of money possible. While one of the opponents drops out, the second one, who is holding a pair of kings, even raises the initial bet by doubling the amount of money. Now, the player knows that the opponent left is either bluffing or has a good hand. He decides to call the bet and wait for the dialer's next card, which is a ten. Hence, the person will win the game (but still without knowing it for certain), unless he will not be made insecure by the reactions of his opponent. ■

In summary, sensor management more or less considers the problem of making decisions sequentially over time in a diverse system of sensors and sensing modalities based on inferring the internal state and potential observations from uncertain information.

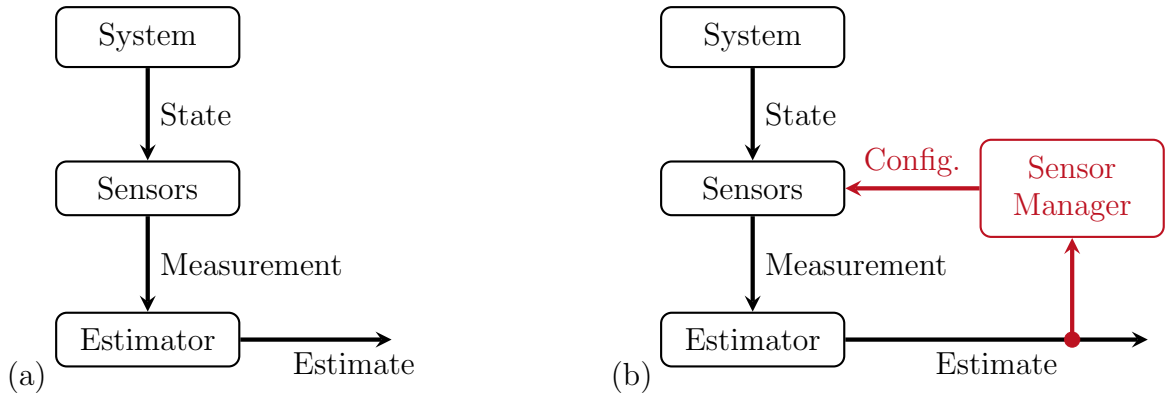


Figure 1.1: (a) Classical state estimation. (b) State estimation with sensor management.

1.1 State Estimation and Sensor Management

State estimation theory, which considers the problem of inferring the internal state of the observed system from uncertain, noisy observations, forms the foundation of sensor management. In classical state estimation as depicted in Figure 1.1 (a), the information provided by the sensors is merely used to estimate the temporal evolution of the system state. However, in many applications significantly better state estimation results, i.e., information on the system state can be obtained by exploiting and adapting the sensing modalities of the sensors. Exemplary applications include environmental monitoring by means of a sensor network [47, 53], localization of landmines [87] or the identification of contamination sources by means of mobile sensors [38, 130]. To maximize the utility of sensor measurements, sensor management extends classical state estimation by an additional component, which is referred to as *sensor manager* as depicted in Figure 1.1 (b).

In case of the considered probabilistic and continuous modeling of the occurring uncertainties, even solving the state estimation problem generally requires the computation and storage of continuous-valued functions with no finite parameterization. By additionally considering the sensor management aspect, a stochastic control problem on the basis of the state estimates for configuring the sensors has to be solved. Determining the solution of this control problem involves optimization over multiple time steps, where future, not yet known measurements of the sensors are anticipated and the sensors are treated jointly for improved estimation results [152]. Despite of few exceptions including the well-studied linear Gaussian systems (see e.g. [6, 96, 126]), no analytical and closed-form solution of the state estimation and consequently of the sensor management problem can be found. In order to avoid computationally demanding numerical solutions, especially with regard to resource-constrained applications, it is inevitable to apply efficient approximation techniques for feasible sensor management.

In recent years, many approximate solution approaches to the sensor management problem have been proposed, where the optimization is often performed only over the next time step, which is commonly referred to as greedy or myopic sensor management. On the other side, extensions to multiple time steps are often restricted to specific problem classes (see Section 2.6 for a detailed overview). Less attention is paid in sensor management literature to the underlying estimation problem, whose accurate and efficient treatment builds the foundation for feasible sensor management. On this account and in contrast to existing approaches, the framework proposed in this thesis provides techniques for solving the overall sensor management problem in itself *and* for performing efficient state estimation, whereas less assumptions on the underlying estimation problem are made in order to facilitate versatile applicability.

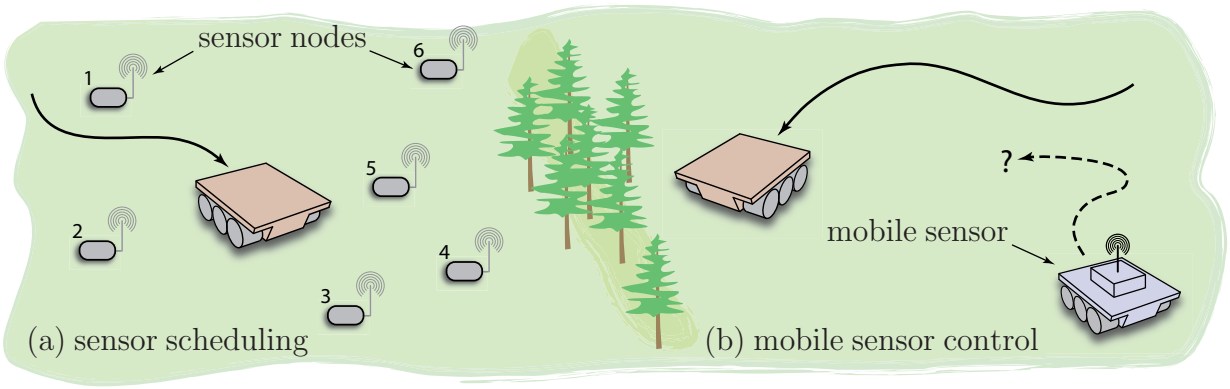


Figure 1.2: Considered canonical problems: (a) sensor scheduling in a sensor network and (b) control of a mobile sensor for localizing and tracking an unknown movable object (red vehicle).

1.2 Canonical Problems and Applications

In order to demonstrate the utility of the proposed sensor management and state estimation techniques, this section is devoted to a detailed description of two canonical sensor management problems examined in this thesis. In Figure 1.2, the *sensor scheduling* problem in a sensor network and the *mobile sensor control* problem are depicted. It is important to note that the proposed framework is not restricted to these problems. In fact, many other sensor management problems share the same structure and thus, the provided insight can be transferred.

For both canonical sensor management problems, localizing and tracking of a single movable object is considered as example application throughout this thesis. The aim of this application, which is typically referred to as *(single) target localization and tracking* and which is a well-studied estimation task (see for example [14, 117] and the references therein), can be generally described as estimating the target kinematics such as position, orientation or velocity from noisy measurements obtained from sensors. In consideration of both canonical sensor management problems, target tracking is performed by either scheduling sensors in a sensor network or by controlling the motion of a mobile sensor, as depicted in Figure 1.2 (a) and (b), respectively. For both problems, the objective is to influence the sensors in such a way that informative measurements are performed in order to reduce the uncertainty of the target kinematics to a low level. Hereby, the main focus is on exposing the effect of the proposed management and estimation techniques of the framework. Further aspects associated to target tracking such as target detection, measurement gating or target classification are omitted for clarity reasons.

In the following, both canonical problems, i.e., sensor scheduling and mobile sensor control are introduced in detail.

1.2.1 Problem: Scheduling in Sensor Networks

For monitoring physical systems as it is the case in, e.g., environmental monitoring, structural monitoring of buildings, object tracking, or surveillance tasks [2, 46], large-scale *sensor networks* have attracted great interest in research and practice. For a meaningful and detailed view on the physical system, an intelligent processing of the data provided by the distributed sensor nodes is essential.

Obviously, the best state estimate of the observed system is obtained if each sensor node of the network performs measurements at each time instant. However, the obtained measurement values have to be transmitted to some kind of fusion center or leader node, which requires costly wireless communication and may lead to communication errors or breakdowns due to interferences if many sensor nodes transmit their measurement values simultaneously. Furthermore,

often only those measurements are informative that are provided by sensors in close vicinity of the observed system [189]. To increase the operational lifetime of the sensor network, the measurement rate should be as low as possible, which on the other hand leads to a decrease in information gain and consequently in estimation accuracy. *Sensor scheduling*, which is also referred to as sensor selection, constitutes a specific sensor management problem and a promising solution to this trade-off. A sensor schedule specifies a time sequence of sensor nodes to be allocated for performing future measurements. It can be also considered as some kind of time multiplex for measurement transmission.

Problems of selecting one out of many sensing modalities of a sensor are closely related to sensor scheduling. An example is the selection of the beam position in case of a phased-array radar for detecting unknown objects and for updating the location of known objects (see e.g. [99]). At each time step, the sensor manager has to decide which object needs to be observed next.

1.2.2 Problem: Mobile Sensor Control

A different type of sensor management problems arises, if the sensor manager has to control the motion of a mobile sensor in order to perform informative measurements, e.g., to detect a contamination source [143] or to estimate unknown parameters of spatially distributed systems [149]. Here, the sensor possesses an internal state, which evolves over time in dependence of the sensor management decisions and which affects the outcome of the sensor measurements in the long term. Due to the internal state, optimization over multiple time steps is here even more important compared to the previously discussed scheduling problem. Consider for example the detection of a movable source of contamination. If the maneuverability of the mobile sensor is constrained, the future behavior of the source has to be anticipated at an early stage for obtaining good sensor positionings before all traces of the source are vanished and thus, the source will be untraceable.

The structure of this management problem also occurs in sensor network tasks like the dynamical routing of sensor measurements through the network or the time varying assignment of the role of a fusion center/leader node to sensor nodes for minimizing communication costs [189].

1.3 Outline and Contributions

As stated before, this thesis consists of two main parts, not accounting for this introduction, problem statement, and conclusions. In the first part of the thesis (Chapter 3 and Chapter 4), two different sensor management approaches are introduced, whose application domain depends on the complexity of the underlying estimation problem, i.e., whether the estimation problem is close to the well-known linear Gaussian problem or not. Both management approaches are tightly knit with the proposed estimation techniques that form the second part of the framework (Chapter 5, Chapter 6, and Chapter 7). The structure of the thesis and the relationship between the chapters is illustrated in Figure 1.3. In the following, a brief overview of the chapters and their contributions is given.

Chapter 2 In this chapter, the sensor management problem is described formally and a detailed view on the sensor manager component as well as its degrees of freedom is provided. Here, the introduction to Bayesian estimation and stochastic control, which build the theoretical foundation of sensor management, is of special interest. Due to the probabilistic modeling of the uncertainties affecting the sensor measurements and the temporal evolution of the observed

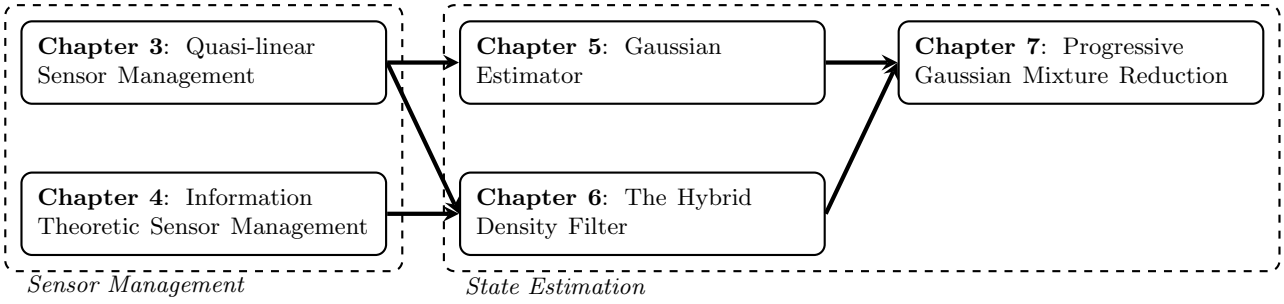


Figure 1.3: Interdependency between the chapters of this thesis.

system and due to the assumption of a continuous state space and measurement space, it is pointed out that these theories merely provide a conceptual solution to the sensor management problem. The chapter closes with a survey of typical objective functions for quantifying the utility of measurements as well as a survey of existing work on sensor management.

Chapter 3 For an effective sensor management, the long-term effects of decisions on sensor configurations have to be considered, which requires optimization over long time horizons. With regard to computationally constrained sensor systems, the proposed *quasi-linear sensor management* approach facilitates efficient and meaningful decision making on the basis of two key features: (1) statistical linearization for converting the nonlinear non-Gaussian sensor management problem into a linear Gaussian one and (2) optimal pruning for quickly solving the linear Gaussian problem. The conversion coincides with an open-loop control approximation, which explicitly captures the uncertainty of the system state over the time horizon. Due to the conversion, the evolution of the uncertainty is exclusively described by the evolution of the system covariance and thus, it is independent of actual measurement values. By this means, optimizing the sensor configurations can be performed in a deterministic fashion, where the information contribution of the sensor configurations for reducing the state covariance can be easily exploited. This in turn facilitates optimal pruning for an early exclusion of insufficient configurations from the optimization.

Chapter 4 In case of strong nonlinearities and multimodal probability density functions, the conversion into a linear Gaussian sensor management problem is of limited fidelity. For this reason, the *information theoretic sensor management* approach pursues a closed-loop model predictive control strategy, where the arrival of future measurements is anticipated and the measurement values are incorporated in the state estimates. This anticipation and incorporation is based on a deterministic sampling of future measurement values from predicted measurement density functions that in turn result from the current state density function. In order to achieve high accuracy, the occurring probability density functions are represented and propagated in form of Gaussian mixtures. By employing mutual information for quantifying the utility of the sensor configurations, an accurate capturing and incorporation of the information encoded in the density functions is provided. Unfortunately, the Gaussian mixture representation of the density functions refuses a closed-form evaluation of the mutual information. In order to avoid the application of demanding numerical approximation techniques, tight and computationally cheap bounds on the entropy terms that form mutual information are derived instead.

Chapter 5 Statistical linearization is a key technique the quasi-linear sensor manager (Chapter 3) relies on. For an effective determination of regression points, a novel deterministic approximation scheme is developed. Here, a parametric density function representation of the

regression points is employed, which allows approximating the cumulative distribution function of the Gaussian density that represents the state estimate. On the basis of this regression point selection scheme, a *Gaussian estimator* is introduced, for which the computationally demanding parts for regression point determination are carried out off-line in order to obtain an efficient estimator. Additionally, an extension for non-Gaussian scenarios is presented, where the state estimates are represented by means of Gaussian mixture densities. In contrast to existing estimators employing statistical linearization like the well-known unscented Kalman filter [91, 183], estimation quality of the novel Gaussian estimator can be adapted by adjusting the number of regression points used.

Chapter 6 In nonlinear state estimation, it is generally inevitable to incorporate approximations of the exact estimation algorithm. There are two possible ways for approximation: Approximating the nonlinear system and sensor models or approximating the probability density function of the state. The key idea of the introduced novel estimator called *hybrid density filter* (HDF) relies on approximating the probabilistic representation of the nonlinear system and sensor models, namely the transition density and the likelihood, respectively. These conditional densities relate the current system state to the future system state for the prediction step or to potential measurements for the measurement update step. A hybrid density consisting of both Dirac delta distributions and Gaussian densities is used for approximation. This chapter addresses the optimization problem for treating the conditional density approximation. Furthermore, efficient estimation algorithms are derived based upon the special structure of the hybrid density, which yields a Gaussian mixture representation of the state density.

Chapter 7 For an efficient state estimation regarding the mixture version of the Gaussian estimator (Chapter 5), the hybrid density filter (Chapter 6), and many other existing state estimators, it is inevitable to approximate the occurring Gaussian mixtures by mixtures with fewer components to keep the computational and memory complexity bounded. Appropriate approximations can be typically generated by exploiting the redundancy in the shape description of the original mixture. In contrast to the common approach of successively merging pairs of components to maintain a desired complexity, the novel *progressive Gaussian mixture reduction* (PGMR) algorithm introduced in this chapter avoids to directly reduce the original Gaussian mixture. Instead, an approximate mixture is generated in a bottom-up fashion by employing homotopy continuation. This allows starting the approximation with a single Gaussian, which is constantly adapted to the progressively incorporated true Gaussian mixture. Whenever a user-defined bound on the deviation of the approximation cannot be maintained during the continuation, further components are added to the approximation. This constructive procedure facilitates a significant reduction of the number of components even for complicated Gaussian mixtures.

Chapter 8 The thesis closes with conclusions and an outlook on future extensions to the proposed sensor management framework.

CHAPTER 2

Considered Problem

The sensor management problem has strong relations to stochastic control problems for dynamic systems. However, compared to these problems, there is one essential difference. Instead of controlling the evolution of the dynamic system by selecting appropriate control inputs, sensor management focuses on controlling the evolution of the information gathering on the system state. Once new information becomes available, it is forwarded to an estimator for inferring the state of the dynamic system. Thus, sensor management requires the understanding of several topics from the fields of stochastic optimal control and Bayesian state estimation. This chapter is concerned with establishing the background knowledge on methods from these fields, which are relevant for the proposed probabilistic sensor management framework. Furthermore, the basic assumptions as well as the general structure of the proposed framework are pointed out.

The employed probabilistic models of the observed physical system and of the sensors are introduced in Section 2.1. Based on this information, the general structure of the sensor management framework is described, which fundamentally employs a model predictive and moving horizon control scheme, respectively. The several degrees of freedom arising from this control scheme and the specification of these degrees of freedom in the proposed *quasi-linear sensor management* approach and the *information theoretic sensor management* approach are outlined in Section 2.2. Another essential part of the framework are the employed estimators for planning and for on-line state inference. A brief introduction to Bayesian state estimation, the arising computational problems and an overview on the relevant and utilized (approximate) state-of-the-art estimators is given in Section 2.3. In Section 2.4, those parts of the stochastic control theory relevant for this thesis are briefly described. Moreover, a motivation for employing model predictive control is given. For specifying certain control objectives, the stochastic controller relies on objective functions. In this thesis, both covariance-based and information theoretic objectives are considered. A description of important properties of these objective functions is part of Section 2.5. Finally in Section 2.6, existing work on sensor management related to this thesis is surveyed.

2.1 Probabilistic Models

Throughout this thesis, probabilistic models for characterizing the dynamic behavior of the physical system and for mathematically relating the state of the system to the sensor measurements are employed. The probabilistic models of the system and the sensors are given in state space form, where all uncertainties of the models are described by means of discrete-time stochastic processes. Here, the state vector $\underline{\boldsymbol{x}}_k \in \mathbb{R}^{n_x}$ of the system comprises the smallest set of variables necessary for completely determining the dynamic behavior at time steps $k \geq 0$.

2.1.1 System Model

The propagation of $\underline{\boldsymbol{x}}_k$ from time step k to time step $k+1$ is described by a so-called *sytem model*

$$\underline{\boldsymbol{x}}_{k+1} = \underline{a}_k(\underline{\boldsymbol{x}}_k, \underline{\boldsymbol{w}}_k) \ , \tag{2.1}$$

where $\underline{a}_k(\,\cdot\,,\,\cdot\,) : \mathbb{R}^{n_x} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_x}$ is the known nonlinear system function and $\underline{\boldsymbol{w}}_k \in \mathbb{R}^{n_w}$ is zero-mean white noise. For simplicity and brevity, no system input is assumed, as the system input is not a subject of control throughout this thesis. The effect of a system input is implicitly given by the time-variance of the system function. The random vectors $\underline{\boldsymbol{x}}_k$ and $\underline{\boldsymbol{w}}_k$ are characterized by the probability density functions $f_k^x(\underline{x}_k)$ and $f_k^w(\underline{w}_k)$, respectively. Furthermore, the Markov property that $\underline{\boldsymbol{x}}_{k+1}$ only depends on $\underline{\boldsymbol{x}}_k$ is assumed.

Example 2.1: System Model for Target Tracking
The dynamic behavior of the mobile target in tracking scenarios (see Section 1.2) is often described by the so-called constant velocity system model [123], which is given by the linear Gaussian system

$$\underline{\boldsymbol{x}}_{k+1} = \mathbf{A} \cdot \underline{\boldsymbol{x}}_k + \underline{\boldsymbol{w}}_k \ .$$

The system state $\underline{\boldsymbol{x}}_k = [\boldsymbol{x}_k, \dot{\boldsymbol{x}}_k, \boldsymbol{y}_k, \dot{\boldsymbol{y}}_k]^{\mathrm{T}}$ of the target comprises the two-dimensional position $[\boldsymbol{x}_k, \boldsymbol{y}_k]^{\mathrm{T}}$ and the velocities $[\dot{\boldsymbol{x}}_k, \dot{\boldsymbol{y}}_k]^{\mathrm{T}}$ in x and y direction. The system matrix and the covariance matrix of the zero-mean Gaussian white noise $\underline{\boldsymbol{w}}_k$ are

$$\mathbf{A} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \ , \ \mathbf{C}_k^w = q \cdot \begin{bmatrix} \frac{T^3}{3} & \frac{T^2}{2} & 0 & 0 \\ \frac{T^2}{2} & T & 0 & 0 \\ 0 & 0 & \frac{T^3}{3} & \frac{T^2}{2} \\ 0 & 0 & \frac{T^2}{2} & T \end{bmatrix} \ ,$$

respectively, where T is the sampling interval and q is the scalar diffusion strength. ■

2.1.2 Sensor Model

Observations made by sensors provide information on the current system state. For a single sensor $i \in \{1, 2, \ldots, S\}$, the probabilistic *sensor model*

$$\underline{\boldsymbol{z}}_k^i = \underline{h}_k^i(\underline{\boldsymbol{x}}_k, \underline{u}_k^i, \underline{\boldsymbol{v}}_k^i) \tag{2.2}$$

relates measurements $\underline{\boldsymbol{z}}_k^i \in \mathbb{R}^{n_y^i}$ to the state $\underline{\boldsymbol{x}}_k$, where the actual measurement value $\hat{\underline{z}}_k^i$ is a realization of $\underline{\boldsymbol{z}}_k^i$. The sensor model further consists of the known nonlinear measurement function $\underline{h}_k^i(\,\cdot\,,\,\cdot\,) : \mathbb{R}^{n_x} \times \mathcal{U}^i \times \mathbb{R}^{n_v^i} \to \mathbb{R}^{n_y^i}$ and the zero-mean white measurement noise $\underline{\boldsymbol{v}}_k^u \in \mathbb{R}^{n_v^u}$ with density function $f_k^{v,i}(\underline{v}_k^i)$. Here, it is assumed that for $i \neq j$, $\underline{\boldsymbol{v}}_k^i$ and $\underline{\boldsymbol{v}}_k^j$ are stochastically independent.

For the senor management problems considered here, it is assumed that the sensors offer several sensing modalities. At time step k, a modality for sensor i can be selected by means of a so-called *configuration variable* $\underline{u}_k^i \in \mathcal{U}^i$, which takes values from a finite set. For the case of managing many sensors simultaneously, the configuration vector $\underline{u}_k = [(\underline{u}_k^1)^{\mathrm{T}}, (\underline{u}_k^2)^{\mathrm{T}}, \ldots, (\underline{u}_k^S)^{\mathrm{T}}]^{\mathrm{T}}$ comprises the single sensor configuration variables and also takes values from a finite set denoted by

$$\underline{u}_k \in \mathcal{U} = \mathcal{U}^1 \times \mathcal{U}^2 \times \cdots \times \mathcal{U}^S = \{\underline{u}^{(1)}, \underline{u}^{(2)}, \ldots, \underline{u}^{(|\mathcal{U}|)}\} \ .$$

Accordingly, the probabilistic sensor model comprising all sensors is defined as

$$\underline{\boldsymbol{z}}_k = \begin{bmatrix} \underline{\boldsymbol{z}}_k^1 \\ \underline{\boldsymbol{z}}_k^2 \\ \vdots \\ \underline{\boldsymbol{z}}_k^S \end{bmatrix} = \begin{bmatrix} \underline{h}_k^1(\underline{\boldsymbol{x}}_k, \underline{u}_k^1, \underline{\boldsymbol{v}}_k^1) \\ \underline{h}_k^2(\underline{\boldsymbol{x}}_k, \underline{u}_k^2, \underline{\boldsymbol{v}}_k^2) \\ \vdots \\ \underline{h}_k^S(\underline{\boldsymbol{x}}_k, \underline{u}_k^S, \underline{\boldsymbol{v}}_k^S) \end{bmatrix} =: \underline{h}_k(\underline{\boldsymbol{x}}_k, \underline{u}_k, \underline{\boldsymbol{v}}_k) \ . \tag{2.3}$$

Thus, the vectors $\underline{\boldsymbol{z}}_k$, $\underline{h}_k(\,\cdot\,,\,\cdot\,,\,\cdot\,)$, and $\underline{\boldsymbol{v}}_k$ are obtained by concatenating the corresponding single sensor components.

By means of the configuration vector $\underline{u}_k$, it is for example possible to select among several positions or orientations of a sensor, to select the focal distance in case of a camera sensor, or to select a sensor from a sensor network for performing the next measurement. As the set of configuration vectors $\mathcal{U}$ is finite, selecting from various configurations is equivalent to selecting from various sensor models.

Example 2.2: Mobile Sensors
For the target tracking application considered in Example 2.1, distance measurements or bearing measurements from mobile sensors can be utilized in order to track the state of the target (see the mobile sensor control problem introduced in Section 1.2.2). Here, the nonlinear sensor model is given by

$$\boldsymbol{z}_k = \sqrt{(\boldsymbol{x}_k - x_k^s)^2 + (\boldsymbol{y}_k - y_k^s)^2} + \boldsymbol{v}_k \ , \tag{2.4}$$

in case of a mobile distance sensor and

$$\boldsymbol{z}_k = \arctan\left(\frac{\boldsymbol{y}_k - y_k^s}{\boldsymbol{x}_k - x_k^s}\right) + \boldsymbol{v}_k \ , \tag{2.5}$$

in case of a mobile bearing sensor. The (deterministic) position of the sensor $\underline{u}_k = [x_k^s, y_k^s]^{\mathrm{T}}$ can be varied by means of the configuration variable. Since $\underline{u}_k$ is from a finite set, only a finite number of sensor positions can be chosen at time step k. ■

Example 2.3: Sensor Scheduling
For the sensor scheduling problem introduced in Section 1.2.1, the configuration vector is a scalar variable u_k and the set $\mathcal{U} = \{1, 2, \ldots, S\}$ contains the indices of the sensor models (2.2). Thus, for the index u_k of the selected sensor for time step k, the sensor model (2.3) can be reformulated to

$$\underline{\boldsymbol{z}}_k = \underline{\boldsymbol{z}}_k^{u_k} = \underline{h}_k^{u_k}(\underline{\boldsymbol{x}}_k, \underline{\boldsymbol{v}}_k^{u_k}) \ . \qquad ■$$

According to the notation for the sensor scheduling problem, in this thesis the shorthand notation $\underline{x}_k^{\underline{u}_k} := \underline{x}_k(\underline{u}_k)$ is sometimes used for a quantity $\underline{x}_k(\,\cdot\,)$ that depends on the configuration vector $\underline{u}_k$.

2.1.3 Sensor State

For many sensor management problems, the selection of a configuration $\underline{u}_k$ not only affects the sensor model for the current time step k, but also for all future time steps. Here, the sensor model possesses some kind of internal memory or *sensor state* and sensor management problems for those sensor models are called *state-dependent*. In contrast to the system state, the sensor state is assumed to be a deterministic quantity.

Example 2.4: Mobile Sensors (continued)
Consider again the mobile sensors for target tracking described in Example 2.2. Contrary to the previous example, the set of configurations $\mathcal{U}$ does not contain potential sensor positions $[x_k^s, y_k^s]^{\mathrm{T}}$. Instead, $\mathcal{U}$ comprises possible steering angles for the sensor. The position $[x_k^s, y_k^s]^{\mathrm{T}}$ and the orientation ϕ_k^s of the mobile sensor is changed by applying the steering angle $u \in \mathcal{U}$ according to the kinematic model

$$\begin{bmatrix} x_k^s \\ y_k^s \\ \phi_k^s \end{bmatrix} = \begin{bmatrix} x_{k-1}^s \\ y_{k-1}^s \\ \phi_{k-1}^s \end{bmatrix} + \begin{bmatrix} v \cdot \cos(\phi_{k-1}^s + u) \\ v \cdot \sin(\phi_{k-1}^s + u) \\ u \end{bmatrix} , \tag{2.6}$$

where v is the known velocity of the sensor. Here, $[x_k^s, y_k^s, \phi_k^s]^{\mathrm{T}}$ represents the sensor state that is permanently affected by the configuration u. As described in Section 1.2.2, selecting an appropriate steering angle u for such a mobile sensor corresponds to a state-dependent sensor management problem. ■

If not stated elsewhere, the thesis is concerned with state-dependent problems. The opposite case, where a configuration $\underline{u}_k$ affects the sensor model merely for time step k, the management problem and the sensor model are referred to as *stateless*. The sensor scheduling problem considered in Example 2.3 is a typical example for a stateless sensor management problem.

2.2 Sensor Management

Formally, sensor management is aimed at determining a time sequence of configurations for a set of sensors in order to achieve a specific objective, e.g., to gain maximum information about the observed system. In this thesis, the sensor manager calculates the optimal configuration $\underline{u}_k^*$ for the current time step k considering the following N time steps. The optimal configuration is applied to the sensors, i.e., the sensors perform the measurement given the new configuration, and the resulting measurement value $\hat{\underline{z}}_k$ is used for updating the state estimate by means of the Bayesian estimator. Afterwards, a new sensor management process is initiated. This procedure is depicted in Figure 2.1.

2.2.1 Model Predictive Control

From a control perspective, the sensor manager repeatedly determines the optimal configuration sequence $\underline{u}_{k,0:N-1}^*$ by solving an optimal control problem over a finite N-step time horizon. As sensor management controls the information gathering process about the observed system, the optimal sequence $\underline{u}_{k,0:N-1}^*$ corresponds to an optimal sequence of state estimates $\underline{\boldsymbol{x}}_{k,0:N}^*$, which comprises the lowest amount of uncertainty about the true system state.

For clarifying the notation, the first time index k denotes the current time step, while the second time index $n \in \{0, 1, \ldots, N\}$ indicates time steps within the optimization horizon. The first element $\underline{u}_k^* = \underline{u}_{k,0}^*$ of the optimal configuration sequence $\underline{u}_{k,0:N-1}^*$ is applied to the sensors. As the optimization or updating of the configuration sequence is repeated at the next time step $k+1$ based on the new information obtained from the system given in form of a new measurement value $\hat{\underline{z}}_k$, this management procedure corresponds to the so-called *model predictive control* paradigm[1]. A detailed description of different control schemes and the relation to the considered model predictive control paradigm are given in Section 2.4.2.

1 Alternatively: moving, receding, or rolling horizon control

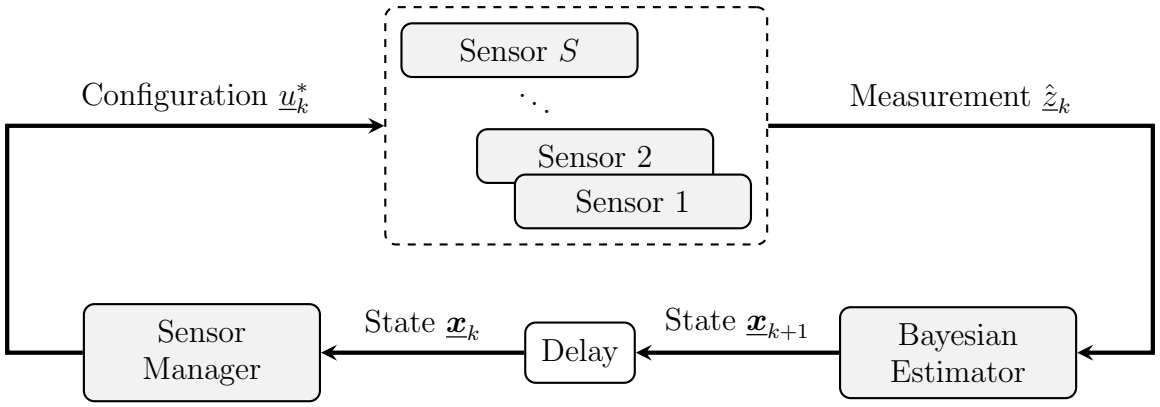


Figure 2.1: General sensor management framework.

2.2.2 Degrees of Freedom

According to Figure 2.1, the proposed probabilistic sensor management framework consists of two main components: 1. the sensor manager, and 2. the Bayesian estimator for state inference. While the set of sensors is typically predetermined by the application, sensor manager and estimator for inference can be adjusted in order to meet given constraints on computational, energy, and memory resources of the sensor nodes, required estimation accuracy, or implementation complexity. The sensor manager offers four basic parameters or degrees of freedom for adjustment:

Length of Time Horizon The management process for determining the optimal configuration $\underline{u}_k^*$ may be *myopic* (one-step time horizon) or *non-myopic* (long-term planning). In many situations, long-term planning will provide better configurations than myopic management, especially, where the dynamics of the observed system or the gain of sensors change predictably [42, 107]. The better estimation performance of a non-myopic manager is at the expense of an increased computational burden and memory requirements, which grow exponentially with the length of the time horizon.

Type of Optimization If the manager anticipates and utilizes future information about the system state within the time horizon, *closed-loop control* is applied. In contrast to this, *open-loop control* only constructs a plan based on the current system state without anticipating future state information (see Section 2.4). Due to this difference in utilizing state information, closed-loop control typically outperforms open-loop control. However, a sensor manager employing closed-loop control is of higher complexity.

Estimator for Planning The sensor manager itself employs a state estimator in order to determine the next configuration. Especially for long-term sensor management, this estimator has to be executed multiple times. Thus, the more accurate and thus complex the employed estimator, the larger the required resources and the better the estimation performance.

Here, it is important to note that the estimator employed for sensor management not necessarily has to be of the same type or the same complexity as the one used for state inference given the actual measurement value $\hat{\underline{z}}_k$. A practical way to trade off estimation accuracy against resource consumption is to employ an accurate estimator like a particle filter [9, 54] or the hybrid density filter (HDF, see Chapter 6) for state inference, while a less complex estimator like a Gaussian estimator (see Chapter 5) or a HDF with fewer components is utilized for management purposes.

Objective Function By means of the objective function, the sensor manager quantifies the utility of a particular configuration (myopic) or sequence of configurations (non-myopic). Two

different function classes are considered throughout this thesis: *covariance-based* objective functions employ scalar functions of the state covariance matrix, while *information theoretic* objective functions are based on the differential entropy (see Section 2.5). Information theoretic objectives generally consider sufficient statistics of the system state, which are given by the probability density function of the state. This leads to a more demanding design of the sensor manager.

2.2.3 Proposed Realizations

Altogether, the proposed sensor management framework offers various combinations for adjustment to a given application. However, these parameters are not fully orthogonal and for some scenarios, parameters are not arbitrary. Employing a Gaussian estimator for planning for example requires the use of covariance-based objective functions, while for linear Gaussian systems, closed-loop control and open-loop control result in exactly the same planning results.

In this thesis, efficient algorithms for two specific parameter settings are proposed (see Table 2.1). The first realization *quasi-linear sensor management* presented in Chapter 3 is devoted to very resource-efficient open-loop model predictive control based computation of configurations, where a Gaussian estimator is employed during planning. Due to neglecting higher-order information about the system state, the complexity of the resulting sensor manager can be drastically reduced. Thus, this realization is ideally suited for computationally constrained sensor nodes.

On the other hand, the realization *information theoretic sensor management* presented in Chapter 4 anticipates future measurement values and utilizes full information on the system state, as a closed-loop model predictive control scheme together with an information theoretic objective function and Gaussian mixtures for characterizing the density function of the system state are employed. The resulting manager is indeed more complex as the one presented in Chapter 3, but in exchange it is more appropriate for highly nonlinear and non-Gaussian scenarios and thus, the achievable estimation performance in such scenarios is superior.

These realizations mark the two extremes of the sensor management framework. For designing sensor managers with regard to the remaining parameter combinations, the algorithms and techniques developed for both proposed realizations can be utilized.

Table 2.1: Proposed realizations of the sensor management framework.

	Chapter 3	Chapter 4
Length of time horizon	arbitrary	arbitrary
Type of optimization	open-loop	closed-loop
Estimator for planning	Gaussian	Gaussian mixture
Objective function	covariance-based	information theoretic
Estimator for inference	arbitrary	arbitrary

2.3 State Estimation

In all models described in Section 2.1, the state as well as the noise are described by means of random vectors. Thus, for propagating the state vector over time and for incorporating measurements, the Bayesian estimator framework is used. Since calculating the Bayesian estimate of the state in closed form is impossible in general, an approximate solution is required. In the following, the general Bayesian estimator framework and the approximate state-of-the-art estimators, which are relevant for this thesis, are briefly introduced.

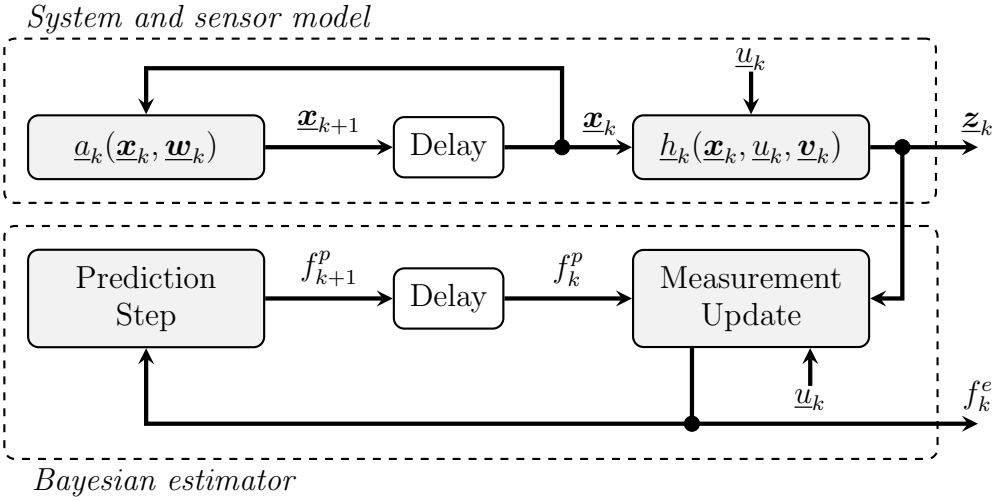


Figure 2.2: Interaction between the system and sensor model and the Bayesian estimator.

2.3.1 Bayesian Estimator

For determining a state estimate of $\underline{\boldsymbol{x}}_k$ at time step k in terms of a probability density function $f^x_k(\underline{x}_k|\hat{\underline{z}}_{0:k}, \underline{u}_{0:k})$ given the measurement sequence $\hat{\underline{z}}_{0:k} = (\hat{\underline{z}}_0, \ldots, \hat{\underline{z}}_k)$ and the configuration sequence $\underline{u}_{0:k} = (\underline{u}_0, \ldots, \underline{u}_k)$, two steps have to be performed alternately in a Bayesian setting, namely the *prediction step* and the *measurement update* (see Figure 2.2). Here, it is assumed that an initial probability density function $f^x_0(\underline{x}_0)$ at time step $k = 0$ is available.

Prediction step

The prediction step of the Bayesian estimator employs (2.1) and the prior density $f^x_k(\underline{x}_k|\hat{\underline{z}}_{0:k}, \underline{u}_{0:k})$ at time step k for recursively propagating the state estimate in time. It is described according to the Chapman-Kolmogorov equation (see for example [165]) and results in a predicted density

$$f^p_{k+1}(\underline{x}_{k+1}) := f^x_{k+1}(\underline{x}_{k+1}|\hat{\underline{z}}_{0:k}, \underline{u}_{0:k}) = \int_{\mathbb{R}^{n_x}} f^T_k(\underline{x}_{k+1}|\underline{x}_k) \cdot f^x_k(\underline{x}_k|\hat{\underline{z}}_{0:k}, \underline{u}_{0:k}) \,\mathrm{d}\underline{x}_k \tag{2.7}$$

characterizing the predicted state $\underline{\boldsymbol{x}}^p_{k+1}$, i.e., $\underline{\boldsymbol{x}}^p_{k+1}$ is the state estimate of $\underline{\boldsymbol{x}}_{k+1}$ for time $k+1$ incorporating the configuration sequence $\underline{u}_{0:k}$ and the measurement sequence $\hat{\underline{z}}_{0:k}$. In (2.7), $f^T_k(\underline{x}_{k+1}|\underline{x}_k)$ is the transition density

$$f^T_k(\underline{x}_{k+1}|\underline{x}_k) = \int_{\mathbb{R}^{n_w}} \delta(\underline{x}_{k+1} - \underline{a}_k(\underline{x}_k, \underline{w}_k)) \cdot f^w_k(\underline{w}_k) \,\mathrm{d}\underline{w}_k \tag{2.8}$$

depending on the system model in (2.1) and the density $f^w_k(\underline{w}_k)$ of the noise $\underline{\boldsymbol{w}}_k$. Typically, the posterior density is used in (2.7), i.e., $f^x_k(\underline{x}_k) = f^e_k(\underline{x}_k)$.

Measurement Update

The *posterior density* $f^e_k(\underline{x}_k)$ characterizing the posterior state estimate $\underline{\boldsymbol{x}}^e_k$ is itself updated by considering (2.3) and applying Bayes' law [165] according to

$$f^e_k(\underline{x}_k) := f^x_k(\underline{x}_k|\hat{\underline{z}}_{0:k}, \underline{u}_{0:k}) = c_k \cdot f^L_k(\hat{\underline{z}}_k|\underline{x}_k, \underline{u}_k) \cdot f^p_k(\underline{x}_k) \;, \tag{2.9}$$

where $c_k = 1/\int f^L_k(\hat{\underline{z}}_k|\underline{x}_k, \underline{u}_k) \cdot f^p_k(\underline{x}_k) \,\mathrm{d}\underline{x}_k$ is a normalization constant and $f^L_k(\hat{\underline{z}}_k|\underline{x}_k, \underline{u}_k)$ is the so-called *likelihood* for a given measurement $\hat{\underline{z}}_k$. The likelihood is derived from the conditional

density

$$f_k(\underline{z}_k|\underline{x}_k,\underline{u}_k) = \int_{\mathbb{R}^{n_v}} \delta(\underline{z}_k - \underline{h}_k(\underline{x}_k,\underline{u}_k,\underline{v}_k)) \cdot f_k^v(\underline{v}_k)\,\mathrm{d}\underline{v}_k \tag{2.10}$$

that gives the probability for the occurrence of a measurement $\underline{\boldsymbol{z}}_k$ given the sensor model (2.3), the density $f_k^v(\underline{v}_k)$ of the noise $\underline{\boldsymbol{v}}_k$, the current state estimate $\underline{\boldsymbol{x}}_k$ and the sensor configuration $\underline{u}_k$, i.e., $f_k(\underline{z}_k|\underline{x}_k,\underline{u}_k)$ can be interpreted as the aggregation of all possible likelihoods.

Recursive Estimation

The predicted and posterior densities resulting from alternately applying the two steps of the Bayesian estimator contain all knowledge of the history, i.e., only the density of the current estimate has to be stored, while all measurement values and previously calculated estimates can be discarded. This allows recursively performing state estimation on the basis of the current estimate. However, recursive Bayesian estimation for nonlinear systems is of conceptual value only, since the complex shapes of the transition density and likelihood prevent a closed-form and efficient solution.

Exceptions include the case of linear systems with Gaussian random variables for which the Kalman filter provides exact solutions in an efficient manner [96]. For the case of nonlinear systems with arbitrarily distributed random variables, there exists no analytic density that can be calculated without changing the type of representation in general. To overcome this problem, an appropriate approximation is inevitable. The well-known extended Kalman filter uses linearization to apply the Kalman filter equations to nonlinear systems [169, 172], while linear regression Kalman filters offer higher-order accuracy by using a deterministic sampling approach [114, 115].

All these Kalman filter based estimators provide estimates of the system state in terms of mean vectors and covariance matrices. An equivalent representation of the estimate would be a Gaussian density function, whereby these estimators can be considered as Gaussian estimators. This consideration is justified by the fact that given just the first two moments mean and covariance, a Gaussian density is the Kullback-Leibler divergence minimizing or entropy maximizing density function for approximating the true density function [35]. Since these Gaussian estimators are used in Section 3 for planning purposes, brief descriptions are provided in the next sections. Non-Gaussian estimators are treated in Section 6.

2.3.2 Kalman Filter

For the important special case of linear dynamic systems with Gaussian random variables, the optimal Bayesian estimate can be calculated in closed form. Here, the temporal behavior of the observed system is described by the linear discrete-time probabilistic model

$$\underline{\boldsymbol{x}}_{k+1} = \mathbf{A}_k \cdot \underline{\boldsymbol{x}}_k + \mathbf{B}_k \cdot \underline{\boldsymbol{w}}_k \ , \tag{2.11}$$

where $\mathbf{A}_k \in \mathbb{R}^{n_x \times n_x}$ and $\mathbf{B}_k \in \mathbb{R}^{n_x \times n_w}$ are real-valued matrices, $\underline{\boldsymbol{w}}_k$ is white Gaussian noise with mean $\underline{\hat{w}}_k$ and covariance matrix $\mathbf{C}_k^w$, and the initial state vector $\underline{\boldsymbol{x}}_0$ is also Gaussian with mean $\underline{\hat{x}}_0$ and covariance matrix $\mathbf{C}_0^x$.

A sensor is described by the linear discrete-time sensor model

$$\underline{\hat{z}}_k = \mathbf{H}_k^{\underline{u}_k} \cdot \underline{\boldsymbol{x}}_k + \underline{\boldsymbol{v}}_k \ , \tag{2.12}$$

where $\mathbf{H}_k^{\underline{u}_k} = \mathbf{H}_k(\underline{u}_k) \in \mathbb{R}^{n_z \times n_x}$ is the real-valued time-variant measurement matrix, and $\underline{\boldsymbol{v}}_k$ is zero-mean white Gaussian noise with positive definite covariance matrix $\mathbf{C}_k^v$ affecting the sensor.

Thanks to the fact that the transition density (2.8) is Gaussian for a specific value of $\underline{x}_{k+1}$, the solution of the Bayesian prediction step leads directly to the prediction step of the Kalman filter

$$\begin{aligned}\hat{\underline{x}}_{k+1}^{p} &= \mathbf{A}_k \cdot \hat{\underline{x}}_k^{e} + \mathbf{B}_k \cdot \hat{\underline{w}}_k \ , \\ \mathbf{C}_{k+1}^{p} &= \mathbf{A}_k \mathbf{C}_k^{e} \mathbf{A}_k^{\mathrm{T}} + \mathbf{B}_k \mathbf{C}_k^{w} \mathbf{B}_k^{\mathrm{T}} \ ,\end{aligned} \tag{2.13}$$

where $\hat{\underline{x}}_{k+1}^{p}$ is the mean vector of the state estimate $\underline{\boldsymbol{x}}_{k+1}^{p}$ at time step $k+1$ conditioned on the measurement values up to time k, and $\mathbf{C}_{k+1}^{p}$ is the corresponding covariance matrix. Since the likelihood (2.10) is Gaussian due to the additive Gaussian measurement noise $\underline{\boldsymbol{v}}_k^{i}$, the optimal solution of the Bayesian estimator leads to the measurement update of the Kalman filter according to [156]

$$\hat{\underline{x}}_k^{e} = \hat{\underline{x}}_k^{p} + \mathbf{K}_k^{\underline{u}_k} \cdot \left(\hat{\underline{z}}_k - \mathbf{H}_k^{\underline{u}_k} \cdot \hat{\underline{x}}_k^{p} \right) \ , \tag{2.14}$$

$$\mathbf{C}_k^{e} = \mathbf{C}_k^{p} - \mathbf{K}_k^{\underline{u}_k} \mathbf{H}_k^{\underline{u}_k} \mathbf{C}_k^{p} \ , \tag{2.15}$$

with Kalman gain

$$\mathbf{K}_k^{\underline{u}_k} = \mathbf{C}_k^{p} \left(\mathbf{H}_k^{\underline{u}_k} \right)^{\mathrm{T}} \left(\mathbf{H}_k^{\underline{u}_k} \mathbf{C}_k^{p} \left(\mathbf{H}_k^{\underline{u}_k} \right)^{\mathrm{T}} + \mathbf{C}_k^{v} \right)^{-1} \ .$$

Hence, the mean vector $\hat{\underline{x}}_k^{e}$ of the state estimate $\underline{\boldsymbol{x}}_k^{e}$ results from $\hat{\underline{x}}_k^{p}$ by incorporating the measurement value $\hat{\underline{z}}_k$, while $\mathbf{C}_k^{e}$ is the corresponding covariance matrix. In the Gaussian case, the mean (2.14) and covariance matrix (2.15) completely describe the posterior density function. Same is true for the prediction step.

Of special importance for the sensor management problem considered in Section 3 is the combination of the (2.15) with (2.13) resulting in the so-called discrete-time *Riccati equation*

$$\mathbf{C}_{k+1}^{p} = \mathbf{A}_k \mathbf{C}_k^{p} \mathbf{A}_k^{\mathrm{T}} + \mathbf{B}_k \mathbf{C}_k^{w} \mathbf{B}_k^{\mathrm{T}} - \mathbf{A}_k \mathbf{K}_k \mathbf{H}_k^{\underline{u}_k} \mathbf{C}_k^{p} \mathbf{A}_k^{\mathrm{T}} \ . \tag{2.16}$$

It is worth mentioning that (2.16) is invariant to the actual measurement sequence $\hat{\underline{z}}_{0:k} = (\hat{\underline{z}}_0, \ldots, \hat{\underline{z}}_k)$. Accordingly, the covariance matrix can be computed off-line in advance.

2.3.3 Extended Kalman Filter

The Kalman filter provides optimal estimates only for linear Gaussian models. However, the Kalman filter equations are often applied to nonlinear models by means of the extended Kalman filter (EKF). Here, it is assumed that the nonlinear model can be approximated by a linear model through a first-order Taylor-series expansion around a nominal value [169]. For the system model (2.1), the linearized version expanded around $\hat{\underline{x}}_k^{e}$ and $\hat{\underline{w}}_k$ is given by

$$\underline{\boldsymbol{x}}_{k+1} \approx \underline{a}_k(\hat{\underline{x}}_k^{e}, \hat{\underline{w}}_k) + \bar{\mathbf{A}}_k \cdot (\underline{\boldsymbol{x}}_k - \hat{\underline{x}}_k^{e}) + \bar{\mathbf{B}}_k \cdot (\underline{\boldsymbol{w}}_k - \hat{\underline{w}}_k) \ ,$$

where

$$\bar{\mathbf{A}}_k = \left. \frac{\partial \underline{a}_k(\underline{x}_k, \underline{w}_k)}{\partial \underline{x}_k^{\mathrm{T}}} \right|_{\underline{x}_k = \hat{\underline{x}}_k^{e}, \underline{w}_k = \hat{\underline{w}}_k} \ , \ \bar{\mathbf{B}}_k = \left. \frac{\partial \underline{a}_k(\underline{x}_k, \underline{w}_k)}{\partial \underline{w}_k^{\mathrm{T}}} \right|_{\underline{x}_k = \hat{\underline{x}}_k^{e}, \underline{w}_k = \hat{\underline{w}}_k} \ .$$

For the sensor model (2.3), linearization around $\hat{\underline{x}}_k^{p}$ and $\hat{\underline{v}}_k$ leads to a matrix $\bar{\mathbf{H}}_k$. To obtain an approximate solution of the Bayesian equations, only the first two moments, i.e., mean and covariance are calculated. While the mean vector still can be determined by employing the nonlinear models (2.1) and (2.3), the matrices $\bar{\mathbf{A}}_k$, $\bar{\mathbf{B}}_k$, and $\bar{\mathbf{H}}_k$ are used in the Kalman filter equations (2.13) and (2.15) for calculating the covariance. In case of mild nonlinearities

or large noise covariances, the approximation error of the EKF is acceptable. However, for linearization the spread of the state vector is not taken into account. Furthermore, only a first-order Taylor-series expansion of the nonlinear functions is employed, which often leads to the divergence of the EKF for strong nonlinearities [134].

Compared to the Kalman filter, the covariance matrices calculated by the extended Kalman filter now depend on the measurement values since the nominal values used for linearization are variant to the measurement values. Thus, off-line computation is not longer possible.

2.3.4 Linear Regression Kalman Filter

To overcome the flaws of the EKF, linear regression Kalman filters (LRKF) employ deterministic sampling techniques for propagating the mean *and* variance of the state vector through the system and measurement function. This allows linearizing the nonlinear functions by *weighted statistical linear regression* [114, 115], where additionally the covariance of the linearization error is determined and considered for estimation purposes. Thus, LRKFs can provide estimation results with superior accuracy compared to the EKF (see e.g. [129, 183]), while the computational complexity is approximately the same.

More precisely, weighted statistical linear regression calculates a matrix $\mathbf{A}$ and a vector $\underline{b}$ such that

$$\underline{\boldsymbol{y}} = \underline{g}(\underline{\boldsymbol{x}}) \approx \mathbf{A} \cdot \underline{\boldsymbol{x}} + \underline{b} \;, \tag{2.17}$$

where the nonlinear transformation $\underline{\boldsymbol{y}} = \underline{g}(\underline{\boldsymbol{x}})$ substitutionally represents the prediction of the state $\underline{\boldsymbol{x}}_k$ or the measurement $\underline{\boldsymbol{z}}_k$ by means of the nonlinear functions $\underline{a}_k(\,\cdot\,)$ or $\underline{h}_k(\,\cdot\,)$, respectively. For this purpose, $\underline{\boldsymbol{y}} = \underline{g}(\underline{\boldsymbol{x}})$ is evaluated at L regression points $\{\underline{x}_i, \underline{y}_i\}$ with weights ω_i, where $\underline{y}_i = \underline{g}(\underline{x}_i)$ and $\sum_i \omega_i = 1$. Then $\mathbf{A}$, $\underline{b}$ are determined by minimizing the sum of squared errors

$$\{\mathbf{A}, \underline{b}\} = \arg \min_{\mathbf{A},\underline{b}} \sum_{i=1}^{L} \omega_i \cdot (\underline{e}_i)^{\mathrm{T}} \cdot \underline{e}_i \;, \tag{2.18}$$

with $\underline{e}_i = \underline{y}_i - (\mathbf{A}\underline{x}_i + \underline{b})$. The solution of (2.18) is

$$\mathbf{A} = \mathbf{C}_{xy}^{\mathrm{T}} \mathbf{C}_x^{-1} \;,\; \underline{b} = \underline{\hat{y}} - \mathbf{A} \cdot \underline{\hat{x}} \;,$$

where $\underline{\hat{x}} = \sum_i \omega_i \cdot \underline{x}_i$, $\underline{\hat{y}} = \sum_i \omega_i \cdot \underline{y}_i$, $\mathbf{C}_x = \sum_i \omega_i (\underline{x}_i - \underline{\hat{x}})(\underline{x}_i - \underline{\hat{x}})^{\mathrm{T}}$, and $\mathbf{C}_{xy} = \sum_i \omega_i (\underline{x}_i - \underline{\hat{x}})(\underline{y}_i - \underline{\hat{y}})^{\mathrm{T}}$. The linearization error characterized by the deviations $\underline{e}_i$ has zero-mean and covariance matrix

$$\mathbf{C}_e = \mathbf{C}_y - \mathbf{A}\mathbf{C}_x\mathbf{A}^{\mathrm{T}} \;.$$

and acts as additional noise source [181].

For a statistical linearization of (2.1) and (2.3), the idea is to use augmented vectors $\underline{\boldsymbol{X}}_k = [\underline{\boldsymbol{x}}_k^{\mathrm{T}}, \underline{\boldsymbol{w}}_k^{\mathrm{T}}, \underline{\boldsymbol{v}}_k^{\mathrm{T}}]^{\mathrm{T}}$ comprising the state and the noise. Several estimators developed in the recent years are based on this idea and thus can be classified as linear regression Kalman filters. These estimators differ in the number of used regression points and the way these points are chosen. The most popular linear regression Kalman filter is the unscented Kalman filter (UKF), which is based on the so-called unscented transform [91, 92][2]. Further LRKFs are the central differences filter [160], divided differences filter [137], or the Gauss-Hermite filter [7, 85]. The Gaussian estimator proposed in Section 5 can also be considered as LRKF.

2 A Matlab implementation of the UKF is available at [50].

2.4 Stochastic Control

The main task of the considered sensor management problem is the sequential selection of configurations, where planning over a finite time horizon with length N, i.e., planning N time steps ahead, is involved. Such sequential decision problems under uncertainty are covered by the theory of *Markov decision processes* (MDPs). The primary assumption of MDPs is that the measurements provide enough information for exactly determining the system state. This assumptions does not hold for the sensor management framework considered here. Due to the noise-corrupted nonlinear sensor model (2.3), inferring the system state by means of actual measurement values leaves residual uncertainty concerning the system state. Problems of this type are covered by *partially observable Markov decision processes* (POMDPs), which are a special case of MDPs [56, 10].

2.4.1 POMDP

A partially observable Markov decision process consists of

- the time index $k \in \{0, 1, \ldots, K-1\}$, with K being the length of the observation period,
- the state $\underline{\boldsymbol{x}}_k$ of the dynamic system,
- a set $\mathcal{U}$ of controls $\underline{u}_k \in \mathcal{U}$,
- a real-valued single step objective functions $g_k(\underline{\boldsymbol{x}}_k, \underline{u}_k)$ to be minimized (or maximized), and
- the transition density $f_k^T(\underline{x}_{k+1}|\underline{x}_k)$ and likelihood $f_k^L(\hat{\underline{z}}_k|\underline{x}_k, \underline{u}_k)$.

This formulation coincides to the closed-loop control scheme, where the controller anticipates and utilizes future information. It can be reformulated as a fully observed MDP by introducing the so-called information set $\mathcal{I}_k = \{\underline{u}_0, \hat{\underline{z}}_0, \ldots, \underline{u}_{k-1}, \hat{\underline{z}}_{k-1}\}$ [34]. This set comprises the history of all applied controls and the resulting measurement values. The solution of a POMDP comes in the form of a policy

$$\underline{u}_k = \underline{\mu}_k(\underline{\boldsymbol{x}}_k) \ , \tag{2.19}$$

i.e., a control law that specifies which control variable should be applied given the system state at a particular time step. Here, it is assumed that $\underline{\boldsymbol{x}}_k$ is characterized by a *sufficient statistic* that subsumes all available information on $\underline{\boldsymbol{x}}_k$ at time step k, i.e., the sufficient statistic represents an estimate of the state based on the initial state $\underline{\boldsymbol{x}}_0$ and the information set $\mathcal{I}_k$. Due to the Markov assumption, the probability density function $\underline{\boldsymbol{x}}_k \sim f_k^x(\underline{x}_k|\mathcal{I}_k)$ is a sufficient statistic for the entire information set [27]. Thus, merely the density function at time step k needs to be stored.

Based on the system model (2.1), the sensor model (2.3) and the policy (2.19), the *cumulative objective* results from the backward recursion

$$J_k(\underline{\boldsymbol{x}}_k) = \min_{\underline{\mu}_k} \left\{ g_k(\underline{\boldsymbol{x}}_k, \underline{\mu}_k(\underline{\boldsymbol{x}}_k)) + \mathrm{E}_{\underline{z}_k} \left\{ J_{k+1}(\underline{\boldsymbol{x}}_{k+1}) | \underline{\boldsymbol{x}}_k, \underline{\boldsymbol{z}}_k, \underline{\mu}_k(\underline{\boldsymbol{x}}_k) \right\} \right\} \tag{2.20}$$

commencing from the terminal objective $J_{K-1}(\underline{\boldsymbol{x}}_{K-1}) = \min_{\underline{\mu}_{K-1}} g_{K-1}(\underline{\boldsymbol{x}}_{K-1}, \underline{\mu}_{K-1}(\underline{\boldsymbol{x}}_{K-1}))$. This recursion is also known as Bellman's equation [16, 17] and the summand $\mathrm{E}_{\underline{z}_k}\{\,\cdot\,\}$ is often referred to as *value-to-go*. According to this, the optimal policy satisfies

$$\underline{u}_k^* = \underline{\mu}_k^*(\underline{\boldsymbol{x}}_k) = \arg \min_{\underline{\mu}_k} \left\{ g_k(\underline{\boldsymbol{x}}_k, \underline{\mu}_k(\underline{\boldsymbol{x}}_k)) + \mathrm{E}_{\underline{z}_k} \left\{ J_{k+1}(\underline{\boldsymbol{x}}_{k+1}) | \underline{\boldsymbol{x}}_k, \underline{\boldsymbol{z}}_k, \underline{\mu}_k(\underline{\boldsymbol{x}}_k) \right\} \right\} \ .$$

Under certain assumptions, it is possible to prove that sequential decision processes converge, i.e., the uncertainty of the state decreases over time [51]. However, solving POMDPs even for discrete states and discrete measurement values is PSPACE-complete[3], as the size of $\mathcal{I}_k$ grows rapidly with the number of measurement values and control values [140]. In case of continuous-valued states, solving POMDPs is intractable in general. A famous exception is LQG[4], where the separation principle leads to a decomposition into an estimation part and a control part [18, 21]. For an survey of POMDP solution methods see [77]. Related fields to stochastic control and POMDPs are reinforcement learning [93, 175] and experimental design [25, 147].

2.4.2 Closed-loop vs. Open-loop Control

The primary assumption of POMDPs is that state feedback is employed, i.e., the information about the system state is revealed to the controller and the optimal policy utilizes that new information as it becomes available. This procedure corresponds to a *closed-loop control* scheme. In many practical applications, however, the total time horizon that has to be considered by the stochastic controller, i.e., the length K of observation period of the system, may be too long. Employing *open-loop control* instead, where the primary assumption of POMDPs is not made, leads to an approximate procedure, where the optimal plan (rather than the policy as in the closed-loop case) can often be found with a significantly lower computational demand. Admittedly, no state feedback is employed.

As an compromise between open-loop control and closed-loop control, *open-loop feedback control* (OLFC) can be used alternatively. Like in open-loop control, no future information is anticipated. But when a new measurement value becomes available, it is used for calculating an updated plan, i.e., the open-loop feedback controller first determines the configuration sequence for the observation period in an open-loop fashion, executes one or more steps of the sequence, and then calculates a new sequence that incoporates the newly received measurement value. It can be shown, that OLFC is no worse than open-loop control, but the deviation from closed-loop control can be arbitrarily large [21]. The plans, however, are still constructed considering the total length K of the observation period.

To avoid the computational burden of optimizing over long time horizons, a model predictive control scheme is employed for the proposed sensor management framework instead. As stated in Section 2.2, the sensor manager repeatedly determines the optimal configuration sequence over a moving or rolling horizon. The length N of this moving horizon is considered to be much smaller than the observation period K of the system, which may be infinite. Depending on the optimization type, two possible model predictive control schemes arise: at each step of the management problem, the *closed-loop model predictive controller* solves a POMDP for the moving horizon, while the *open-loop model predictive controller* determines a plan for the moving horizon that does not anticipate the availability of future information[5]. Whenever a new measurement value is received, both control schemes utilize this new information about the system state by calculating an updated policy and plan, respectively. Obviously, the open-loop model predictive controller provides an approximate solution to the closed-loop model predictive control problem. The computations, however, are typically less complicated since no expectations with respect to uncertain measurements are necessary.

3 PSPACE is a superset of NP and is the set of decision problems that can be solved by a Turing machine using a polynomial amount of memory and unlimited computation time.

4 LQG: *l*inear *q*uadratic *G*aussian control, i.e., linear system and sensor model with quadratic objective function and additive Gaussian noise

5 The combination of model predictive control with closed-loop/open-loop control is often also referred to as limited lookahead closed-loop/open-loop control. Open-loop model predictive control can be considered as the combination of OLFC with a limited and moving time horizon.

2.5 Objective Functions

The primary goal of the proposed sensor management framework is the determination of a sequence of configurations in order to gain maximum information about the state of the observed system. For achieving this goal, the objective function $g_k(\underline{\boldsymbol{x}}_k, \underline{u}_k)$ to be minimized (or maximized) by the employed control scheme has to be specified appropriately. In the following, definitions of the objective function are given, which are employed for sensor management throughout this thesis. They can be categorized into two classes of objective functions: covariance-based and information theoretic.

2.5.1 Covariance-based Objective Functions

Covariance-based objective functions depend on the *expected posterior covariance matrix*, which is defined according to

$$\mathbf{C}_k^e(\underline{u}_k) := \mathrm{Cov}\{\underline{\boldsymbol{x}}_k^e|\underline{\boldsymbol{z}}_k, \underline{u}_k\} = \mathrm{E}_{\underline{x}_k^e, \underline{z}_k}\left\{(\underline{x}_k^e - \underline{\hat{x}}_k^e)(\underline{x}_k^e - \underline{\hat{x}}_k^e)^{\mathrm{T}}|\underline{u}_k\right\} \ ,$$

where $\underline{\hat{x}}_k^e$ is the mean vector of posterior state estimate (see Section 2.3.1). The covariance matrix indicates the amount of uncertainty subsumed by the state estimate $\underline{\boldsymbol{x}}_k^e$. By employing the following scalar functions, a quantification of the subsumed uncertainty is possible:

Determinant The determinant-based objective function is defined as

$$g_k(\underline{\boldsymbol{x}}_k, \underline{u}_k) = |\mathbf{C}_k^e(\underline{u}_k)| \ . \tag{2.21}$$

Minimizing the determinant of the posterior state estimate leads to a minimization of the total variance of the estimate, as the determinant of $\mathbf{C}_k^e(\underline{u}_k)$ is proportional to the volume of the covariance ellipsoid [135].

Trace The trace-based objective function is defined as

$$g_k(\underline{\boldsymbol{x}}_k, \underline{u}_k) = \mathrm{trace}(\mathbf{C}_k^e(\underline{u}_k)) \ . \tag{2.22}$$

Minimizing the trace of the expected covariance matrix leads to a minimization of the expected squared estimation error, as $\mathrm{trace}(\mathbf{C}_k^e(\underline{u}_k)) = \mathrm{E}_{\underline{x}_k^e, \underline{z}_k}\left\{(\underline{x}_k^e - \underline{\hat{x}}_k^e)^{\mathrm{T}}(\underline{x}_k^e - \underline{\hat{x}}_k^e)|\underline{u}_k\right\}$. This also corresponds to minimizing the perimeter of the rectangular region enclosing the covariance ellipsoid [135].

Eigenvalue The eigenvalue objective function is defined as

$$g_k(\underline{\boldsymbol{x}}_k, \underline{u}_k) = \lambda_{\max}(\mathbf{C}_k^e(\underline{u}_k)) \ .$$

The maximum eigenvalue $\lambda_{\max}$ of $\mathbf{C}_k^e(\underline{u}_k)$ corresponds to the length of the largest axis of the covariance ellipsoid [135]. Thus, minimizing $\lambda_{\max}$ leads to a minimization of the largest variance of $\underline{\boldsymbol{x}}_k^e$.

Additionally, the expected covariance matrix can be weighted by means of a weighting matrix $\mathbf{W}_k$ according to $\mathbf{W}_k \cdot \mathbf{C}_k^e(\underline{u}_k) \cdot \mathbf{W}_k^{\mathrm{T}}$. This matrix allows adjusting or underrating different units of the state vector, e.g., if the state vector comprises information about position and velocity as in the target tracking Example 2.1. In cases, where the attention is on achieving a minimal terminal covariance $\mathbf{C}_{N-1}^e$, while the values of intermediate covariances are of no importance, one can set $\mathbf{W}_k = \mathbf{0}$ for $k \in \{0, 1, \ldots, N-2\}$.

It is worth mentioning, that in optimum experimental design theory similar scalar objective functions are defined based on the Fisher information matrix (or its inverse) instead of the ones based on the covariance matrix [11, 178]. Minimizing such objective functions corresponds to minimizing a lower bound of the covariance matrix.

2.5.2 Information Theoretic Objective Functions

Shannon's information theory is the foundation for information theoretic objective functions. Instead of considering the covariance matrix of the state estimate for minimizing the uncertainty, the *differential entropy* is employed. The differential entropy extends Shannon's entropy for discrete random variables (see [167]) to continuous random variables and quantifies the uncertainty of the random variable based on the density function. In contrast to covariance-based objective functions, entropy quantifies areas of probabilities and not the average deviation to a single point. It indicates how much additional information is necessary for inferring the exact value from an estimate.

Differential Entropy

The differential entropy of a random vector $\underline{\boldsymbol{x}}$ is defined as

$$H(\underline{\boldsymbol{x}}) = \mathrm{E}_{\underline{x}}\{-\log f(\underline{x})\} = -\int_{\mathbb{R}^{n_x}} f(\underline{x}) \log f(\underline{x}) \,\mathrm{d}\underline{x} \ . \tag{2.23}$$

While the entropy for discrete random variables is always non-negative, the differential entropy may be negative. Of special importance for sensor management is the conditional version of the entropy

$$H(\underline{\boldsymbol{x}}|\underline{\boldsymbol{z}}) = -\int_{\mathbb{R}^{n_z}} f(\underline{z}) \int_{\mathbb{R}^{n_x}} f(\underline{x}|\underline{z}) \log f(\underline{x}|\underline{z}) \,\mathrm{d}\underline{x} \,\mathrm{d}\underline{z} \ . \tag{2.24}$$

Between the entropy (2.23) and its conditional version (2.24), the ordering

$$H(\underline{\boldsymbol{x}}) \geq H(\underline{\boldsymbol{x}}|\underline{\boldsymbol{z}}) \tag{2.25}$$

holds, which means that conditioning on a random vector reduces entropy. By means of the conditional entropy the objective function

$$g_k(\underline{\boldsymbol{x}}_k, \underline{u}_k) = H(\underline{\boldsymbol{x}}_k|\underline{\boldsymbol{z}}_k, \underline{u}_k) \tag{2.26}$$

can be defined. For linear Gaussian systems, minimizing (2.26) is equivalent to minimizing the determinant-based objective function (2.21) since

$$\begin{aligned} H(\underline{\boldsymbol{x}}_k|\underline{\boldsymbol{z}}_k, \underline{u}_k) &= -\int_{\mathbb{R}^{n_z}} f_k^z(\underline{z}_k) \underbrace{\int_{\mathbb{R}^{n_x}} \mathcal{N}(\underline{x}_k; \hat{\underline{x}}_k, \mathbf{C}_k(\underline{u}_k)) \log \mathcal{N}(\underline{x}; \hat{\underline{x}}_k, \mathbf{C}_k(\underline{u}_k)) \,\mathrm{d}\underline{x}_k}_{=-\frac{1}{2}\log((2\pi e)^{n_x}|\mathbf{C}_k(\underline{u}_k)|) \text{ (independent of } \underline{z}_k)} \mathrm{d}\underline{z}_k \\ &= \tfrac{1}{2}\log\left((2\pi e)^{n_x}|\mathbf{C}_k(\underline{u}_k)|\right) \ , \end{aligned}$$

where e is Euler's number.

Mutual Information

An objective function often employed for sensor management (see e.g. [94, 119, 121, 185]) is the mutual information between two random vectors $\underline{\boldsymbol{x}}$ and $\underline{\boldsymbol{z}}$

$$\begin{aligned} I(\underline{\boldsymbol{x}}; \underline{\boldsymbol{z}}) &= \int_{\mathbb{R}^{n_z}} \int_{\mathbb{R}^{n_x}} f(\underline{x}, \underline{z}) \log \frac{f(\underline{x}, \underline{z})}{f(\underline{x}) \cdot f(\underline{z})} \,\mathrm{d}\underline{x} \,\mathrm{d}\underline{z} \\ &= H(\underline{\boldsymbol{x}}) - H(\underline{\boldsymbol{x}}|\underline{\boldsymbol{z}}) \\ &= H(\underline{\boldsymbol{z}}) - H(\underline{\boldsymbol{z}}|\underline{\boldsymbol{x}}) \ , \end{aligned} \tag{2.27}$$

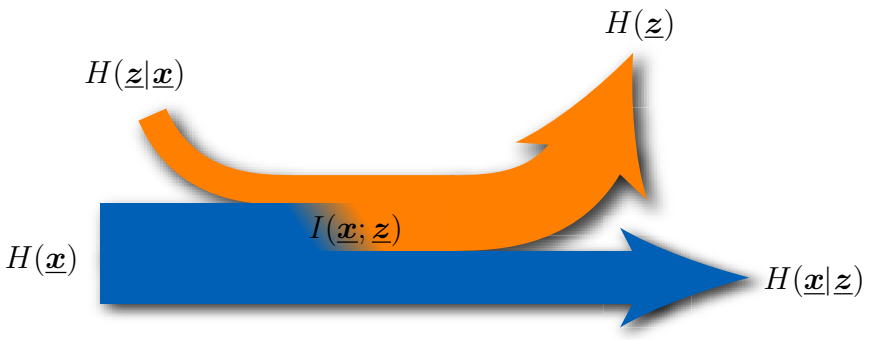


Figure 2.3: Illustration of the flow of entropy (adapted from [205]). The uncertainty $H(\underline{\boldsymbol{x}})$ of $\underline{\boldsymbol{x}}$ consists of the conditional entropy $H(\underline{\boldsymbol{x}}|\underline{\boldsymbol{z}})$ and a portion of the uncertainty of the measurement $\underline{\boldsymbol{z}}$.

which corresponds to the reduction of the entropy in one random vector due to the incorporation of another random vector. The last two lines follow directly from the definition of the mutual information [45] and are illustrated in Figure 2.3. By the chain rule for mutual information

$$I(\underline{\boldsymbol{x}};\underline{\boldsymbol{z}}_1,\dots\underline{\boldsymbol{z}}_n) = \sum_{i=1}^{n} I(\underline{\boldsymbol{x}};\underline{\boldsymbol{z}}_i|\underline{\boldsymbol{z}}_1,\dots,\underline{\boldsymbol{z}}_{i-1}) \ , \tag{2.28}$$

an expansion into a sum of mutual information terms is possible.

According to [57], minimizing the conditional entropy is equivalent to maximizing the mutual information between the system state $\underline{\boldsymbol{x}}_k$ and the measurement $\underline{\boldsymbol{z}}_k$.[6] Thus, the mutual information objective function is given by

$$g_k(\underline{\boldsymbol{x}}_k,\underline{u}_k) = I(\underline{\boldsymbol{x}}_k;\underline{\boldsymbol{z}}_k|\underline{u}_k) = H(\underline{\boldsymbol{x}}_k) - H(\underline{\boldsymbol{x}}_k|\underline{\boldsymbol{z}}_k,\underline{u}_k) = H(\underline{\boldsymbol{z}}_k|\underline{u}_k) - H(\underline{\boldsymbol{z}}_k|\underline{\boldsymbol{x}}_k,\underline{u}_k) \ , \tag{2.29}$$

where the conditioning is on a particular value $\underline{u}_k$ and not on a random vector.

Kullback-Leiber Divergence

Strongly related to entropy and mutual information is the relative entropy or Kullback-Leibler divergence (KLD, [113])

$$D\big(\tilde{f}(\underline{x})||f(\underline{x})\big) = \int_{\mathbb{R}^{n_x}} \tilde{f}(\underline{x}) \log \frac{\tilde{f}(\underline{x})}{f(\underline{x})} \,\mathrm{d}\underline{x} \ \geq \ 0 \tag{2.30}$$

for quantifying the difference between two probability density functions $\tilde{f}(\underline{x})$ and $f(\underline{x})$. Comparing (2.27) with (2.30) yields the relation

$$I(\underline{\boldsymbol{x}};\underline{\boldsymbol{z}}) = D\left(f(\underline{x},\underline{z})||f(\underline{x})\cdot f(\underline{z})\right) = \mathrm{E}_{\underline{z}}\left\{D\left(f(\underline{x}|\underline{\boldsymbol{z}})||f(\underline{x})\right)\right\}$$

between mutual information and KLD. Hence, in context of sensor management, the KLD is employed for calculating the expected difference between the posterior density $f(\underline{x}|\underline{z})$ and the prior density $f(\underline{x})$. Besides employing KLD for sensor management (see e.g. [42]), it is also an important measure for comparing density functions in estimation tasks like Gaussian mixture reduction, Chapter 7, or density estimation [69].

6 For a detailed discussion on minimizing the conditional entropy vs. maximizing the mutual information see Section 4.1.1.

2.6 Related Work

The sensor management problem has been intensively investigated within the last years. This section gives a short overview on the most interesting approaches and strategies. The material is coarsely categorized, where the first category is concerned with approaches for linear models. Approaches for nonlinear models can be found in the remaining categories, which distinguish between the considered time horizon.

2.6.1 Linear System and Sensor Models

One of the first works on sensor management can be found in [126], where control is available for both, system and sensor. Considering linear Gaussian systems and a quadratic cost function, it is shown that a separation principle similar to the basic LQG problem holds, i.e., the configuration of the sensor can be determined independently of the system control policy and independently of the measurements. Furthermore, the optimal sensor configuration results from off-line traversing a search tree. An extension to the continuous-time LQG control problem with discrete-time measurements can be found in [171]. The findings for the discrete-time LQG problem are exposed in [6] from a different viewpoint using conditioning arguments highlighting the independence of the measurement policy from the measurement values. The separation principle still holds for the additional consideration of running measurement costs, which are associated with the requested level of information [194].

Works toward pure sensor management strategies neglecting the control of the dynamic system and employing more adequate objective functions can be found in [37, 88, 119, 138, 148]. In [119], mutual information is chosen as the objective function, which is merely a function of the state and noise covariance in case of linear Gaussian systems. However, as shown in this work, open-loop control and closed-loop control lead to the same configuration sequence. If minimizing the largest eigenvalue of the covariance matrix is the objective, a very efficient management algorithm can be employed based on the information filter formulation of the Kalman filter [138]. The information form is also exploited in [148] for identifying the parameters of a static system observed via a network of correlated sensor nodes. For gaining maximum information even for strongly correlated sensors, an optimal whitening techniques is presented. Based on this, the sensor management process is performed in a myopic and distributed manner. A solution together with some pitfalls of non-myopic sensor management based on covariance-based objective functions and dynamic programming with state space discretization is given in [37], which extends the approach presented in [84]. The sensor selection problem, i.e., choosing the best n-element subset of sensors for observing a static system, is formulated as convex optimization problem in [88]. Even if a relaxation is exploited for approximately but efficiently solving the problem, a bound on the best performance possible can be stated. Various types of covariance-based objectives are discussed and the difference to experimental design is highlighted as well.

In order to reduce the computational demand of off-line performing a tree search for obtaining the optimal sensor configuration, a stochastic algorithm for the sensor scheduling problem is introduced in [67]. Here, the sensors are switched randomly according to an optimal probability distribution for obtaining the best expected steady-state performance. An extension to the general sensor management problem can be found in [68]. For a practical realization of management strategies, the authors in [158] suggest model predictive or moving horizon control and provide conditions for which myopic optimization is optimal.

2.6.2 Myopic or Greedy Approaches

A collection of different objective functions for myopic sensor management is described in detail and compared for a target tracking scenario in [43, 199]. In accordance with [57], it is shown that objective functions based on the expected posterior density function are merely a measure of the predicted density and thus of limited use. More appropriate are objective functions that evaluate the expected information gain as for example mutual information or the expected entropy do. This finding is exploited in [52, 186]. For a reduced computational burden, entropy differences defined over the measurement space are used in [186] for approximating mutual information instead of considering the joint state and measurement space, while in [52] Monte Carlo evaluation of mutual information is employed.

In [12], the estimation performance of mutual information is compared with objective functions based on the expected posterior covariance. While both objective functions provide similar estimation results, the computation time for mutual information is one order of magnitude less the time necessary for the covariance-based function. Furthermore, it is shown that employing linearization of the nonlinear sensor model and the Kalman filter equations leads to an efficient approximation of mutual information, as it becomes nothing else than a measurement independent covariance-based objective function. Only in few scenarios, this approximation leads to a drastically degenerated estimation performance. Based on this linearization approach, in [64, 65] a decentralized sensor management and data fusion scheme is proposed.

For determining the measurement rate or frequency in order to minimize the localization error of a team of mobile robots, [131, 132] also employ linearization based on first-order Taylor-series expansion. The resulting linear system and sensor model is then converted into an equivalent continuous-time model, for which the steady-state covariance of the corresponding continuous-time Riccati equation can be expressed as a function of the measurement frequencies. Its is shown that the optimal frequencies result from a convex optimization problem. This unconventional approach is restricted to sensor management problems for which the continuous-time model is linear and time-invariant.

Myopic sensor management approaches based on the expected Rényi information divergence are presented for example in [75, 108, 111]. The Rényi information divergence is a generalization to the differential entropy, whose emphasis of the tails of the density function can be adjusted by means of a scalar parameter. In [75, 108], algorithms for a discrete set of configurations are presented. An extension to continuous-valued configurations based on the determination of a potential field of expected information gain is presented in [111], while [109] deals with a two-step time horizon using additional simulation.

For objective functions fulfilling the property of submodularity, it is possible to prove that the performance of greedy selection of configurations for static systems and sensors is merely by a constant factor worse than the optimal strategy. Mutual information is one of these specific objective functions. This interesting theoretical result has been applied to the problems of sensor subset selection [103], sensor scheduling [104], or sensor placement selection [105]. Extensions to sensor management problems for dynamic systems can be found in [191, 192].

2.6.3 Non-myopic Sensor Management

For discrete state spaces, solutions to the closed-loop control sensor management problem based on dynamic programming can be found for example in [33, 112]. Especially in [112] it is shown that for piece-wise linear objective functions, the expected objective is also piece-wise linear and convex, which facilitates off-line calculation of the optimal configuration by applying linear programming algorithms for POMDPs. Based on these results, suboptimal algorithms for quadratic objective functions are derived.

As directly solving the closed-loop control problem for sensor management with continuous states is computationally very demanding or even impossible, suboptimal open-loop approaches (besides the previously discussed myopic approaches) are used instead. One class of these suboptimal approaches employ the posterior Cramér-Rao lower bound (PCRLB) for the covariance matrix, which is defined to be the inverse of the Fisher information matrix. As the PCRLB depends on the future system state, the PCRLB is calculated approximately in a predictive manner based on particle filtering [41, 80], a priori and constant estimation of the state [143, 179], or adaptive discretization [79]. As the PCRLB is merely a lower bound, which is above all not always tight [20], the resulting performance can divergence arbitrarily from the closed-loop performance.

For target tracking applications with linear system dynamics and additive measurement noise, [40, 190] present open-loop control approaches, where the nonlinear sensor models are linearized multiple time steps ahead, based on the current system estimate. This facilitates the use of the Kalman filter when determining the next configuration. Optimization criteria include the trace of the covariance matrix [40] or mutual information with communication costs between sensor nodes [190]. An extension presented in [42] employs minimization of the Kullback-Leibler divergence between the predicted and the posterior density as objective, particle filtering for state prediction over the time horizon, and the unscented transform for measurement value prediction. To bound the search complexity for the best configuration sequence, pruning strategies based on breadth-first search, uniform-cost search, and greedy search are also introduced.

A closed-loop model predictive control approach to target tracking based on the expected Rényi divergence is introduced in [107]. Three strategies are presented for incorporating the effect of future measurements. A Monte Carlo rollout strategy for calculating the expected Rényi divergence potentially provides the best estimation performance at the expense of a large computational burden. In order to obtain computationally more tractable solutions, the remaining two strategies provide approximations to the Monte Carlo rollout. The first approximation merely investigates configuration sequences that are most informative by employing an information-directed search. For the second approximation strategy, the value-to-go term in Bellman's equation is replaced by an computationally cheap function that considers the Rényi divergence between myopic gains at the current and a future time step.

One of a few approaches to non-myopic sensor management with continuous configuration space can be found in [170]. In order to determine the best configuration sequence, a simulation-based approximation that use stochastic gradients is presented. This algorithm is applied to a target tracking scenario with bearings-only sensor model. Therefore, convergence of the proposed algorithms are demonstrated theoretically and via simulation.

2.7 Summary

Concluding, it can be stated that calculating optimal configuration sequences for long time horizons in case of continuous-valued states and measurements has not attracted great interest in existing work. The most challenging tasks, however, arise from non-myopic sensor management. These tasks are the anticipation and incorporation of the effect of future sensor measurements on sensor management decisions and the repeated state estimation for nonlinear system and sensor models. Due to the employed model predictive control scheme, potentially long time horizons and large sets of configurations, sensor management decisions and state estimations have to be performed very frequently, which requires computationally efficient algorithms for an overall feasible sensor management. To achieve this, appropriately chosen approximations have to be applied in order to not degrade the benefits of considering long time horizons.

CHAPTER 3

Quasi-linear Sensor Management

Thanks to the independence from actual measurement values, sensor management for linear Gaussian models can be solved by deterministic optimization. The resulting configuration sequence is optimal even in a closed-loop sense. However, the determination of the optimal configuration sequence is computationally demanding, as a tree of depth N has to be exhaustively searched. To deal with this NP-hard problem, a novel optimal pruning algorithm named *information-based pruning* (IBP) is presented in Section 3.2, where complete sub-trees corresponding to inadequate estimation results are pruned as early as possible, while preserving the optimal configuration sequence is guaranteed. In contrast to existing pruning methods that are typically based on branch-and-bound techniques, IBP exploits the information contribution of a specific configuration to the estimation process. This contribution is represented by the so-called sensor information matrix. Based on this matrix, pruning is performed in two ways by

1. exploiting the partial ordering of the configurations, which is implied by the *sensor information matrix* and is preserved by the monotonicity property of the Riccati equation
2. and calculating a so-called *bounding sensor* that provides a novel lower bound to the posterior covariance matrices.

In order to benefit from the nice properties of linear Gaussian sensor management as well as from the effectiveness of the proposed pruning algorithm, this chapter is further devoted to the derivation of an efficient open-loop model predictive control scheme for nonlinear non-Gaussian sensor management problems with covariance-based objective functions. The adaptation to the linear Gaussian case is achieved in two steps. At first, a trajectory of the system state along the N-step time horizon is calculated by recursively propagating a set of regression points through the nonlinear system function. The calculation of this so-called *linearization trajectory* does not rely on the anticipation or incorporation of future measurements, which automatically implies an open-loop control structure. In the second step, statistical linearization about the state trajectory is employed for obtaining a linearized approximation of the nonlinear sensor models.

After solving the now linear Gaussian sensor management problem and applying the first element of the determined configuration sequence, the linearization procedure is repeated for the next time step. Since the linearization procedure is derived independent from a particular calculation scheme for the set of regression points, various linear regression Kalman filters like the Gaussian estimator proposed in Section 5 but also the extended Kalman filter can be employed, depending on the concrete application and required accuracy.

The novel open-loop model predictive sensor management approach named *quasi-linear sensor management* proposed in this section is based on [218, 221]. Extensions to these publications are in particular the linearization procedure as well as the derivation of the bounding sensor for pruning.

3.1 Linear Gaussian Sensor Management

Due to the application of statistical linearization over the finite time horizon, the formerly problem of sensor management for nonlinear system and sensor models is converted into a management problem for linear models. On this account it makes sense to first give an introduction to the linear Gaussian sensor management problem, where the system is described by means of the linear discrete-time probabilistic system model (2.11). In anticipation of the linearized sensor models resulting from statistical linearization presented in Section 3.3.3, the more general linear Gaussian sensor model

$$\underline{\boldsymbol{z}}_k = \mathbf{H}_k^{\underline{u}_k} \cdot \underline{\boldsymbol{x}}_k + \underline{\boldsymbol{v}}_k^{\underline{u}_k} \ , \tag{3.1}$$

is considered here. In contrast to (2.12), not only the sensor matrix $\mathbf{H}_k^{\underline{u}_k} = \mathbf{H}_k(\underline{u}_k)$ but also the Gaussian measurement noise $\underline{\boldsymbol{v}}_k^{\underline{u}_k} = \underline{\boldsymbol{v}}_k(\underline{u}_k)$ varies with a particular configuration. More precisely, the covariance matrix $\mathbf{C}_k^{v,\underline{u}_k}$ and the mean vector $\hat{\underline{v}}_k^{\underline{u}_k}$ of $\underline{\boldsymbol{v}}_k^{\underline{u}_k}$ change depending on the configuration $\underline{u}_k$. It is further assumed that the sensor model (3.1) is stateless, i.e., the configuration $\underline{u}_k$ only affects the sensor model for the current time step k (see Section 2.1.3). The relaxation of this assumption is discussed in Section 3.3.4.

It is worth mentioning that linear models have applications in their own right, besides the usage as a result of statistical linearization within this thesis. For example, the temporal behavior of distributed physical phenomena can be represented by means of a linear model after applying spatial and temporal discretization, as described in [13, 159].

3.1.1 Problem Structure

In case of the considered model predictive control problem with covariance-based objective functions, the cumulative objective function (2.20) can be rewritten for the moving finite N-step time horizon according to

$$J_{k,0}(\underline{\boldsymbol{x}}_{k,0}^p) = \min_{\underline{\mu}_{k,0:N-1}} V_k\left(\underline{\boldsymbol{x}}_{k,0}^p, \underline{\mu}_{k,0:N-1}\right) \ , \tag{3.2}$$

with configuration dependent cumulative objective function

$$V_k\left(\underline{\boldsymbol{x}}_{k,0}^p, \underline{\mu}_{k,0:N-1}\right) = \mathrm{E}_{\underline{z}_{k,0:N-1}}\left\{\sum_{n=0}^{N-1} g_n(\underline{\boldsymbol{x}}_{k,n}^p, \underline{\mu}_{k,n}(\underline{\boldsymbol{x}}_{k,n}^p))\right\} ,$$

where $\underline{\boldsymbol{x}}_{k,0}^p = \underline{\boldsymbol{x}}_k^p \sim \mathcal{N}(\underline{x}_k^p; \hat{\underline{x}}_k^p; \mathbf{C}_k^p)$ is the current predicted state estimation at time k. This function can be further simplified under consideration of the special properties of the Kalman filter covariance matrix recursion:

1. The covariance calculation of the Kalman filter measurement update step is independent of the measurement value. Taking the expectation with respect to $\underline{z}_{k,0:N-1}$, for evaluating (3.2) and the objective functions $g_n(\underline{\boldsymbol{x}}_{k,n}^p, \underline{\mu}_{k,n}(\underline{\boldsymbol{x}}_{k,n}^p))$ has no effect. Thus, the expected posterior covariance matrix employed for $g_n(\underline{\boldsymbol{x}}_{k,n}^p, \underline{\mu}_{k,n}(\underline{\boldsymbol{x}}_{k,n}^p))$ coincides with the posterior covariance resulting from the Kalman filter measurement update step.[1]

2. The Kalman filter recursion for the covariance matrix is independent from the Kalman filter recursion of the mean vector. Thus, the covariance matrix already contains all information that is necessary to evaluate the objective functions within the time horizon, i.e., the covariance matrix is a sufficient statistic of the state in this context.

1 Hence, employing the entropy objective function (2.26) would lead to equivalent results as the determinant-based objective.

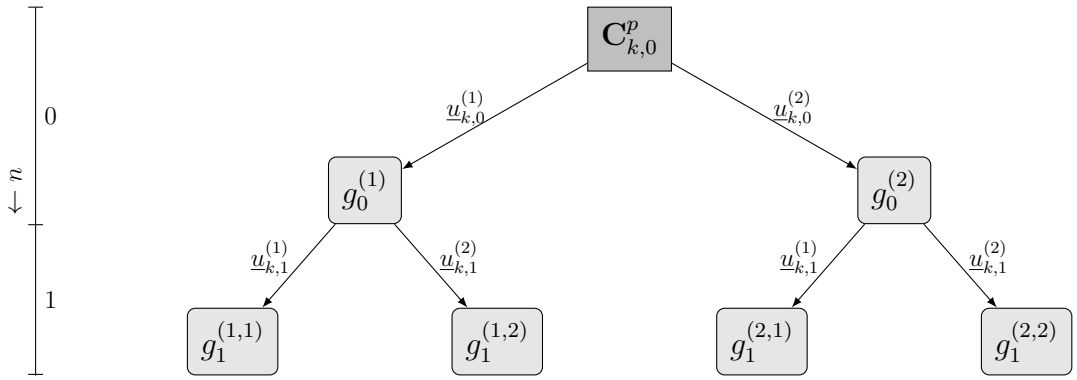


Figure 3.1: Search tree for $N = 2$ time steps and $|\mathcal{U}| = 2$ different configurations with root node $\mathbf{C}^p_{k,0}$.

Putting all together, the cumulative objective function (3.2) for the closed-loop model predictive control problem coincides with the open-loop model predictive control problem given by

$$J_{k,0}(\underline{\boldsymbol{x}}^p_{k,0}) = \min_{\underline{u}_{k,0:N-1}} V_k\left(\underline{\boldsymbol{x}}^p_{k,0}, \underline{u}_{k,0:N-1}\right) = \min_{\underline{u}_{k,0:N-1}} \sum_{n=0}^{N-1} g_n(\mathbf{C}^p_{k,n}, \underline{u}_{k,n}) \ , \tag{3.3}$$

where the step objective functions $g_n(\mathbf{C}^p_{k,n}, \underline{u}_{k,n})$ can be

- the determinant $g_n(\mathbf{C}^p_{k,n}, \underline{u}_{k,n}) = |\mathbf{C}^e_{k,n}(\underline{u}_{k,n})|$,
- the trace $g_n(\mathbf{C}^p_{k,n}, \underline{u}_{k,n}) = \mathrm{trace}(\mathbf{C}^e_{k,n}(\underline{u}_{k,n}))$,
- or the largest eigenvalue $g_n(\mathbf{C}^p_{k,n}, \underline{u}_{k,n}) = \lambda_{\max}(\mathbf{C}^e_{k,n}(\underline{u}_{k,n}))$.

3.1.2 Solution Procedure

Even if the control problem (3.3) is independent of actual measurement values, determining the optimal configuration sequence $\underline{u}^*_{k,0:N-1}$ is still a demanding problem, as the covariance matrices in (3.3) are a function of the configuration sequence. For a given configuration sequence $\underline{u}_{k,0:n}$ and under consideration of the extended linear sensor model (3.1), the covariance matrices evolve according to Riccati equation

$$\mathbf{C}^p_{k,n+1}(\underline{u}_{k,0:n}) = \mathbf{A}_{k,n}\Big(\underbrace{\mathbf{C}^p_{k,n}(\underline{u}_{k,0:n-1}) - \mathbf{K}^{\underline{u}_n}_{k,n}\mathbf{H}^{\underline{u}_n}_{k,n}\mathbf{C}^p_{k,n}(\underline{u}_{k,0:n-1})}_{=\mathbf{C}^e_{k,n}(\underline{u}_{k,0:n})}\Big)\mathbf{A}^{\mathrm{T}}_{k,n} + \mathbf{B}_{k,n}\mathbf{C}^w_{k,n}\mathbf{B}^{\mathrm{T}}_{k,n} \ , \tag{3.4}$$

with corresponding Kalman gain $\mathbf{K}^{\underline{u}_n}_{k,n}$. The estimated covariance matrix $\mathbf{C}^e_{k,n}(\underline{u}_{k,0:n})$ is employed for evaluating the objective functions $g_n(\,\cdot\,)$ and thus, determining the optimal configuration sequence requires the evaluation of the cumulative objective for every possible sequence $\underline{u}_{k,0:N-1}$. This corresponds to an exhaustive search in a tree with depth N and branching factor $|\mathcal{U}|$. Every possible configuration sequence forms a path in the tree, while the nodes correspond to the step objective functions $g_n(\,\cdot\,)$. In Figure 3.1, such a tree is depicted for $N = 2$, where the nodes of the tree are $g_n^{(i_0,i_1,\ldots,i_n)} := g_n\big(\mathbf{C}^p_{k,n}(\underline{u}^{(i_0)}_{k,0}, \underline{u}^{(i_1)}_{k,1}, \ldots, \underline{u}^{(i_{n-1})}_{k,n-1}), \underline{u}^{(i_n)}_{k,n}\big)$. Since the number of paths in the tree grows exponentially with the length of the time horizon, performing an exhaustive tree search is only tractable for short time horizons and a small set of configurations.

3.2 Optimal Pruning

However, since long time horizons and/or large sets of configurations are common in practical applications, an exhaustive tree-search has to be avoided. This can be achieved by employing pruning techniques. Existing pruning techniques can be classified into suboptimal methods and optimal methods. By employing an optimal pruning method, deleting the optimal configuration sequence is impossible. This leads to the evaluation of potentially many complete configuration sequences for determining the optimal sequence (see e.g. [42, 119]). Abdicating the guarantee of conserving the optimal sequence as suboptimal methods do allows for drastic savings in computational demand (see e.g. [4, 66]).

For significantly reducing the computational demand an early identification of sub-trees that contain only suboptimal configurations is essential. The novel optimal pruning technique *information-based pruning (IBP)* introduced in this section achieves this goal in two ways. By employing the so-called *sensor information matrix* and the monotonicity of the Riccati equation (3.4), suboptimal sub-trees are pruned without explicitly evaluating the Riccati equation. In a second step, the sensor information matrix is again utilized for calculating a so-called *bounding sensor*. For the remaining sub-trees, this bounding sensor provides a tight lower bound for the cumulative objective function (3.3). Employing branch-and-bound techniques with this novel lower bound leads to a further improvement in pruning performance and thus, to significant savings in computation time.

3.2.1 Preliminaries

All derivations and calculations are done for a fixed time index k, while merely the time index n varies within the time horizon. Hence, the index k is omitted from this point on. The development of the IBP method requires the monotonicity property of the Riccati equation, stated in the following theorem. For a proof see [67], Lemma 2.

Theorem 3.1 (Monotonicity of Riccati Equation) *Given two covariance matrices* $\mathbf{C}_n^p$ *and* $\tilde{\mathbf{C}}_n^p$ *with* $\mathbf{C}_n^p \succeq \tilde{\mathbf{C}}_n^p$, *i.e.,* $\mathbf{C}_n^p - \tilde{\mathbf{C}}_n^p$ *is positive semi-definite, applying the Riccati equation* (3.4) *for an arbitrary configuration* $\underline{u} \in \mathcal{U}$ *results in*

$$\mathbf{C}_n^e(\underline{u}) \succeq \tilde{\mathbf{C}}_n^e(\underline{u}) \quad \textit{and} \quad \mathbf{C}_{n+1}^p(\underline{u}) \succeq \tilde{\mathbf{C}}_{n+1}^p(\underline{u}) \ .$$

Thus, the positive semi-definite ordering between covariance matrices will not change by applying the Riccati equation. The ordering of covariance matrices automatically implies an ordering of the covariance-based objective functions.

Lemma 3.1 *Suppose that* $\mathbf{C}_n^e \succeq \tilde{\mathbf{C}}_n^e$, *then*

- $|\mathbf{C}_n^e| \geq |\tilde{\mathbf{C}}_n^e|$,
- $\operatorname{trace}(\mathbf{C}_n^e) \geq \operatorname{trace}(\tilde{\mathbf{C}}_n^e)$,
- *and* $\lambda_{\max}(\mathbf{C}_n^e) \geq \lambda_{\max}(\tilde{\mathbf{C}}_n^e)$.

For a proof of the lemma see [1].

Pruning nodes on the basis of the values of the step objectives $g_n(\,\cdot\,)$ only by neglecting the ordering of the covariance matrices automatically leads to suboptimal pruning results, because Lemma 3.1 merely provides a necessary condition for the positive semi-definite ordering between $\mathbf{C}_n^e$ and $\tilde{\mathbf{C}}_n^e$. Hence, the ordering of covariance matrices needs to be determined explicitly for each node of the search tree. On this account, the optimal pruning procedure proposed in

[119] evaluates the Riccati equation for each configuration, which is computational demanding for large state-spaces or for high dimensional measurement vectors. The IBP method instead allows determining the order of the covariance matrices without explicitly evaluating the Riccati equation multiple times.

3.2.2 Sensor Information Matrix

To commence, the information form of the covariance update equation (2.15) of the Kalman filter measurement update step is exploited. For an arbitrary configuration $\underline{u} \in \mathcal{U}$, the information form (see for example [95]) is given by

$$(\mathbf{C}_n^e)^{-1} = (\mathbf{C}_n^p)^{-1} + (\mathbf{H}_n^{\underline{u}})^{\mathrm{T}} (\mathbf{C}_n^{v,\underline{u}})^{-1} \mathbf{H}_n^{\underline{u}} ,$$

where $(\mathbf{C}_n^e)^{-1}$ is the Fisher information matrix and $\mathbf{C}_n^{v,\underline{u}}$ is the configuration dependent measurement noise (see (3.1)). This equation can be interpreted as gain of information over $\underline{\boldsymbol{x}}_n$ when performing a measurement for a given configuration $\underline{u}$. In this context the matrix $\mathbf{M}_n^{\underline{u}} := (\mathbf{H}_n^{\underline{u}})^{\mathrm{T}} (\mathbf{C}_n^{v,\underline{u}})^{-1} \mathbf{H}_n^{\underline{u}} \in \mathbb{R}^{n_x \times n_x}$ is denoted as *sensor information matrix*. It subsumes the information contribution of the configuration $\underline{u}$ at time step n and is symmetric as well as positive semi-definite. Based on this matrix, configurations can be excluded from searching the optimal configuration sequence without evaluating the Riccati equation.

Theorem 3.2 (Order of Sensor Information Matrices) *Given the covariance matrix $\mathbf{C}_n^p$ and the sensor information matrices $\mathbf{M}_n^{\tilde{\underline{u}}}$ and $\mathbf{M}_n^{\underline{u}}$ for two configurations $\tilde{\underline{u}}, \underline{u} \in \mathcal{U}$ such that*

$$\mathbf{M}_n^{\tilde{\underline{u}}} \succeq \mathbf{M}_n^{\underline{u}} , \tag{3.5}$$

i.e., $\mathbf{M}_n^{\tilde{\underline{u}}} - \mathbf{M}_n^{\underline{u}}$ is positive semi-definite, then $\mathbf{C}_n^e(\tilde{\underline{u}}) \preceq \mathbf{C}_n^e(\underline{u})$ and $\mathbf{C}_{n+1}^p(\tilde{\underline{u}}) \preceq \mathbf{C}_{n+1}^p(\underline{u})$.

Proof. Multiplying both sides of (3.5) from left with $\mathbf{C}_n^p$, adding the identity matrix $\mathbf{I}$, and inverting both sides leads to

$$\left(\mathbf{I} + \mathbf{C}_n^p \mathbf{M}_n^{\tilde{\underline{u}}}\right)^{-1} \preceq (\mathbf{I} + \mathbf{C}_n^p \mathbf{M}_n^{\underline{u}})^{-1} .$$

As the identity matrix is positive definite and the second summand is always positive semi-definite, the matrix inversion lemma [95]

$$(\mathbf{A} + \mathbf{BC})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}\left(\mathbf{I} + \mathbf{CA}^{-1}\mathbf{B}\right)^{-1}\mathbf{CA}^{-1}$$

can be applied. Afterwards, multiplying both sides from right with $\mathbf{C}_n^p$ yields $\mathbf{C}_n^e(\tilde{\underline{u}}) \preceq \mathbf{C}_n^e(\underline{u})$. Employing the covariance matrix prediction of the Kalman filter for $\mathbf{C}_n^e(\tilde{\underline{u}})$ and $\mathbf{C}_n^e(\underline{u})$ yields $\mathbf{C}_{n+1}^p(\tilde{\underline{u}}) \preceq \mathbf{C}_{n+1}^p(\underline{u})$. □

Thus, given the covariance matrix $\mathbf{C}_n^p$ at a arbitrary level of the search tree, it is possible the determine the ordering (3.5) between two configurations $\tilde{\underline{u}}$ and $\underline{u}$. Selecting configuration $\tilde{\underline{u}}$ will then provide a smaller covariance matrix at time step $n+1$. Due to the monotonic character of the Riccati equation, even arbitrary future configuration sequences $\underline{u}_{n:N-1}$ do not affect an existing order of covariance matrices.

Corollary 3.1 *Given two covariance matrices $\mathbf{C}_n^p$ and $\tilde{\mathbf{C}}_n^p$ with $\mathbf{C}_n^p \succeq \tilde{\mathbf{C}}_n^p$, $\forall u_{n:t}$, where $t \in \{n, n+1, \ldots, N-1\}$, exists a configuration sequence $\tilde{\underline{u}}_{n:t}$ such that*

$$\mathbf{C}_{t+1}^p(\underline{u}_{n:t}) \succeq \tilde{\mathbf{C}}_{t+1}^p(\tilde{\underline{u}}_{n:t}) .$$

Proof. Due to Theorem 3.1, at least the configuration sequence $\tilde{\underline{u}}_{n:t} = \underline{u}_{n:t}$ yields $\mathbf{C}_{t+1}^p(\underline{u}_{n:t}) \succeq \tilde{\mathbf{C}}_{t+1}^p(\tilde{\underline{u}}_{n:t})$. □

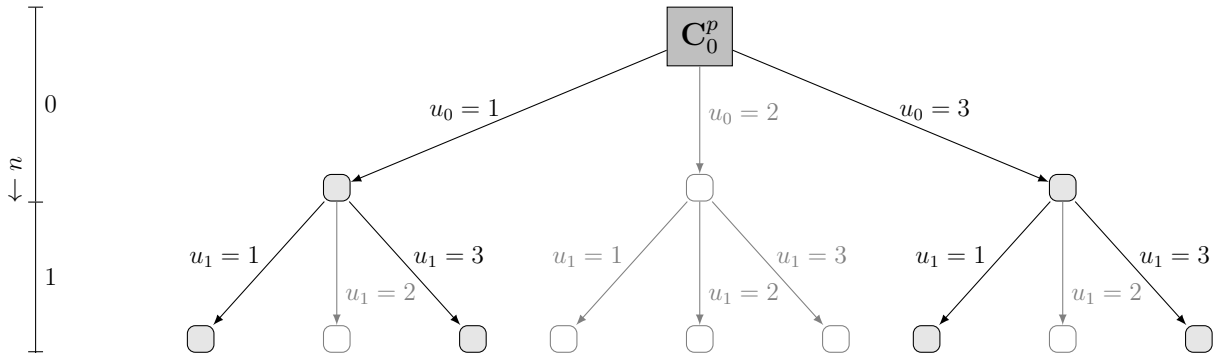


Figure 3.2: Search tree for Example 3.1 with $N = 2$ time steps and $|\mathcal{U}| = 3$ sensors. Sensor two can be excluded from search as its sensor information matrix is "smaller" than the corresponding matrix of sensor one.

Corollary 3.2 (Pruning based on Sensor Information Matrices) *Suppose that the covariance matrix $\mathbf{C}_n^p(u_{0:n-1})$ for the sensor sequence $\underline{u}_{0:n-1}$ is given. If for two configurations $\underline{u}_n$ and $\tilde{\underline{u}}_n$*

$$\mathbf{M}_n^{\underline{u}_n} \preceq \mathbf{M}_n^{\tilde{\underline{u}}_n} \ ,$$

then for the cumulative objective functions holds $V\big(\mathbf{C}_0^p, \underline{u}_{0:N-1}\big) \geq V\big(\mathbf{C}_0^p, \tilde{\underline{u}}_{0:N-1}\big)$ for any sequence $\underline{u}_{n+1:N-1}$, where $\tilde{\underline{u}}_{0:N-1} := \big(\underline{u}_{0:n-1}, \tilde{\underline{u}}_n, \underline{u}_{n+1:N-1}\big)$.

Proof. Follows directly from Theorem 3.2 and Corollary 3.1. □

Thus, without evaluating the Riccati equation at time step n and only by comparing the sensor information matrices, it can be decided that only configuration $\tilde{\underline{u}}_n$ needs to expanded for determining the optimal configuration sequence, while the covariance matrices $\mathbf{C}_{n+1}^p(\underline{u}_{0:n}), \ldots, \mathbf{C}_N^p(\underline{u}_{0:N-1})$ need not be determined for any sequence $\underline{u}_{n:N-1}$ with $\underline{u}_n \neq \tilde{\underline{u}}_n$. Hence, the complete sub-tree of configuration $\underline{u}_n$ can be pruned. In the following example, the effectiveness of the proposed optimal scheduling method is illustrated.

Example 3.1: Pruning based on Sensor Information Matrices
In this example a sensor scheduling problem for target tracking is considered. The dynamic behavior of the target is modeled by means of the constant velocity system model from Example 2.1 with sampling time interval $T = 0.2$ and diffusion strength $q = 0.02$. The target is observed by three linear sensors with time-invariant measurement matrices

$$\mathbf{H}^1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \ , \ \mathbf{H}^2 = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \ , \ \mathbf{H}^3 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

and noise covariance matrices $\mathbf{C}^{v,1} = \mathbf{C}^{v,3} = \mathrm{diag}([0.1, 0.1])$, $\mathbf{C}^{v,2} = 0.1$. This leads to the sensor information matrices

$$\mathbf{M}^1 = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \ , \ \mathbf{M}^2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \ , \ \mathbf{M}^3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix}$$

for all time steps, where a time horizon of $N = 2$ is assumed. It follows that $\mathbf{M}^1 \succeq \mathbf{M}^2$, while all other comparisons between sensor information matrices do not result in positive semi-definite differences. Due to the time-invariance of the measurement models, sensor 2 never needs to be considered for determining the optimal schedule. The resulting pruned search tree is depicted in Figure 3.2. According to this, the complete search tree consisting of 13 nodes is reduced to 7 nodes. ■

3.2.3 Order of Sensor Information Matrices

When comparing sensor information matrices (or equivalently covariance matrices), the difference is in some cases indefinite, i.e., it is not determinable, if one information matrix is "larger" than another (see $\mathbf{M}^2$ and $\mathbf{M}^3$ in Example 3.1). This is due to the fact that the order relation $\preceq$ of positive semi-definite matrices leads to partial orders. Thus, in general, not all configurations can be pruned at a specific time step.

There is one exception in case of scalar systems. Here, the partial order becomes a total order. Per time step, it is now possible to prune all configurations except of one, which is equivalent to selecting the configuration that minimizes the covariance at each time step. This greedy strategy leads automatically to the optimal configuration sequence. Similar findings can be found in [82] for the case of scheduling multiple scalar systems over time to a single sensor.

3.2.4 Branch-and-Bound

Pruning based on sensor information matrices leads to a significant reduction of possible configuration sequences. However, due to the partial order, the remaining number of nodes in the search tree may still be large. To further prune the tree, the proposed technique is combined with branch-and-bound (BB) pruning algorithms. Branch-and-bound pruning is common for classical problems like traveling-salesman or resource allocation [154]. The basic idea of branch-and-bound is to assign a lower bound of the achievable cumulative objective function value to any visited node. Based on these bounds, the tree is further expanded (branched), whereas nodes with a smaller lower bound are considered more promising to lead to the optimal configuration sequence and thus are expanded first. Nodes are pruned if their lower bound is larger than the cumulative objective value of an already completely evaluated configuration sequence.

For a particular node that was reached during tree-search by employing the configuration sequence $\underline{u}_{0:n-1}$, the cumulative objective function can be written according to

$$V\big(\mathbf{C}_0^p, \underline{u}_{0:N-1}\big) = \underbrace{V_0\left(\mathbf{C}_0^p, \underline{u}_{0:n-1}\right)}_{\text{known}} + \underbrace{V_n\left(\mathbf{C}_n^p(\underline{u}_{0:n-1}), \underline{u}_{n,N-1}\right)}_{\text{unknown}} \ , \tag{3.6}$$

where the value of the first summand $V_0(\,\cdot\,)$ is already known, while the value of the second summand $V_n(\,\cdot\,)$ is not calculated yet. A lower bound $\bar{V}\big(\underline{u}_{0:n-1}\big)$ for the cumulative objective (3.6) can be obtained by bounding $V_n(\,\cdot\,)$ from below such that

$$\bar{V}\big(\underline{u}_{0:n-1}\big) \leq V\big(\mathbf{C}_0^p, \underline{u}_{0:N-1}\big) \ , \ \forall \underline{u}_{n,N-1} \ .$$

Based on this bound, the sub-tree corresponding to the remaining configuration sequences $\underline{u}_{n,N-1}$ can be pruned, if the lower bound $\bar{V}(\,\cdot\,)$ is larger than a *global bound* $V_{\min}$, which is the cumulative objective value of the currently best completely evaluated configuration sequence. Obviously, the closer the bound $\bar{V}(\,\cdot\,)$ to the true value of (3.6), the earlier complete sub-trees can be pruned and thus, the better the pruning performance.

3.2.5 Bounding Sensor

Due to the cumulative structure of the objective function (3.6) with non-negative summands, a simple lower bound can be obtained by setting the second summand $V_n(\,\cdot\,)$ in (3.6) equal to zero. In the following, this simple bound is referred to as *zero bound* (ZB). This bound can be interpreted as expecting a complete reduction of uncertainty for each time step $n, \ldots, N-1$. Such a reduction can merely be achieved by a sensor information matrix with infinite trace or determinant. Obviously, such a sensor information matrix provides the coarsest lower bound possible.

A significantly tighter lower bound can be derived by means of a so-called *bounding sensor* with sensor information matrix $\bar{\mathbf{M}}_n$ for which the covariance matrices evolve according to

$$\bar{\mathbf{C}}_{n+1}^{p} = \mathbf{A}_n \underbrace{\left(\mathbf{I} + \bar{\mathbf{C}}_n^{p} \cdot \bar{\mathbf{M}}_n\right)^{-1}}_{=:\bar{\mathbf{C}}_n^{e}} \mathbf{A}_n^{\mathrm{T}} + \mathbf{B}_n \mathbf{C}_n^{w} \mathbf{B}_n^{\mathrm{T}} \ , \tag{3.7}$$

commencing from $\bar{\mathbf{C}}_0^{p} = \mathbf{C}_0^{p}$. For each time step n, this special information matrix fulfills

$$\bar{\mathbf{M}}_n \succeq \mathbf{M}_n^{\underline{u}} \tag{3.8}$$

for all configurations $\underline{u} \in \mathcal{U}$, where $\bar{\mathbf{M}}_n$ is as close to $\mathbf{M}_n^{\underline{u}}$ as possible. According to Theorem 3.2, for the predicted and estimated covariance matrices obtained from (3.7) holds $\bar{\mathbf{C}}_n^{e} \preceq \mathbf{C}_n^{e}(\underline{u})$ and $\bar{\mathbf{C}}_{n+1}^{p} \preceq \mathbf{C}_{n+1}^{p}(\underline{u})$ for all time steps. Hence, the cumulative objective function values for all possible configuration sequences $\underline{u}_{n,N-1}$, whose calculation requires exponential computation time, can be bounded from below by merely calculating the cumulative objective function for the bounding sensor for the time steps $n, \ldots, N-1$. Because of the utilization of one single configuration, which is implied by the bounding sensor, the determination of the lower bound can be done in linear time.

In the following, determining the bounding sensor with information matrix $\bar{\mathbf{M}}_n$ is described in detail.

Bounding Sensor for Two Configurations

Since sensor information matrices are symmetric and positive semi-definite, they can be graphically interpreted as ellipsoids similar to the covariance ellipsoids that correspond to covariance matrices (see Section A.1). In the following, the ellipsoid corresponding to a sensor information matrix is referred to as *sensor information ellipsoid*. Unlike covariance ellipsoids, sensor information ellipsoids have no distinguished position. Hence, it can be assumed that all sensor information ellipsoids are centered around the origin.

Based on this interpretation, determining the bounding sensor information matrix $\bar{\mathbf{M}}_n$ can be considered as the determination of the covering ellipsoid that contains the ellipsoids of all sensor information matrices. This problem is similar to the so-called *covariance union* or Löwner ellipsoid problem (see e.g. [23, 198]). Thanks to the fact that all sensor information ellipsoids have the same center, the covariance union problem can be significantly simplified.

At first, it is sufficient to consider the determination of the sensor information matrix of the bounding sensor for merely two configurations $\underline{u}^{(1)}, \underline{u}^{(2)} \in \mathcal{U}$ with information matrices $\mathbf{M}_n^{(1)}, \mathbf{M}_n^{(2)}$. For the two configurations case, a bounding sensor information matrix with minimum determinant, i.e., a minimum volume covering ellipsoid, results from solving a generalized eigenvalue problem as summarized in the following theorem and as depicted in Figure 3.3.

Theorem 3.3 (Minimum Bounding Sensor Information Matrix) *The bounding sensor information matrix $\bar{\mathbf{M}}_n$ with minimum determinant that fulfills* (3.8) *for two sensor information matrices $\mathbf{M}_n^{(1)}$ and $\mathbf{M}_n^{(2)}$ is given by*

$$\bar{\mathbf{M}}_n = \left(\mathbf{V}^{\mathrm{T}}\right)^{-1} \cdot \max\left(\mathbf{V}^{\mathrm{T}} \mathbf{M}_n^{(1)} \mathbf{V}, \mathbf{V}^{\mathrm{T}} \mathbf{M}_n^{(2)} \mathbf{V}\right) \cdot \mathbf{V}^{-1} \ , \tag{3.9}$$

where $\max(\,\cdot\,)$ *is the element-wise maximum of matrices and* $\mathbf{V}$ *is the matrix of generalized eigenvectors of $\mathbf{M}_n^{(1)}$ and $\mathbf{M}_n^{(2)}$.*

Proof. At first, the sensor information matrices $\mathbf{M}_n^{(1)}$ and $\mathbf{M}_n^{(2)}$ have to be diagonalized simultaneously, which requires solving the generalized eigenvalue problem

$$|\mathbf{M}_n^{(1)} - \lambda \mathbf{M}_n^{(2)}| = 0 \ . \tag{3.10}$$

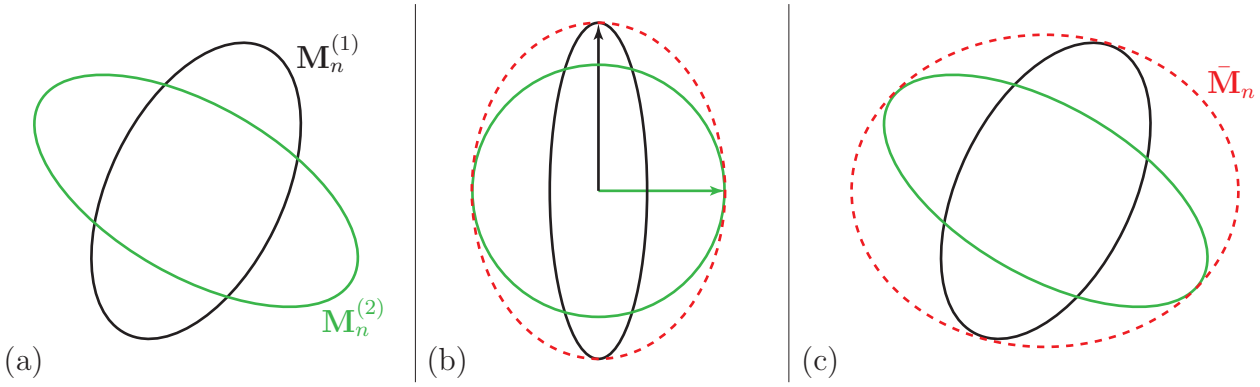


Figure 3.3: Graphical illustration of the determination the minimum bounding sensor information matrix for the two configurations case. (a) Sensor information ellipsoids of two sensor information matrices $\mathbf{M}_n^{(1)}, \mathbf{M}_n^{(2)}$. (b) Result of the simultaneous diagonalization of $\mathbf{M}_n^{(1)}$ and $\mathbf{M}_n^{(2)}$. The covering ellipsoid (red dashed ellipsoid) corresponding to the bounding sensor information matrix results from taking the maximum eigenvalue (or longest principal axis, indicated by the arrows) for each dimension. (c) Transforming back yields the desired minimum covering ellipsoid corresponding to $\bar{\mathbf{M}}_n$.

The solution of (3.10) is given by the eigenvalues $\lambda_i, i \in \{1, 2, \ldots, n_x\}$ and the matrix of eigenvectors $\mathbf{V}$. This allows diagonalizing $\mathbf{M}_n^{(1)}$ and $\mathbf{M}_n^{(2)}$ according to

$$\begin{aligned} \mathbf{V}^{\mathrm{T}} \mathbf{M}_n^{(1)} \mathbf{V} &= \operatorname{diag}\left([\lambda_1, \ldots, \lambda_{n_x}]\right) \ , \\ \mathbf{V}^{\mathrm{T}} \mathbf{M}_n^{(2)} \mathbf{V} &= \mathbf{I} \ . \end{aligned} \tag{3.11}$$

Determining the minimum volume covering ellipsoid for diagonal matrices is straightforward by taking the maximum of each diagonal element of the matrices in (3.11) according to

$$\max\left(\mathbf{V}^{\mathrm{T}} \mathbf{M}_n^{(1)} \mathbf{V}, \mathbf{V}^{\mathrm{T}} \mathbf{M}_n^{(2)} \mathbf{V}\right) \ . \tag{3.12}$$

This corresponds to taking the longest principal axis for each dimension or equivalently to taking the maximum of $(1, \lambda_i)$ for each $i \in \{1, 2, \ldots, n_x\}$ (see Figure 3.3 (b)). Multiplying $\left(\mathbf{V}^{\mathrm{T}}\right)^{-1}$ from left and $\mathbf{V}^{-1}$ from right to (3.12) reverses the diagonalization and leads to the desired minimum sensor information matrix $\bar{\mathbf{M}}_n$ of the bounding sensor (see Figure 3.3 (c)). □

Recursive Calculation for an Arbitrary Number of Configurations

Determining the bounding sensor with minimum sensor information matrix for an arbitrary number of configurations is computationally expensive in general, as numerical optimization is required [23]. However, a very tight but not necessarily minimum bounding sensor information matrix can be efficiently calculated by employing the solution of the two configurations case recursively in a pairwise fashion. In doing so, the complexity for determining the bounding sensor information matrix is linear with the number of configurations.

The recursion commences from the initial solution $\bar{\mathbf{M}}_n^{(2)}$ according to (3.9) for the first two configurations $\underline{u}^{(1)}, \underline{u}^{(2)} \in \mathcal{U}$. For the remaining configurations $\underline{u}^{(i)} \in \mathcal{U}$, with $i \in \{3, 4, \ldots, |\mathcal{U}|\}$, the recursion

$$\bar{\mathbf{M}}_n^{(i)} = \left(\mathbf{V}^{\mathrm{T}}\right)^{-1} \cdot \max\left(\mathbf{V}^{\mathrm{T}} \bar{\mathbf{M}}_n^{(i-1)} \mathbf{V}, \mathbf{V}^{\mathrm{T}} \bar{\mathbf{M}}_n^{(i)} \mathbf{V}\right) \cdot \mathbf{V}^{-1}$$

applies, where $\mathbf{V}$ is the matrix of eigenvectors of $\bar{\mathbf{M}}_n^{(i-1)}$ and $\bar{\mathbf{M}}_n^{(i)}$. The final solution $\bar{\mathbf{M}}_n^{(i)}$ for $i = |\mathcal{U}|$ is the desired bounding sensor information matrix $\bar{\mathbf{M}}_n$ that fulfills (3.8).

Algorithm 1 IBP($\mathbf{C}_n^p(\underline{u}_{0:n-1})$), where initially the global bound is set to $V_{\min} = \infty$.

```
1:  if n = N − 1 then
2:      V_min ← V(C_0^p, u_{0:N−1})        // Global bound given by currently best sequence u_{0:N−1}
3:  else
4:      U ← ∅                                                    // List of configurations to expand
5:      for u, ũ ∈ 𝒰 do
6:          if M_n^ũ ≻ M_n^u then
7:              U ← U ∪ {ũ}
8:          end if
9:      end for
10:     Calculate lower bound V̄(u_{0:n−1}, u) , ∀u ∈ U
11:     U ← sort(U)                                  // Sort configurations based on V̄(u_{0:n−1}, u)
12:     for all configurations u ∈ U do
13:         if V̄(u_{0:n−1}, u) < V_min then
14:             IBP(C_{n+1}^p(u_{0:n−1}, u))
15:         end if
16:     end for
17: end if
```

3.2.6 Information-based Pruning: Algorithm

In order to combine pruning based on the ordering of sensor information matrices and the bounding sensor, the information-based pruning method employs a depth-first search as summarized in Algorithm 1. Here, in line 5–9, the search space is at first reduced by employing pruning based on the ordering of sensor information matrices. For each of the remaining configurations in U a lower bound employing the bounding sensor is calculated and the configurations are sorted in ascending order according to their lower bound values (line 10–11). In doing so, further expanding the search tree is continued with the most promising configuration sequence first and traversing the search tree is accelerated, since nodes are not further expanded once the bound of a node is larger than the current global bound $V_{\min}$ (line 12–16). By completely evaluating a configuration sequence $\underline{u}_{0,N-1}$, the currently best sequence is automatically given and thus, the global bound is reduced (line 1–3).

3.2.7 Simulation Example

The effectiveness of the proposed information-based pruning method is demonstrated in the following by means of simulations employing a sensor scheduling problem for target tracking similar to Example 3.1. While a constant velocity system model with identical sampling time and diffusion strength is utilized, the target is now observed via a small sensor network consisting of six linear sensors with time-invariant measurement matrices

$$\mathbf{H}^1 = \mathbf{H}^3 = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \ , \ \mathbf{H}^2 = \mathbf{H}^5 = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \ ,$$
$$\mathbf{H}^4 = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \ , \ \mathbf{H}^6 = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}$$

and noise covariance matrices $\mathbf{C}^{v,1} = 0.2$, $\mathbf{C}^{v,2} = \mathbf{C}^{v,3} = \mathbf{C}^{v,4} = 0.1$, and $\mathbf{C}^{v,5} = \mathbf{C}^{v,6} = 0.05$. The mean vector and the covariance matrix of the initial state $\underline{\boldsymbol{x}}_0$ are $\hat{\underline{x}}_0 = [0, 1, 0, 1]^{\mathrm{T}}$ and $\mathbf{C}_0^x = \mathbf{I}$, respectively. For comparing the pruning performance, four different pruning methods are employed:

SIM Pruning based on the ordering of sensor information matrices only as described in Section 3.2.2.

ZB Branch-and-bound pruning based on the zero bound, i.e., the cumulative objective function (3.6) is simply bounded from below by setting $V_n(\,\cdot\,) = 0$.

SIM* Combines the previous two methods.

IBP The proposed information-based pruning method according to Algorithm 1.

As covariance-based objective function the trace is employed. Furthermore, the time horizon is set to $N = 8$, which leads to a search tree with $\sum_{i=1}^{8} 6^i \approx 2 \cdot 10^6$ nodes, where the optimal sensor sequence is given by $u^*_{0:N-1} = (6, 3, 5, 4, 3, 5, 6, 5)$.

As shown in Table 3.1, by merely using the sensor information matrices for pruning (SIM), the number of expanded nodes during the calculation of the optimal configuration sequence is reduced by a factor of about 20 compared to an exhaustive tree search. The branch-and-bound method based on the coarse zero bound (ZB) leads to more drastic reductions in number of expanded nodes. By additionally employing the order of sensor information matrices together with the zero bound (SIM*) further reduces the number of expanded nodes by a factor of four from 4232 to 1203. Here, additionally employing sensor information matrices can be interpreted as a preselection on candidate sensor sequences, since always sequences containing sensor 1 and 2 are pruned, while branch-and-bound pruning further thins out this candidate set. In doing so, the savings in computation time are even significantly greater, by a factor of 24, which is essential for sensor networks consisting of less capable sensor nodes.

The superior pruning performance and thus most savings in computation time provides the proposed information-based pruning method. While the ordering of sensor information matrices again preselects candidate sequences, the tight lower bound calculated on the basis of the bounding sensor allows pruning complete sub-trees much earlier as employing branch-and-bound with the coarse zero bound.

Table 3.1: Number of expanded nodes and computation time.

	SIM	ZB	SIM*	IBP
nodes expanded	87380	4232	1203	458
time in s	4643.74	209.49	8.58	1.72

3.3 Open-loop Sensor Management based on Statistical Linearization

In order to exploit the independence of the covariance recursion of the Kalman filter from measurement values and moreover to employ the previously introduced information-based pruning method, this section is devoted to converting the sensor management problem for nonlinear models and non-Gaussian densities to the linear Gaussian case. This conversion yields a computationally efficient open-loop model predictive control approximation to the original closed-loop model predictive control problem (3.2). The key idea behind this approximation lies in employing statistical linearization in a predictive fashion, which leads to a linearized representation of the nonlinear models over the considered time horizon. The linearized models are merely utilized for efficiently determining the next configuration, while the state inference still relies on the nonlinear models.

In the following section, the basic principle of predictive statistical linearization is introduced. Here, merely the propagation of the system state over the time horizon is considered for clarity. Further essential aspects, especially the incorporation of system and measurement noise into linearization and the calculation of the linearized system and sensor models are discussed afterwards.

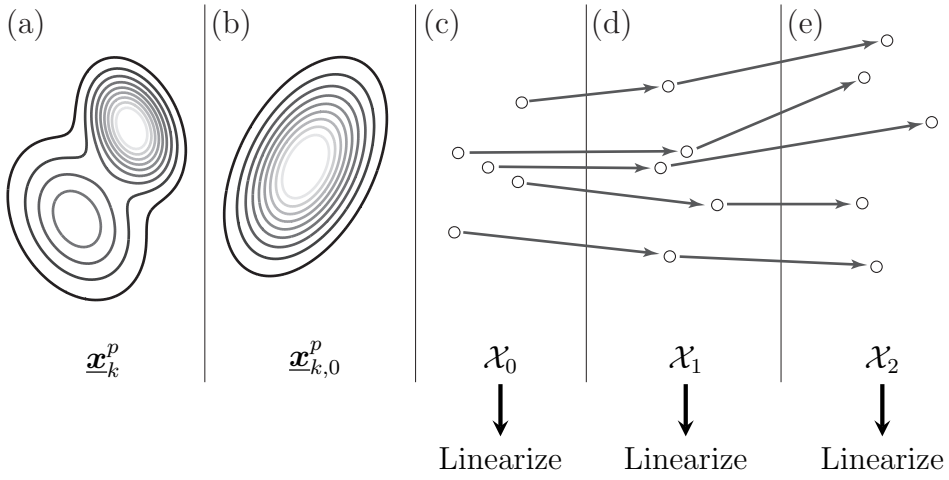


Figure 3.4: Calculation of the linearization trajectory for a time horizon of $N = 3$ steps. (a) Current system state characterized by means of an arbitrary density. (b) Gaussian density fitted to the first two moments of $\underline{\boldsymbol{x}}_k^p$. (c) Set of regression points for the Gaussian $\underline{\boldsymbol{x}}_{k,0}^p$. (d) Predicted set of regression points for time step $n = 1$. (e) Predicted set of regression points for time step $n = 2$.

3.3.1 Linearization Trajectory

To obtain the statistical linearization, a so-called *linearization trajectory* of regression points is necessary (see Figure 3.4 (c)-(e)). The basic idea is to represent the current predicted state estimate $\underline{\boldsymbol{x}}_k^p$ by means of a set of weighted regression points $\mathcal{X}_0 = \{\omega_{0,i}, \underline{x}_{0,i} | i = 1, \ldots, L\}$, at time step $n = 0$. Recursively propagating the regression points through the nonlinear system function $\underline{a}_{k+n}(\cdot)$ at each time step of the time horizon results in the desired trajectory of regression points $\mathcal{X}_{n+1} = \{\omega_{n+1,i}, \underline{x}_{n+1,i}\}$ that represent the predicted state $\underline{\boldsymbol{x}}_{n+1}^p$. Now, the nonlinear system and sensor model can be linearized about the trajectory of regression points by applying weighted statistical linear regression (see Section 2.3.4). Because of the predictive nature that merely utilizes measurement information available up to time step $k-1$, calculating the linearized models is performed in an open-loop fashion.

Existing predictive linearization approaches for open-loop sensor management like those in [42, 190] rely on first-order Taylor-series expansions, where the system and sensor models are only linearized about a single point, i.e., the mean of the predicted state estimate. However, the predictive nature typically leads to an unbounded growth of the uncertainty of the predicted state within the time horizon, which cannot be considered by standard linearization techniques. In contrast to this, the linearization trajectory of the proposed approach comprises a set of regression points $\mathcal{X}_n$ at each time step n within the time horizon. By this means, the uncertainty of the predicted state can be incorporated for linearization, which leads to more appropriate linearized models for approximating the nonlinear models in the uncertainty region captured by $\mathcal{X}_n$. As demonstrated in the simulations in Section 3.3.5, this positively affects the determination of appropriate configuration sequences and thus, the estimation accuracy.

3.3.2 Further Aspects

For the general quasi-linear sensor management approach considered in this chapter, some additional aspects for obtaining the linearization trajectory have to be considered: (1) Non-Gaussian density representation of the current system state estimate, (2) the incorporation of system and measurement noise into the calculation of the linearization trajectory, and (3) bounding the growth of the set of regression points resulting from the noise incorporation.

Initial Conversion of the State Density The predicted state estimate $\underline{\boldsymbol{x}}_k^p$ can be represented by means of an arbitrary density, depending on the employed state estimator for inference. A typical representation may be for example a Gaussian mixture density as employed by the hybrid density filter (see Chapter 6) or a set of particles in case of particle filters. Since the deterministic sampling techniques for calculating the regression points and the Kalman filter itself operate on Gaussian densities, the Gaussian $\underline{\boldsymbol{x}}_{k,0}^p \sim \mathcal{N}(\underline{x}_{k,0}^p; \underline{\hat{x}}_{k,0}^p, \mathbf{C}_{k,0}^p)$ that exactly captures the first two moments of the state estimate $\underline{\boldsymbol{x}}_k^p$ needs to be calculated initially at step $n = 0$ (see Figure 3.4 (a)-(b)).

Incorporation of Noise Since the considered probabilistic models are corrupted by noise, state augmentation has to be employed in order to incorporate the effects of noise into the statistical linearization. Thus, for the augmented state vector $\underline{\boldsymbol{X}}_n = [(\underline{\boldsymbol{x}}_n^p)^{\mathrm{T}}, \underline{\boldsymbol{w}}_n^{\mathrm{T}}, \underline{\boldsymbol{v}}_n^{\mathrm{T}}]^{\mathrm{T}}$ with mean vector and covariance matrix

$$\underline{\hat{X}}_n = \begin{bmatrix} \underline{\hat{x}}_n^p \\ \underline{0} \\ \underline{0} \end{bmatrix} \ , \ \mathbf{C}_n^X = \begin{bmatrix} \mathbf{C}_n^p & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_n^w & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{C}_n^v \end{bmatrix} \ ,$$

a set of augmented regression points $\mathcal{X}_n = \{\omega_{n,i}, \underline{X}_{n,i}\}$ needs to be determined. Here, $\underline{\hat{x}}_n^p = \sum_i \omega_{n,i} \cdot \underline{x}_{n,i}$ and $\mathbf{C}_n^p = \sum_i \omega_{n,i} \cdot (x_{n,i} - \underline{\hat{x}}_n^p)(x_{n,i} - \underline{\hat{x}}_n^p)^{\mathrm{T}}$. Furthermore, $\mathbf{C}_n^w$ and $\mathbf{C}_n^v$ are the covariance matrices of the noise terms $\underline{\boldsymbol{w}}_n = \underline{\boldsymbol{w}}_{k+n}$ and $\underline{\boldsymbol{v}}_n = \underline{\boldsymbol{v}}_{k+n}$, respectively. Since white noise is assumed, determining the sets of regression points $\mathcal{W}_n$ and $\mathcal{V}_n$ for the noise can be carried out independently of the state. There are many ways for combining the sets representing the noise with the regression points representing the system state in order to determine the set of augmented regression points $\mathcal{X}_n$. For example, each state regression point $\underline{x}_{n,i}$ can be combined with each point in $\mathcal{W}_n$ and $\mathcal{V}_n$, respectively. Independent of the considered method for combination, the number of regression points in $\mathcal{X}_n$ increases with the length of the time horizon.[2] One way to bound this growth is discussed in the following.

Gaussian Conversion of $\mathcal{X}_n$ Alternatively to recursively propagating the regression points $\mathcal{X}_n$ of the predicted state $\underline{\boldsymbol{x}}_n^p$ over the time horizon, the regression points can also be converted into a Gaussian density at each time step n. Afterwards, a novel set of regression points is calculated for this Gaussian. This alternative approach is used for all simulations conducted throughout this thesis. It automatically bounds the growth of the number of regression points in $\mathcal{X}_n$ caused by the necessary consideration of noise and keeps the computational demand on a constant level. Furthermore, this conversion makes sense in cases where outliers are among the regressions points of the set $\mathcal{X}_n$. Here, interim conversion of the regression points into a Gaussian can be seen as regularization similar to resampling for particle filters. However, as stated in [184, 196], by employing Gaussian conversion, information on odd-moments that are captured by the original set of regression points is discarded, which in turn decreases the linearization quality.

3.3.3 Calculation of Linearized System and Sensor Models

The complete algorithm for calculating the linearization trajectory is listed in Algorithm 2. For the linearization of the sensor model $\underline{h}_{k+n}(\underline{\boldsymbol{x}}_n, \underline{u}, \underline{\boldsymbol{v}}_n)$ for a given configuration $\underline{u}_n = \underline{u} \in \mathcal{U}$ in line 5, the regression points $\{\underline{x}_{n,i}, \underline{v}_{n,i}\}$ with corresponding weights $\omega_{n,i}$ have to be considered.

2 Simplifications can be made in case of additive noise affecting the system and the sensors. Here, calculating $\mathcal{W}_n$ and $\mathcal{V}_n$ is not necessarily required. Drawbacks of not calculating $\mathcal{W}_n$ and $\mathcal{V}_n$ are discussed in [196] for the unscented transform. However, these results can also be transferred on other LRKFs.

Algorithm 2 Statistical linearization for time step k

1: Calculate Gaussian approximation $\underline{\boldsymbol{x}}_{k,0}^{p} \sim \mathcal{N}(\underline{x}_{k,0}^{p}; \underline{\hat{x}}_{k,0}^{p}, \mathbf{C}_{k,0}^{p})$
2: **for** $n = 0, \ldots, N-1$ **do**
3: Calculate set of augmented regression points $\mathcal{X}_n = \{\omega_{n,i}, \underline{X}_{n,i}\}$
4: **for** each configuration $\underline{u} \in \mathcal{U}$ **do**
5: Linearize sensor model
6: **end for**
7: Propagate state regression points $\underline{x}_{n+1,i} = \underline{a}_{k+n}\left(\underline{x}_{n,i}, \underline{w}_{n,i}\right)$ // Calculate trajectory
8: Linearize system model
9: **end for**
10: Determine optimal configuration $\underline{u}_k^*$ // Solve linear Gaussian problem

By setting $\mathbf{A} = [\mathbf{H}_n^{\underline{u}}, \mathbf{D}_n^{\underline{u}}]$[3] and $\underline{\boldsymbol{x}} = [\underline{\boldsymbol{x}}_n^p, \underline{\boldsymbol{v}}_n]^{\mathrm{T}}$ in (2.17), the linearized sensor model analogous to (3.1) is given by

$$\underline{\boldsymbol{z}}_n = \mathbf{H}_n^{\underline{u}} \cdot \underline{\boldsymbol{x}}_n^p + \bar{\underline{\boldsymbol{v}}}_n^{\underline{u}} \,, \tag{3.13}$$

affected by the linearized measurement noise $\bar{\underline{\boldsymbol{v}}}_n^{\underline{u}}$ with mean vector and covariance matrix

$$\underline{\hat{v}}_n^{\underline{u}} = \underline{\hat{z}}_n - \mathbf{H}_n^{\underline{u}} \cdot \underline{\hat{x}}_n^p \,,$$
$$\mathbf{C}_n^{v,\underline{u}} = \mathbf{C}_n^z - \mathbf{H}_n^{\underline{u}} \cdot \mathbf{C}_n^p \cdot \left(\mathbf{H}_n^{\underline{u}}\right)^{\mathrm{T}} \,,$$

respectively. The mean vector $\underline{\hat{z}}_n$ and the covariance matrix $\mathbf{C}_n^z$ result from the weighted sample mean and covariance of $\mathcal{Z}_n = \{\omega_{n,i}, \underline{z}_{n,i} = \underline{h}_{k+n}(\underline{x}_{n,i}, \underline{u}, \underline{v}_{n,i})\}$.

Similarly, the nonlinear system model can be linearized by setting $\mathbf{A} = [\mathbf{A}_n, \mathbf{B}_n]$ and $\underline{\boldsymbol{x}} = [\underline{\boldsymbol{x}}_n^p, \underline{\boldsymbol{w}}_n]^{\mathrm{T}}$ in (2.17). Through this, the linearized system model is given by

$$\underline{\boldsymbol{x}}_{n+1}^p = \mathbf{A}_n \cdot \underline{\boldsymbol{x}}_n^p + \bar{\underline{\boldsymbol{w}}}_n \tag{3.14}$$

affect by the linearized system noise $\bar{\underline{\boldsymbol{w}}}_n$ with mean vector and covariance matrix

$$\underline{\hat{w}}_n = \underline{\hat{x}}_{n+1}^p - \mathbf{A}_n \cdot \underline{\hat{x}}_n^p \,,$$
$$\mathbf{C}_n^{\bar{w}} = \mathbf{C}_{n+1}^p - \mathbf{A}_n \cdot \mathbf{C}_n^p \cdot \mathbf{A}_n^{\mathrm{T}} \,,$$

respectively. The predicted state $\underline{\boldsymbol{x}}_{n+1}^p \sim \mathcal{N}(\underline{x}_{n+1}^p; \underline{\hat{x}}_{n+1}^p, \mathbf{C}_{n+1}^p)$ results from the predicted regression points $\underline{x}_{n+1,i}$ with weights $\omega_{n+1,i}$ determined at line 7 of Algorithm 2.

After executing line 1-9 of the linearization algorithm, $|\mathcal{U}| \cdot N$ linearized sensor matrices and measurement noise covariance matrices as well as N linearized system matrices and system noise covariance matrices are available. Based on these matrices, a linear Gaussian sensor management problem for the linearized system (3.14) and sensor models (3.13) can be solved (line 10). Linearizing the nonlinear system and sensor model is performed anew at time step $k+1$, after applying the optimal configuration $\underline{u}_k^*$ to the sensors.

3.3.4 Applicability

Obviously, the proposed open-loop model predictive control scheme is not restricted to a specific deterministic sampling scheme for determining the set of regression points. Thus, arbitrary linear regression Kalman filters can be employed within this framework. Even utilizing first-order Taylor-series expansion of the extended Kalman filter is possible. Here, merely a single-point linearization trajectory needs to be calculated. However, as discussed in Section 2.3.4, employing statistical regression instead of Taylor-series expansion typically leads to an improved linearization accuracy and finally, to an improved estimation performance.

3 Although the matrix $\mathbf{D}_n^{\underline{u}} \in \mathbb{R}^{n_x \times n_v}$ is calculated by the statistical linear regression, it is not needed further.

Linearization Accuracy

As the basic idea of the proposed quasi-linear sensor management approach depends on statistical linearization, the quality of the linearization trajectory and thus, the applicability and estimation accuracy, does not only depend on the particular scheme for determining the regression points. Generally, an adequate linearization accuracy is given in cases, where the nonlinearity of the system function $\underline{a}_k(\cdot)$, the uncertainty of the current state estimate $\underline{\boldsymbol{x}}_k^p$, and the strength of the system noise are relatively low. Furthermore, for finding an adequate length N of the time horizon, trading linearization accuracy against the consideration of long-term effects is necessary.

Stateless vs. State-dependent Sensor Management

Thanks to the initial assumption that the considered sensor model is stateless, the linearized system and sensor models can be determined prior to performing linear Gaussian sensor management (see Algorithm 2). In case of a state-dependent sensor management problem, statistical linearization can still be employed, where for the worst case, statistical linearization and tree search have to be performed simultaneously, i.e., for each node of the search tree linearization of the sensor model has to be performed during tree search. In contrast to a stateless problem, this procedure increases the number of linearized sensor matrices and thus increases the computational as well as the memory demand. The number of linearized system matrices remains unchanged, as linearization is still performed in a open-loop fashion.

A state-dependent problem also affects the application of the information-based pruning method introduced in Section 3.2. During tree search, future sensor models may be unknown due to the long-term dependence on the selected configuration and thus, employing pruning based on the ordering of sensor information matrices as well as calculating the bounding sensor is restricted.

Example 3.2: Degrading Applicability due to State-dependency
The unrestricted applicability of the proposed open-loop model predictive control scheme has to be checked from case to case. This is discussed in the following by means of three application examples, where the degree of applicability gradually degrades.

Sensor Scheduling As scheduling sensors for measurement merely affects the selected sensor for a specific time step, the sensor scheduling problem is stateless and the proposed open-loop model predictive control scheme can be fully applied.

Sensors with Low Sampling Rate Consider a modified sensor scheduling scenario, where some of the sensors cannot perform repeated measurements, i.e., these sensors are unavailable from time to time. This may happen e.g. in case of ultra-sonic sensors, which have to wait until all reflections are faded away. Even if this problem is state-dependent, a priori linearization can still be applied. The same is true for calculating the bounding sensor. Here, the sensors with low sampling rate have to be considered for calculating the lower bound even if they are sometimes unavailable, which leads to a more conservative lower bound.

Mobile Sensor Control Controlling the movement of the mobile distance sensor introduced in Example 2.4 leads to state-dependent sensor management problem, where statistical linearization has to be performed simultaneously with the tree search. In contrast to the previous example, selecting a configuration affects *all* possible future models. Hence, information-based pruning cannot or only suboptimally be employed, whereas branch-and-bound pruning based on the zero bound is applicable. ■

3.3.5 Simulation Results

To evaluate the performance of the proposed open-loop model predictive control scheme for sensor management, Monte Carlo simulations for two target tracking scenarios are conducted. The system model of the target coincides with the constant velocity model introduced in Example 2.1. For simulation purposes, the sampling time is set to $T = 1\,\mathrm{s}$ and the diffusion strength to $q = 0.5$. For state inference, a combination of Kalman filters and hybrid density filter (HDF), which is introduced in Chapter 6, is used. For the measurement update step, the HDF facilitates an accurate state estimation for the nonlinear sensor models employed in both simulations. For this purpose, the HDF processes $9 \cdot 10^5$ hybrid density components. Thanks to the linearity of the system model, the Gaussian mixture densities resulting from the HDF are optimally processed in the prediction step by means of a bank of Kalman filters.

For performing statistical linearization, the deterministic sampling scheme proposed for the Gaussian estimator in Chapter 5 is employed. Here, the density function of the (augmented) state is represented by five regression points per dimension, which results in a total of $5 \cdot 9 = 45$ regression points.

Sensor Scheduling

For the first simulation, the target is observed over 20 time steps via a small heterogeneous sensor network consisting of five sensors, where two distance sensors and three bearing sensors are used. The task is to select one sensor per time step for measurement, i.e., a sensor scheduling problem is considered here and the set of configurations $\mathcal{U} = \{1, 2, \ldots, 5\}$ represents the sensor indices. For the bearing sensors with indices $u \in \{1, 4, 5\}$, the sensor model is given by

$$\boldsymbol{z}_k^u = \arctan\left(\frac{\boldsymbol{y}_k - y^u}{\boldsymbol{x}_k - x^u}\right) + \boldsymbol{v}_k^u \,, \tag{3.15}$$

where the measurement noise $\boldsymbol{v}_k^u$ is zero-mean white Gaussian with standard deviation $\sigma_k^{v,u} = 0.02\,\mathrm{rad}$ for $u \in \{1, 4, 5\}$. The sensors two and three are distance sensors with sensor model

$$\boldsymbol{z}_k^u = \sqrt{(\boldsymbol{x}_k - x^u)^2 + (\boldsymbol{y}_k - y^u)^2} + \boldsymbol{v}_k^u \,, \tag{3.16}$$

where the zero-mean white Gaussian noise $\boldsymbol{v}_k^u$ has the standard deviation $\sigma_k^{v,u} = 0.5\,\mathrm{m}$ for $u \in \{2, 3\}$. The position $[x^u, y^u]^{\mathrm{T}}$ of each sensor is depicted in Figure 3.5 (a), together with an example trajectory of the target. The initial state of the target $\underline{\boldsymbol{x}}_0$ is represented by means of a Gaussian density with mean vector $\underline{\hat{x}}_0 = [0\,\mathrm{m}, 1\,\mathrm{m/s}, 0\,\mathrm{m}, 1\,\mathrm{m/s}]^{\mathrm{T}}$ and covariance matrix $\mathbf{C}_0 = \mathrm{diag}\left([50\,\mathrm{m}^2, 10\,\mathrm{m}^2/\mathrm{s}^2, 50\,\mathrm{m}^2, 10\,\mathrm{m}^2/\mathrm{s}^2]\right)$.

For this setup, 50 Monte Carlo simulation runs are performed for four different quasi-linear sensor managers with time horizon $N \in \{1, 2, 3, 4\}$. Additionally, a random sensor manager and a nearest neighbor manager (NN) are employed. While the random manager selects the next sensor randomly, the NN manager selects the sensor which minimizes the Mahalanobis distance between the sensor position and the current position estimate [199]. As illustrated in Figure 3.5 (b), the proposed quasi-linear sensor managers significantly outperform the random and NN manager with respect to the root mean-square error (rmse) of the target position. The inferior performance of the NN manager compared to the random approach can be explained by the fact that the same sensor is often selected consecutively. Furthermore, for most of the simulation runs, only a subset of the sensors is selected by NN.

As expected, the longer the optimization horizon of the quasi-linear manager, the better is the tracking performance. Especially the long horizons $N = 3$ and $N = 4$ lead to diversified sensor schedules. This in turn results in a fast reduction of the initial uncertainty of the target

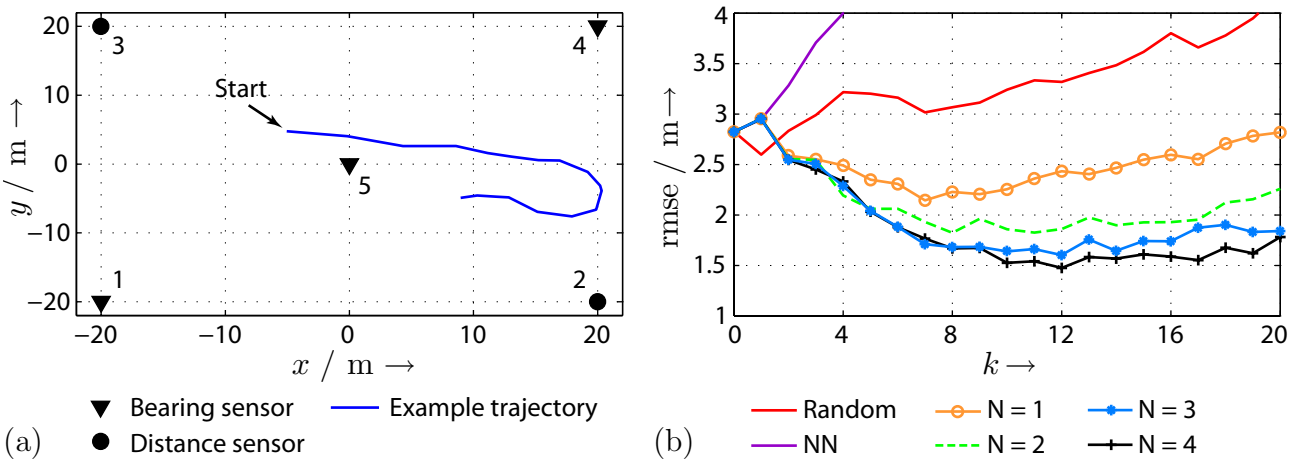


Figure 3.5: (a) Simulation setup. (b) Average rmse over 50 Monte Carlo simulation runs for different sensor managers.

estimate. Furthermore, an almost constant tracking error can be achieved after a transition time of about 10 time steps. In contrast to this, the tracking performance of the greedy strategy ($N = 1$) degrades after 10 time steps.

The better performance of long time horizons typically comes at the expense of a higher computational burden. However, as listed in Table 3.2, the complexity of the tree search can be kept bounded thanks to the employed IBP[4]. While the search tree grows exponentially with a base of five, (see third row of Table 3.2), the number of expanded nodes merely doubles with the length of the time horizon. Thus, the computation time for calculating the optimal sensor sequence increases from 0.025 s for $N = 2$ to 0.119 s for $N = 4$, which is moderate and facilitates target tracking in real-time even for long time horizons.

Table 3.2: Number of expanded nodes for different time horizons N using IBP in comparison to an exhaustive tree search.

	$N = 1$	$N = 2$	$N = 3$	$N = 4$
IBP	5	14	26	50
exhaustive search	5	30	155	780

Controlling a Mobile Distance Sensor

In this scenario, the target is now observed by means of a single but mobile distance sensor. The motion of the sensor is described via the kinematic model (2.6) of Example 2.4. The sensor can change its orientation by one of five possible steering angles $u \in \{\frac{i\pi}{8} | i = -2, \dots, 2\}$. Afterwards, the sensor moves along its new orientation with velocity $v = 20$ m/s. It is assumed that the new position is exactly known to the sensor. Additionally, the sensor can stop its motion for the considered time step. Thus, these six sensor motions define the set $\mathcal{U}$ of possible configurations. This scenario is inspired by the target tracking application described in [42], where a torpedo performing bearings-only measurements is considered for target tracking.

The initial sensor position is $[x_k^s, y_k^s, \phi_k^s]^{\mathrm{T}} = [-100\ \mathrm{m}, 300\ \mathrm{m}, \pi/2\ \mathrm{rad}]^{\mathrm{T}}$ and the measurement noise is assumed to be zero-mean white Gaussian with standard deviation $\sigma_k^v = 0.5$ m. The initial target state $\underline{\boldsymbol{x}}_0$ is represented by means of a Gaussian density with mean vector $\hat{\underline{x}}_0 = [0\ \mathrm{m}, -4.5\ \mathrm{m/s}, 500\ \mathrm{m}, -4.5\ \mathrm{m/s}]^{\mathrm{T}}$ and covariance matrix $\mathbf{C}_0 = \mathrm{diag}([500\ \mathrm{m}^2, 10\ \mathrm{m}^2/\mathrm{s}^2,$

4 Since sensor scheduling is stateless management problem, IBP is fully applicable.

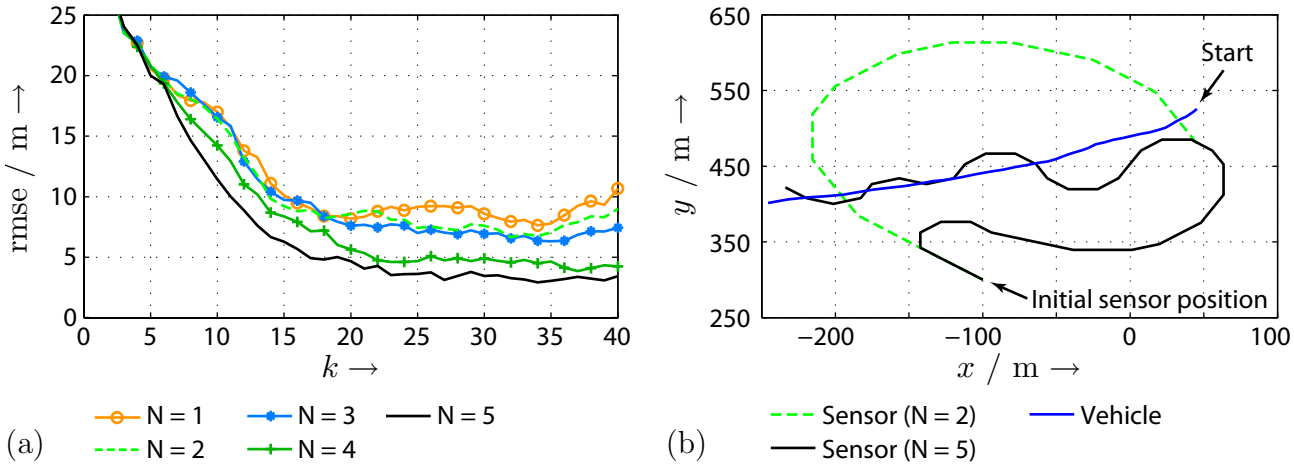


Figure 3.6: (a) Average rmse for time horizons $N \in \{1, 2, \ldots, 5\}$. (b) Sensor trajectories for the time horizons $N = 2$ and $N = 5$ together with an example target trajectory.

500 $\mathrm{m}^2, 10\ \mathrm{m}^2/\mathrm{s}^2$]), i.e., the average velocity of the target is 6.36 m/s, which is approximately by a factor of three smaller than the sensor velocity. The target travels for 40 time steps.

For these parameters, five quasi-linear sensor managers for the time horizons $N \in \{1, \ldots, 5\}$ are investigated in 100 Monte Carlo runs. Similar to the scheduling problem considered before, increasing the length of the time horizon leads to an improved tracking accuracy. This is illustrated in Figure 3.6 (a). Interestingly, the results for the long time horizons $N \in \{4, 5\}$ significantly differ from the results of the horizon lengths $N \in \{1, 2, 3\}$. This finding can be explained by investigating the sensor trajectories (see Figure 3.6 (b)). For short horizons, the resulting trajectory is similar to a circular arc. In this case, sensor management suffers from the constrained sensor movement. Instead, for long lookaheads, the sensor is able to follow the trajectory of the target since the sensor manager utilizes the much higher sensor velocity compared to the target velocity. In doing so, the sensor can catch up, even if the sensor starts its motion far away from the target.

Besides the length of the lookahead, the employed linearization method for determining the linearized sensor models plays an important role for the achievable performance. So far, statistical linearization based on the deterministic sampling scheme of the Gaussian estimator was employed (see Section 5). By replacing this statistical linearization method with linearization by first-order Taylor-series expansion as employed for the well-known extended Kalman filter (EKF), the tracking performance degrades. This is illustrated in Figure 3.7 for the sensor managers with time horizon $N = 2$ and $N = 5$. Determining configurations sequences on the basis of statistical linearization benefits from the consideration of the uncertainty during planning, which is comprised by covariance of the state estimate. Though, the performance difference for the short horizon of $N = 2$ is much more significant compared to the long lookahead, where the rmse is only approximately 1 m lower from time step 25 on. Long horizons amplify the effect that the superior linearization accuracy of statistical linearization is dimished by the uncertainty introduced by the prediction step. However, in case of the long horizon, the manager based on statistical linearization still converges much faster.

3.4 Summary

The determination of optimal configuration sequences for nonlinear system and sensor models as well as covariance-based objective functions generally requires solving a closed-loop model predictive control problem. Due to the incorporation of future information about the system state

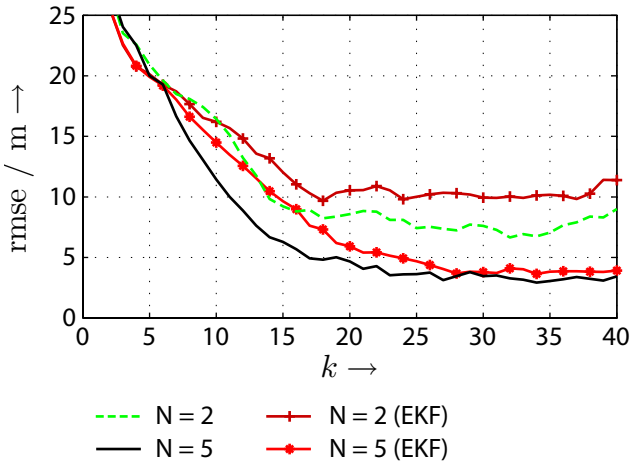


Figure 3.7: Comparing the tracking performance of the proposed statistical linearization based quasi-linear sensor management with sensor management employing linearization by means of first-order Taylor-series expansion.

into the control, calculating the optimal sequence is computationally demanding. An approximation to the optimal sequence can be obtained by considering the corresponding open-loop model predictive control problem. The novel *quasi-linear sensor management* scheme proposed in this chapter provides this approximate solution in a very efficient way. The efficiency of this scheme results from two techniques: statistical linearization and optimal pruning.

Statistical linearization performed in a predictive manner provides a linearized approximation of the nonlinear models over the considered time horizon. In doing so, the nonlinear non-Gaussian sensor management problem is converted into a linear Gaussian problem, for which the optimal configuration sequence can be determined independent of future measurement values. Thanks to the model predictive control structure, novel information about the system state can be made available by applying the first element of the optimal sequence to the sensors. The incorporation of the new measurement values into the state estimate still relies on the actual nonlinear models and non-Gaussian densities, which allows employing sophisticated Bayesian estimators for a high quality state inference.

Existing approaches on open-loop sensor management often rely on specific applications. Thus, dealing with nonlinearities and long optimization horizons is only applicable for restricted problem types. For example, [190] exploits the problem structure of tracking a single target for a significant reduction in computational demand of planning over long time horizons. Other approaches exploit the linearity of the system model considered in typical target tracking problems. By this means, it is sufficient to employ the extended Kalman filter for linearizing the sensor model multiple time steps ahead [42] or to calculate configuration sequences on the basis of minimizing the posterior Cramér-Rao lower bound [41, 80]. In case of nonlinear system models or strong nonlinearities in the sensor model, EKF and PCRLB-based approaches provide poor estimation results as the growing uncertainty along the time horizon is not incorporated during linearization. In case of non-differentiable models, these approaches are even not applicable. The proposed sensor management approach can be considered as a direct extension of existing EKF-based approaches to a larger class of nonlinear sensor management problems. However, for strong nonlinearities in both system and sensor models, and strongly non-Gaussian density functions even statistical linearization will lead to poorly linearized models. Here, dealing directly with the closed-loop model predictive control sensor management problem as it is done in the next chapter is more promising.

By converting the nonlinear non-Gaussian sensor management problem into a linear Gaussian one, the optimal sensor sequence results from solving a deterministic optimization problem

by means of a tree search. Since the tree consists of $|\mathcal{U}|^N$ possible configuration sequences, an exhaustive search is only feasible for a small set of configurations and short time horizons. By means of the proposed optimal pruning technique named information-based pruning, complete subtrees can be pruned very early, while preserving the optimal configuration sequence is guaranteed. Compared to existing branch-and-bound based optimal pruning methods like those introduced in [42], the effectiveness of the proposed method relies on exploiting the properties of the sensor information matrices. This allows pruning without explicitly evaluating the Riccati equation and by calculating a tight lower bound to the cumulative objective function, which in turn facilitates a more effective branch-and-bound pruning. For some state-dependent sensor management problems, information-based pruning is only applicable in a restricted manner. Further improving the pruning performance in such cases is devoted to future research.

CHAPTER 4

Information Theoretic Sensor Management

The Bellman recursion (2.20) for determining the optimal configuration sequence over a finite time horizon merely provides a conceptual solution framework. For a continuous state space and continuous measurement values, calculating the optimal solution in general is computationally intractable or even impossible. Instead, approximate approaches have to be employed.

The approach proposed in the previous chapter represents one way for approximately solving the sensor management problem. Being an open-loop control approach, no information about future measurement values is anticipated for planning the configuration sequence. Moreover, this approach relies on statistical linearization, where uncertainties over the system state are captured by means of Gaussian densities, respectively. Consequently, in cases of mild nonlinearities and nearly Gaussian uncertainties, good results are provided. The stronger the considered scenario or application differs form these conditions, the less advantages can be gained from planning over long time horizons.

This chapter is devoted to a closed-loop model predictive control approach named *information theoretic sensor management* that provides an approximate solution of the sensor management problem on the basis of sequences of simulated measurement values for each possible configuration sequence. By means of these so-called *virtual measurements*, the effect of future measurement values can be incorporated into sensor management. For an adequate consideration of even strong nonlinearities and non-Gaussian uncertainties, this approach further employs

- a Gaussian mixture representation of the state density function and
- mutual information as information theoretic objective function.

In doing so, more sophisticated nonlinear state estimators, e.g., the hybrid density filter introduced in Chapter 6, can be used and the resulting higher-order information of the state estimate can be quantified and considered in the determination of the optimal configuration sequence.

With regard to long time horizons and/or a large set of possible configurations, the proposed information theoretic sensor management approach may become computationally infeasible if implemented in a straightforward manner. Hence, this chapter is also devoted to various techniques that significantly lower the computational demand. Analog to the previous management approach, the optimal configuration sequence results from a tree search. But now, the branching factor of the tree not only depends on the number of configurations but also on the number of virtual measurements. Hence, it is of paramount importance to create few but representative measurement values, which is achieved on the basis of an optimal Dirac mixture approximation method (see Section 4.1.2). Due to the usage of virtual measurements, classical pruning methods for efficiently traversing the search tree are improper. Instead, a novel probabilistic

branch-and-bound algorithm is applied in Section 4.1.4, which accounts for the probabilistic representation of the virtual measurements.

Mutual information, as the employed information theoretic objective function, represents a central part of the proposed sensor management approach that has to tb evaluated multiple times during the tree search. Due to the Gaussian mixture representation of the state density, its closed-form evaluation is not possible. More specifically, the differential entropy terms that form mutual information cannot be evaluated in closed form due to the logarithm of a sum of exponential functions. Instead of falling back on computationally expensive entropy approximations that additionally may deviate from the true entropy values in an arbitrary fashion, Section 4.2 discusses the cheap calculation of tight lower and upper bounds. Besides the importance of these bounds for a computationally tractable determination of the configuration sequence, their application is not restrict to the proposed sensor management approach.

The foundation of this chapter was published in [230] in the context of closed-loop model predictive control of stochastic nonlinear systems. Especially, the concept of the virtual measurements and pruning via the probabilistic branch-and-bound algorithm can be found here. The upper and lower bounds on the differential entropy are part of [213]. The combination of all these techniques to the information theoretic sensor manager and further improvements on the entropy bounds are the main contributions of this chapter.

4.1 Closed-loop Control

For determining the optimal configuration vector $\underline{u}_k^*$ in a closed-loop model predictive control fashion, the optimal control problem

$$\underline{u}_k^* = \underline{\mu}_k^*(\underline{\boldsymbol{x}}_k^p) = \arg \max_{\underline{\mu}_{k,0}} \underbrace{\max_{\underline{\mu}_{k,1:N-1}} \mathrm{E}_{\underline{z}_{k,0:N-1}} \left\{ \sum_{n=0}^{N-1} g_n(\underline{\boldsymbol{x}}_{k,n}^p, \underline{\mu}_{k,n}(\underline{\boldsymbol{x}}_{k,n}^p)) \right\}}_{V_{k,0}\left(\underline{\boldsymbol{x}}_{k,0}^p, \mu_{k,0}(\underline{\boldsymbol{x}}_{k,0}^p)\right)} \tag{4.1}$$

needs to be solved for the moving N-step time horizon at any time step k. The optimal configuration vector $\underline{u}_k^* = \underline{u}_{k,0}^*$ is then applied to the sensors and the whole procedure is repeated at the next time step $k+1$. In the following, the time index k is omitted, since merely the time index n within the time horizon varies for all necessary calculations.

4.1.1 Control Setting

According to Section 2.4, the optimal control policy $\underline{\mu}_k^*(\underline{\boldsymbol{x}}_k^p)$ in (4.1) anticipates the arrival of future measurement values under consideration of the nonlinear system model (2.1) and the nonlinear sensor model (2.3). Determining the optimal policy corresponds to solving a POMDP with continuous state space and measurement space. In the considered stochastic setting, the probability density function characterizing the system state forms the sufficient statistic as it fully describes the system state at a particular time step within the time horizon.

Objective Function

In order to capture all information subsumed in the state densities instead of only considering second-order moments as in the previous chapter, the sensor management approach proposed in this chapter focuses on mutual information (2.27) as objective function. Mutual information belongs to the class of information theoretic objective functions and quantifies the expected information gain, i.e., the reduction of uncertainty over the state estimate by incorporating

sensor measurements. With the step objective $g_n(\underline{\boldsymbol{x}}_n, \underline{u}_n) = I(\underline{\boldsymbol{x}}_n; \underline{\boldsymbol{z}}_n | \underline{z}_{0:n-1}, \underline{\mu}_{0:n})$, where conditioning is on the value of the past measurements $\underline{z}_{0:n-1}$, and assuming that $\underline{u}_k^*$ is determined by applying the policy $\underline{\mu}_{1:N-1}$ within the time horizon, (4.1) can be written according to

$$\underline{u}_k^* = \arg \max_{\underline{u}_{k,0}} \max_{\underline{\mu}_{1:N-1}} \mathrm{E}_{\underline{z}_{0:N-1}} \left\{ \sum_{n=0}^{N-1} I\Big(\underline{\boldsymbol{x}}_n; \underline{\boldsymbol{z}}_n | \underline{z}_{0:n-1}, \underline{\mu}_{0:n}\Big) \right\} . \tag{4.2}$$

This is equivalent to maximizing the joint mutual information over the whole time horizon

$$\underline{u}_k^* = \arg \max_{\underline{u}_{k,0}} \max_{\underline{\mu}_{1:N-1}} I\Big(\underline{\boldsymbol{x}}_{0:N-1}; \underline{\boldsymbol{z}}_{0:N-1} | \underline{\mu}_{0:N-1}\Big) .$$

To see this, the expectation $\mathrm{E}_{\underline{z}_{0:N-1}}$ has to be incorporated into the mutual information objective, which leads to a conditioning on random vectors $\underline{\boldsymbol{z}}_{0:N-1}$. The chain rule for mutual information (2.28) is then applied on (4.2), assuming that the measurement $\underline{\boldsymbol{z}}_n$ is independent of all other measurements and system states when conditioned on $\underline{\boldsymbol{x}}_n$.

Another common information theoretic objective function utilized for sensor management is the conditional entropy (2.24) (see e.g. [43, 57, 186]). For the sake of completeness, it is shown in the following that considering conditional entropy as step objective instead of mutual information merely leads to an upper bound approximation. Applying (2.29) on (4.2) yields

$$\begin{aligned} J_0(\underline{\boldsymbol{x}}_0) :&= \max_{\underline{\mu}_{0:N-1}} \mathrm{E}_{\underline{z}_{0:N-1}} \left\{ \sum_{n=0}^{N-1} I\Big(\underline{\boldsymbol{x}}_n; \underline{\boldsymbol{z}}_n | \underline{z}_{0:n-1}, \underline{\mu}_{0:n}\Big) \right\} \\ &= \max_{\underline{\mu}_{0:N-1}} \mathrm{E}_{\underline{z}_{0:N-1}} \left\{ \sum_{n=0}^{N-1} H\Big(\underline{\boldsymbol{x}}_n | \underline{z}_{0:n-1}, \underline{\mu}_{0:n}\Big) - H\Big(\underline{\boldsymbol{x}}_n | \underline{\boldsymbol{z}}_n, \underline{z}_{0:n-1}, \underline{\mu}_{0:n}\Big) \right\} . \end{aligned} \tag{4.3}$$

Employing the order (2.25) on the first summand in (4.3) results in the upper bound

$$J_0(\underline{\boldsymbol{x}}_0) \leq \max_{\underline{\mu}_{0:N-1}} \mathrm{E}_{\underline{z}_{0:N-1}} \left\{ \sum_{n=0}^{N-1} H(\underline{\boldsymbol{x}}_n) - H\Big(\underline{\boldsymbol{x}}_n | \underline{\boldsymbol{z}}_n, \underline{z}_{0:n-1}, \underline{\mu}_{0:n}\Big) \right\} .$$

Here, the first entropy term $H(\underline{\boldsymbol{x}}_n)$ does not depend on sensor measurements and thus can be omitted, which leads to

$$J_0(\underline{\boldsymbol{x}}_0) \leq \min_{\underline{\mu}_{0:N-1}} \mathrm{E}_{\underline{z}_{0:N-1}} \left\{ \sum_{n=0}^{N-1} H\Big(\underline{\boldsymbol{x}}_n | \underline{\boldsymbol{z}}_n, \underline{z}_{0:n-1}, \underline{\mu}_{0:n}\Big) \right\} . \tag{4.4}$$

Hence, a better cumulative objective value $J_0(\underline{\boldsymbol{x}}_0)$ can be achieved by maximizing per step mutual information instead of per step conditional entropy. Equality in (4.4) holds for myopic sensor management, i.e., for $N = 1$ [57]. Additionally, employing mutual information is in particular beneficial in cases of additive measurement noise and where the dimension of the measurements is smaller than the dimension of the state (see Section 4.1.5). Here, the computational demand of (approximately) calculating the mutual information value can be significantly reduced compared to calculating the value of conditional entropy.

State Estimation and Density Representation

Besides performing state estimation for inferring the system state within the time horizon, the proposed information theoretic sensor management approach also relies on the prediction of measurement values for evaluating the expectation terms in (4.2) (see Section 4.1.2). Both

estimation tasks cannot be performed in closed form due to the employed nonlinear system and sensor models (2.1) and (2.3), respectively. Nevertheless, an approximate but accurate calculation of the occurring density functions is desirable. Especially the representation of multimodalities is of paramount importance, e.g., in case of measurement prediction, multiple modes indicate multiple meaningful measurement values that have to be considered for sensor configuration. Thanks to their universal approximation property, Gaussian mixtures are a convenient type of density functions and thus are employed here (see Appendix A.2). The hybrid density filter proposed in Chapter 6 offers the desired Gaussian mixture representations, where all necessary calculation can be performed in closed form.

According to Section 2.2, employing an accurate estimator for management purposes can be considered as mimicking the estimator for inference within the optimization time horizon. Admittedly, the estimator for planning has to be executed multiple times within the time horizon, which prevents approximations that are as accurate as for inference purposes. Instead, one has to trade off estimation quality against computation time. In case of the HDF, this can be achieved by reducing the number of Gaussian mixture components for planning, which leads to a decrease in the amount of computations for solving (4.1), while the estimation quality can still be kept at a meaningful level.

Solution of the Control Problem

Solving (4.1) in closed form for determining the optimal control policy is not possible for the considered setting due to the nonlinearity of the considered probabilistic models as well as the continuous-valued states and measurement values. More specifically, the mutual information step objectives $I(\,\cdot\,;\,\cdot\,)$ cannot be expressed as a linear function of the density function representing the system state, and the value-to-go, i.e., the expectation term in (4.1), is not piecewise linear convex as in stochastic control problems with finite state space [189]. But even for finite state spaces and finite measurement spaces, calculating the optimal configuration vector is still demanding [118, 140], which prevents the attempt of simply applying discretization of the state space and measurement space. Thus, the following sections are concerned with an approximate approach that trades computational burden off against incorporating as much information into the control as possible.

4.1.2 Virtual Measurements

Determining the optimal sensor configuration in (4.1) requires the incorporation of future, not yet available measurements due to the considered non-myopic optimization. This is achieved by means of the expectation $\mathrm{E}_{\underline{z}_n}$ with respect to all possible future measurement values $\underline{\boldsymbol{z}}_n$ at time step n of the time horizon. Calculating the density function of the future measurements can be achieved in a way similar to the state prediction (2.7) by propagating the predicted state estimate $\underline{\boldsymbol{x}}_n^p \sim f_n^p(\underline{x}_n)$ through the sensor model (2.3), which leads to the so-called *predicted measurement density*

$$f_n^z(\underline{z}_n|\underline{u}_n) = \int_{\mathbb{R}^{n_x}} f_n(\underline{z}_n|\underline{x}_n, \underline{u}_n) \cdot f_n^p(\underline{x}_n)\,\mathrm{d}\underline{x}_n \ . \tag{4.5}$$

Here, the conditional density $f_n(\underline{z}_n|\underline{x}_n, \underline{u}_n)$ corresponds to (2.10). The measurement prediction (4.5) can be efficiently calculated by means of the hybrid density filter (see Chapter 6), which results in Gaussian mixture representation of $f_n^z(\underline{z}_n|\underline{u}_n)$.

Calculating an Expected Density

Given the predicted measurement density, evaluating the expectation in (4.1) is still a difficult task due to the dependence of future state estimates on the measurements $\underline{\boldsymbol{z}}_n$. More specifically, at a specific time step n within the time horizon, predicted future states $\underline{\boldsymbol{x}}_m^p$ for $m > n$ depend on the configuration $\underline{u}_n$ and thus on the measurement $\underline{\boldsymbol{z}}_n$.

For the given continuous measurement space, it could be desirable to determine some kind of an expected predicted state that is independent of the measurements $\underline{\boldsymbol{z}}_n$ but still depends on the configuration $\underline{u}_n$, e.g., by taking the expectation over $\underline{\boldsymbol{x}}_{n+1}^p \sim f_{n+1}^p(\underline{x}_{n+1}) = f_{n+1}^x(\underline{x}_{n+1}|\underline{z}_n, \underline{u}_n)$ according to

$$\hat{f}_{n+1}^x(\underline{x}_{n+1}|\underline{u}_n) = \mathrm{E}_{\underline{z}_n}\left\{f_{n+1}^x(\underline{x}_{n+1}|\underline{\boldsymbol{z}}_n, \underline{u}_n)\right\} \ . \tag{4.6}$$

However, as shown in the following theorem, this leads to predicted states that are both independent of measurements *and* configurations. The non-myopic sensor management problem degenerates to a myopic one.

Theorem 4.1 (Expected Predicted Density) *Given the system state* $\underline{\boldsymbol{x}}_n \sim f_{n+1}^p(\underline{x}_n) = f_{n+1}(\underline{x}_{n+1}|\underline{z}_n, \underline{u}_n)$ *at time step* $n+1$ *and the predicted measurement density* $f_n^z(\underline{z}_n|\underline{u}_n)$ *according to* (4.5)*, taking the expectation according to* (4.6) *yields*

$$\hat{f}_{n+1}^x(\underline{x}_{n+1}|\underline{u}_n) = \mathrm{E}_{\underline{z}_n}\left\{f_{n+1}^x(\underline{x}_{n+1}|\underline{\boldsymbol{z}}_n, \underline{u}_n)\right\} = f_{n+1}^x(\underline{x}_{n+1}) \ .$$

Proof. Applying the Chapman-Kolmogorov equation (2.7) and Bayes' law (2.9) on the predicted density $f_{n+1}^x(\underline{x}_{n+1}|\underline{z}_n, \underline{u}_n)$ leads to

$$\begin{aligned}
\hat{f}_{n+1}^x(\underline{x}_{n+1}|\underline{u}_n) &= \mathrm{E}_{\underline{z}_n}\left\{f_{n+1}^x(\underline{x}_{n+1}|\underline{\boldsymbol{z}}_n, \underline{u}_n)\right\} \\
&= \int_{\mathbb{R}^{n_z}} f_n^z(\underline{z}_n|\underline{u}_n) \int_{\mathbb{R}^{n_x}} f_n^T(\underline{x}_{n+1}|\underline{x}_n) \cdot \frac{f_n(\underline{z}_n|\underline{x}_n, \underline{u}_n) \cdot f_n^p(\underline{x}_n)}{f_n^z(\underline{z}_n|\underline{u}_n)} \,\mathrm{d}\underline{x}_n \,\mathrm{d}\underline{z}_n \\
&= \int_{\mathbb{R}^{n_z}} \int_{\mathbb{R}^{n_x}} f_n^T(\underline{x}_{n+1}|\underline{x}_n) \cdot f_n(\underline{z}_n|\underline{x}_n, \underline{u}_n) \cdot f_n^p(\underline{x}_n) \,\mathrm{d}\underline{x}_n \,\mathrm{d}\underline{z}_n \\
&= f_{n+1}^x(\underline{x}_{n+1}) \ .
\end{aligned}$$

□

The Finite Case

An alternative would be to discretize the continuous measurement space, e.g., by employing Monte Carlo sampling of the predicted measurement density or by means of a grid representation of the continuous measurement space. For the resulting finite measurement space, a straightforward solution would be to calculate predicted states $\underline{\boldsymbol{x}}_m^p$, $m > n$, for each possible configuration and discrete measurement value. Taking the expectation then leads to the desired expected objective function value. However, this procedure is intractable for a large set of discretized measurement values and long time horizons.

Representative Virtual Measurements via Density Approximation

By employing straightforward discretization techniques, the resulting large set comprises many discretized measurement values that are less representative or redundant. Instead, the idea is now to provide an approximate solution to the expectation calculation by employing a finite and small set of so-called *virtual measurements*, which should be as representative and informative

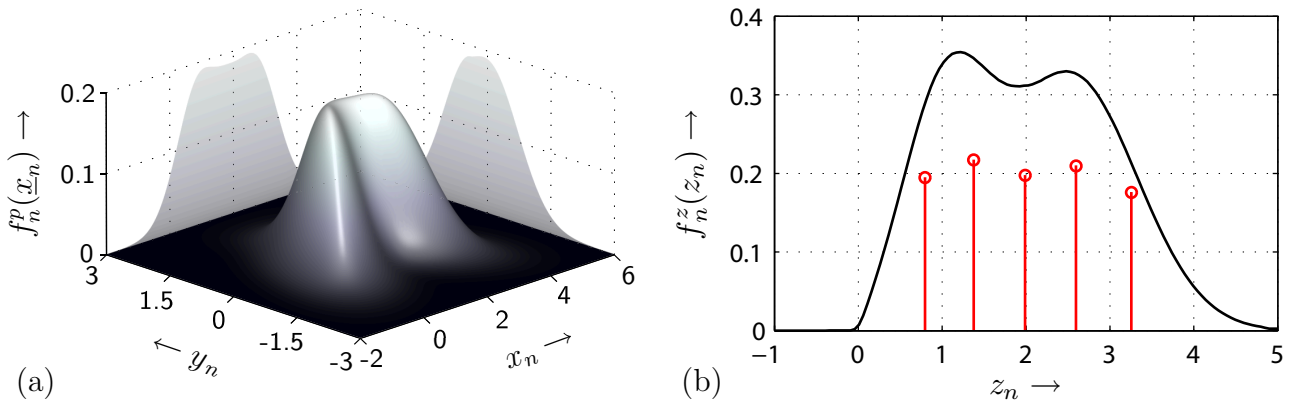


Figure 4.1: Virtual measurements approximating the predicted measurement density. (a) The predicted density at time step n, characterized by means of a Gaussian mixture. (b) The predicted measurement density $f_n^z(z_n)$ (black) that results from the predicted density by applying (4.5) and its approximation by five virtual measurements (red stems).

as possible. The term virtual measurement is used here, as the true measurement values are obviously not available within the time horizon. By means of the virtual measurements, it is possible the calculate the predicted state estimate $\underline{\boldsymbol{x}}_{n+1}^p$ for each virtual measurement value and take the expectation with respect to $\underline{\boldsymbol{z}}_n$.

In order to generate L representative virtual measurements, the predicted measurement density (4.5) is approximated by means of a Dirac mixture (see also Appendix A.2)

$$f_n^z(\underline{z}_n|\underline{u}^{(j)}) \approx \hat{f}_n^z(\underline{z}_n|\underline{u}^{(j)}) = \sum_{i=1}^{L} \omega_{n,i}^{(j)} \cdot \delta(\underline{z}_n - \hat{\underline{z}}_{n,i}^{(j)}) \tag{4.7}$$

for a given configuration $\underline{u}^{(j)} \in \mathcal{U}$. The positions $\hat{\underline{z}}_{n,i}^{(j)}$ of the Dirac delta distributions act as virtual measurement values and the weights $\omega_{n,i}^{(j)}$ approximately represent the probability of the individual virtual measurement.

The number of employed virtual measurements strongly impacts the computational demand of the entire sensor management approach. Each virtual measurement causes a predicted state estimate that is conditioned on the measurement value. Based on these state estimates, further decisions with respect to the next configuration have to be made. Thus, it is highly desirable to minimize the number of required virtual measurements, while preserving a meaningful representation of the predicted measurement density $f_n^z(\underline{z}_n|\underline{u}^{(j)})$. Accordingly, sampling methods like Monte Carlo sampling [9] or unscented transform [91] are inadequate. Monte Carlo sampling requires many virtual measurements for being representative, while the unscented transform and other LRKF techniques are specialized for Gaussian densities. However, the predicted measurement density is typically non-Gaussian and characterized by means of a Gaussian mixture in the proposed approach.

To obtained a high quality approximation that requires just a small number of Dirac components even in case of non-Gaussian measurement densities, the algorithm proposed in [163] is employed in the following. For scalar measurements, the components are carefully placed on the basis of a global optimization that progressively minimizes the Cramér–von Mises distance [24] between $f_n^z(\underline{z}_n|\underline{u})$ and $\bar{f}_n^z(\underline{z}_n|\underline{u})$. The whole procedure of determining virtual measurements is exemplified in the following.

Example 4.1: Virtual Measurements

For a particular time step n within the time horizon, the two-dimensional state $\underline{\boldsymbol{x}}_n = [\boldsymbol{x}_n, \boldsymbol{y}_n]^{\mathrm{T}}$ is represented by the predicted state estimate $\underline{\boldsymbol{x}}_n^p$. The Gaussian mixture density $f_n^p(\underline{x}_n)$ of $\underline{\boldsymbol{x}}_n^p$ is

depicted in Figure 4.1 (a). This is a typical density function occurring in target tracking applications with nonlinear system model, e.g. (4.23) on page 62, in which $\underline{\boldsymbol{x}}_n$ represents the two-dimensional target position.

It is assumed for this example that one distance measurement according to the sensor model (2.4) is performed, where the considered configuration is $\underline{u}_n = [0, 0]^{\mathrm{T}}$, i.e., the sensor is located in the origin. The measurement noise $\boldsymbol{v}_n$ is zero-mean Gaussian with standard deviation $\sigma_n^v = 0.05$ m. For obtaining $L = 5$ virtual measurements, at first the predicted measurement density $f_n^z(z_n|\underline{u}_n)$ needs to be calculated by applying the HDF on (4.5). The resulting predicted measurement density is illustrated in Figure 4.1 (b). Now, the Dirac mixture approximation algorithm [163] is applied, which leads to the five virtual measurements indicated by the stems in Figure 4.1 (b). It can be clearly seen that the Dirac delta distributions cover the regions where the strictly non-Gaussian predicted measurement density comprises most of its probability mass. The components are not equally weighted. Instead, the weights capture the local probability mass around the corresponding component. Hence, the second and forth component possess the largest weights. ■

For multidimensional measurement values, an extension of the approximation algorithm is introduced in [73, 97], where the components are placed in a greedy fashion in order to reduce the computational demand.

4.1.3 Recursive Objective Function Calculation

The optimal configuration $\underline{u}_k^*$ can be easily calculated employing (4.1) if the cumulative objective function $V_0(\underline{\boldsymbol{x}}_0^p, \underline{u}_0)$ is known. Thus, the main task in determining the configuration vector is to calculate the cumulative objective $V_0(\underline{\boldsymbol{x}}_0^p, \underline{u}_0)$. Due to the cumulative structure of the objective function and the consideration of future measurement values, this calculation has to be carried out recursively *backwards* in time by employing

$$\underline{u}_k^* = \arg \max_{\underline{u}_0} V_0(\underline{\boldsymbol{x}}_0^p, \underline{u}_0) = \arg \max_{\underline{u}_0} \left\{ g_0\left(\underline{\boldsymbol{x}}_0^p, \underline{u}_0\right) + \mathrm{E}_{\underline{z}_0}\left\{ J_1(\underline{\boldsymbol{x}}_1^p) | \underline{\boldsymbol{x}}_0^p, \underline{\boldsymbol{z}}_0, \underline{u}_0 \right\} \right\} . \tag{4.8}$$

Due to Bellman's principle of optimality [16], this recursion is equivalent to (4.1) and commences from

$$J_{N-1}(\underline{\boldsymbol{x}}_{N-1}^p) = \max_{\underline{u}_{N-1}} \left\{ g_{N-1}\left(\underline{\boldsymbol{x}}_{N-1}^p, \underline{u}_{N-1}\right) \right\}$$

for a given predicted state $\underline{\boldsymbol{x}}_{N-1}^p$ and configuration $\underline{u}_{N-1} \in \mathcal{U}$.

Based on the initial density function $\underline{\boldsymbol{x}}_0^p \sim f_0^p(\underline{x}_0)$ of the state estimate at time step k, the Gaussian mixture densities describing the system state $\underline{\boldsymbol{x}}_n^p$ within the time horizon can be calculated for various configuration sequences $\underline{u}_{0:n-1}$ and virtual measurement values $\hat{\underline{z}}_{0:n-1}$ using the HDF. Due to the finite number of configurations and measurement values, solving (4.8) leads to a tree structure as depicted in Figure 4.2. At each time step n, the tree branches for each configuration *and* each virtual measurement. Contrary to the open-loop control tree (see Figure 3.2), which merely possesses nodes representing state estimates, the closed-loop control tree additionally possesses levels with nodes for the predicted measurement densitys (light red nodes in Figure 4.2). Hence, the closed-loop control tree consists of $O(|\mathcal{U}|^N \cdot L^{N-1})$ possible pathes. Thanks to the well-directed selection of virtual measurements, the number of additional branches $O(L^{N-1})$ can be kept on a low level.

Since two different types of density functions, namely $f_n^p(\underline{x}_n)$ and $f_n^z(\underline{z}_n)$, appear in the search tree, the recursion (4.8) is partitioned into two different types of cumulative objective functions $J_n^p(\underline{\boldsymbol{x}}_n^p)$ and $J_n^z(\underline{\boldsymbol{x}}_n^p, \underline{u}_n)$, respectively, for clarity reasons.

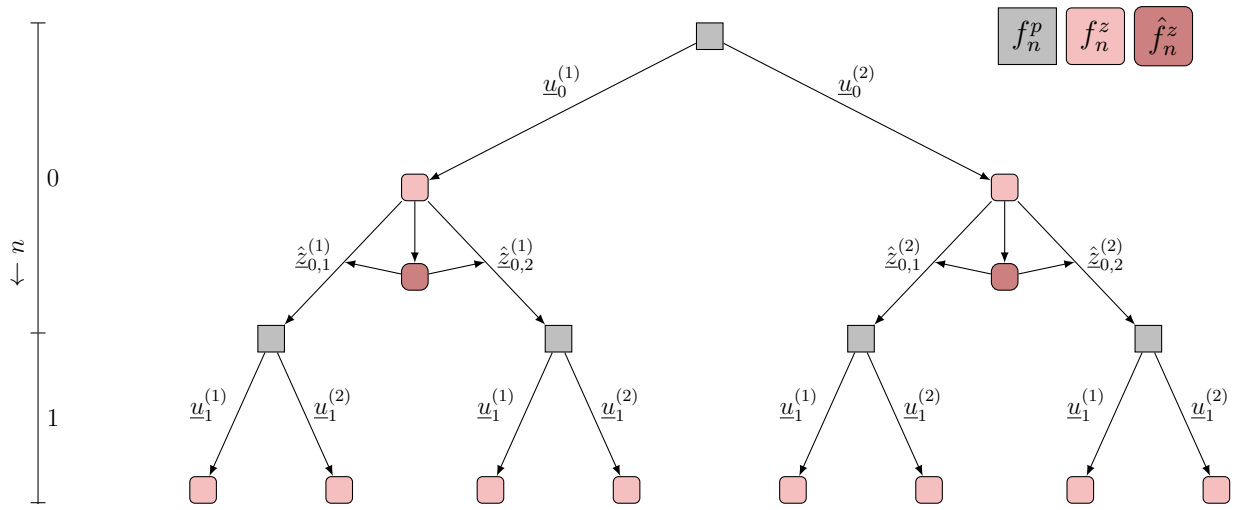


Figure 4.2: Search tree for $N = 2$ time steps, $|\mathcal{U}| = 2$ different configurations and $L = 2$ virtual measurements with root node $f_0^p(\underline{x}_0)$.

Calculation of $J_n^p(\underline{\boldsymbol{x}}_n^p)$

The cumulative objective function $J_n^p(\underline{\boldsymbol{x}}_n^p)$ determines the configuration vector that maximizes the sum of the step objective function $g_n(\underline{\boldsymbol{x}}_n^p, \underline{u}_n)$ and the cumulative objective $J_n^z(\underline{\boldsymbol{x}}_n^p, \underline{u}_n)$ of its successors as depicted in Figure 4.3 (a). Hence, $J_n^p(\underline{\boldsymbol{x}}_n^p)$ is given by

$$J_n^p(\underline{\boldsymbol{x}}_n^p) = \max_{\underline{u}_n} \left\{ g_n(\underline{\boldsymbol{x}}_n^p, \underline{u}_n) + J_n^z(\underline{\boldsymbol{x}}_n^p, \underline{u}_n) \right\} \ . \tag{4.9}$$

By comparing (4.9) with (4.8), it can be seen that the objective function $J_n^p(\underline{\boldsymbol{x}}_n^p)$ coincides with the actual cumulative objective functions $J_n(\underline{\boldsymbol{x}}_n^p)$ at time step n.

Calculation of $J_n^z(\underline{\boldsymbol{x}}_n^p, \underline{u}_n)$

The cumulative objective functions $J_n^z(\underline{\boldsymbol{x}}_n^p, \underline{u}_n)$ can be calculated on the basis of $J_{n+1}^p(\underline{\boldsymbol{x}}_{n+1}^p)$ as illustrated in Figure 4.3 (b), whereas the density function of $\underline{\boldsymbol{x}}_{n+1}^{p|\hat{\underline{z}}_n}$ depends on a particular virtual measurement $\hat{\underline{z}}_n$ or equivalently, on a particular branch of the tree. Hence, gray nodes of the search tree correspond to a measurement update step for incorporating the particular virtual measurement followed by a prediction step. As the occurrence of a measurement value is random to some degree, the cumulative objective $J_n^z(\underline{\boldsymbol{x}}_n^p, \underline{u}_n)$ is the expected value of its successors $J_{n+1}^p\left(\underline{\boldsymbol{x}}_{n+1}^{p|\hat{\underline{z}}_n}\right)$ with regard to the virtual measurements $\hat{\underline{\boldsymbol{z}}}_n \sim \hat{f}_n^z(\underline{z}_n|\underline{u}_n)$, i.e.,

$$J_n^z(\underline{\boldsymbol{x}}_n^p, \underline{u}_n) = \mathrm{E}_{\hat{\underline{z}}_n} \left\{ J_{n+1}^p \left(\underline{\boldsymbol{x}}_{n+1}^{p|\hat{\underline{z}}_n} \right) | \underline{\boldsymbol{x}}_n^p, \underline{u}_n \right\}$$

for $n < N-1$. Due to the Dirac mixture representation (4.1) of the virtual measurements, where the component weights correspond to the probability of a virtual measurement, $J_n^z(\underline{\boldsymbol{x}}_n^p, \underline{u}_n)$ can be written according to

$$J_n^z(\underline{\boldsymbol{x}}_n^p, \underline{u}_n) = \sum_{i=1}^{L} \omega_{n,i} \cdot J_{n+1}^p \left(\underline{\boldsymbol{x}}_{n+1}^{p|\hat{\underline{z}}_{n,i}} \right) \ .$$

For the final step $N-1$ of the time horizon, which corresponds to the leafs of the search tree, the terminal objective function is given by

$$J_{N-1}^z(\underline{\boldsymbol{x}}_{N-1}^p, \underline{u}_{N-1}) = 0 \ .$$

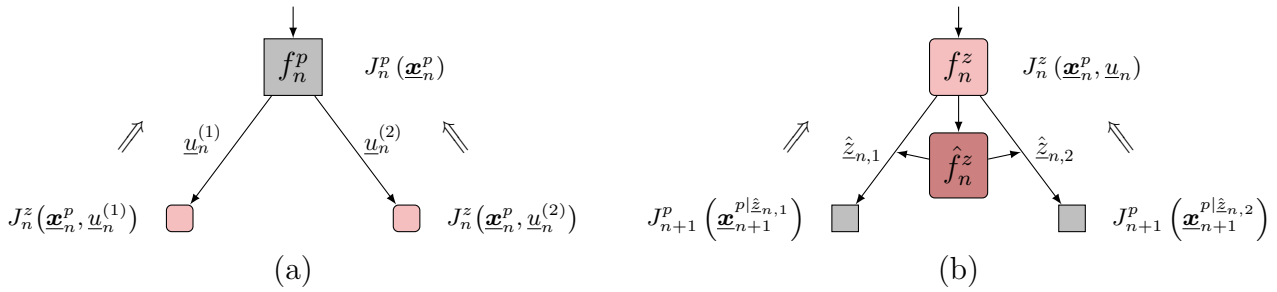


Figure 4.3: Detailed view on the levels of the closed-loop control tree of Figure 4.2 with regard to evaluating the cumulative objective functions (a) $J_n^p(\underline{\boldsymbol{x}}_n^p)$ and (b) $J_n^z(\underline{\boldsymbol{x}}_n^p, \underline{u}_n)$. The state estimate $\underline{\boldsymbol{x}}_{n+1}^{p|\hat{\underline{z}}_{n,i}}$ and thus, the value of $J_{n+1}^p\left(\underline{\boldsymbol{x}}_{n+1}^{p|\hat{\underline{z}}_{n,i}}\right)$ in (b) depends on the taken virtual measurement $\hat{\underline{z}}_{n,i}$.

This special terminal objective function allows for the structure of the mutual information objective (4.11), which can be evaluated at the final step $N-1$ without determination of a posterior state estimate $\underline{\boldsymbol{x}}_{N-1}^e$.

4.1.4 Optimal Pruning

As mentioned before, a straightforward calculation of the configuration vector $\underline{u}_k^*$, i.e., traversing the entire search tree, has exponential complexity. By employing more advanced tree search techniques, this computational demand can be significantly reduced while retaining optimality. One of these techniques is branch-and-bound (BB) pruning as utilized in Section 3.2.4. However, basic branch-and-bound pruning algorithms cannot be applied directly. In the considered sensor management problem, the tree consists not only of edges that represent configurations, which are deterministic decisions (as in the basic BB). There are also edges representing virtual measurements for which only a probabilistic representation is available. To consider this, the so-called *probabilistic branch-and-bound* (PBAB) algorithm is used, which ensures even in this particular case that the optimal solutions is always found. In the following, a brief review of PBAB is given. A detailed description of PBAB can be found in [187] and [230].

Since PBAB operates on negative objective function values for maximization, the step objectives are converted according to

$$\bar{g}_n(\underline{\boldsymbol{x}}_n^p, \underline{u}_n) = g_n(\underline{\boldsymbol{x}}_n^p, \underline{u}_n) - c \leq 0 \ , \tag{4.10}$$

where $c \in \mathbb{R}^+$ is a sufficiently large constant. Based on this modification, PBAB is able to assign an upper bound of the achievable cumulative objective to any visited node. Initially, the upper bound is set to 0 which is always a valid upper bound due to (4.10). In contrast to the open-loop approach employed in Chapter 3, the objective function is evaluated recursively backwards in time due to the dependence on future measurement values. Hence, whenever a new node is expanded, the upper bound values of the parent nodes are update recursively, i.e., the upper bound values are adapted by traversing backwards to the root of the tree. In doing so, an upper bound for a f_n^p-node becomes the exact cumulative objective value, if the largest sum of step objective value and objective value of a child is known exactly. The objective value of a f_n^z-node is exact, if all cumulative objective values of child nodes are known exactly.

For an efficient tree search, i.e., for pruning the tree, not necessarily all children of the f_n^p-nodes need to be fully expanded since the upper bound of a child can be smaller than an exactly known cumulative objective function value. In the best case, the child nodes of one f_n^z-node have significantly higher step objective values (mutual information values) than the

f_n^z-node's siblings' children. In this case, merely one subtree needs to be analyzed and the computational complexity is not exponential in the number of configurations $|\mathcal{U}|$.

4.1.5 Evaluation of the Step Objective Function

Albeit their approximation capabilities, utilizing Gaussian mixtures for characterizing the system states complicates the evaluation of the mutual information step objectives. According to (2.27), the mutual information term can be decomposed into two differential entropy terms, for which no analytical expression can be found in case of Gaussian mixture random vectors [116, 144]. However, the identity

$$I(\underline{\boldsymbol{x}}_n;\underline{\boldsymbol{z}}_n) = H(\underline{\boldsymbol{z}}_n) - H(\underline{\boldsymbol{z}}_n|\underline{\boldsymbol{x}}_n) \tag{4.11}$$

leads to an significant simplification compared to $I(\underline{\boldsymbol{x}}_n;\underline{\boldsymbol{z}}_n) = H(\underline{\boldsymbol{x}}_n) - H(\underline{\boldsymbol{x}}_n|\underline{\boldsymbol{z}}_n)$ in cases where the dimension of the measurement space is smaller than the dimension of the state space, which exemplary is the case in typical target tracking scenarios as the one considered in Example 2.1 and 2.2. Hence, the complexity of evaluating or approximating both entropy terms in (4.11) decreases compared to the entropy terms $H(\underline{\boldsymbol{x}}_n)$ and $H(\underline{\boldsymbol{x}}_n|\underline{\boldsymbol{z}}_n)$.

In this thesis, an approximation of the mutual information value (4.11) is provided by means of calculating an upper bound. This can be achieved by bounding the first entropy term $H(\underline{\boldsymbol{z}}_n)$ from above and by bounding the conditional entropy term $H(\underline{\boldsymbol{z}}_n|\underline{\boldsymbol{x}}_n)$ from below. The calculation of tight and computationally cheap bounds is discussed in the subsequent section. Calculating an upper bound of (4.11) can be further simplified, if the sensor is corrupted by additive white Gaussian mixture noise, i.e., the sensor model is given by

$$\underline{\boldsymbol{z}}_n = \underline{h}_n(\underline{\boldsymbol{x}}_n, \underline{u}_n) + \underline{\boldsymbol{v}}_n \ ,$$

with measurement noise according to

$$\underline{\boldsymbol{v}}_n \sim f_n^v(\underline{v}_n) = \sum_{i=1}^{L} \omega_{n,i}^v \cdot \mathcal{N}(\underline{v}_n; \underline{\hat{v}}_{n,i}, \mathbf{C}_{n,i}^v) \ .$$

For this case, the conditional entropy can be reformulated according to

$$H(\underline{\boldsymbol{z}}_n|\underline{\boldsymbol{x}}_n) = H(\underline{h}_n(\underline{\boldsymbol{x}}_n, \underline{u}_n) + \underline{\boldsymbol{v}}_n|\underline{\boldsymbol{x}}_n) = H(\underline{\boldsymbol{v}}_n|\underline{\boldsymbol{x}}_n) = H(\underline{\boldsymbol{v}}_n) \ .$$

Hence, it is sufficient to bound the differential entropy $H(\underline{\boldsymbol{v}}_n)$ of the noise vector $\underline{\boldsymbol{v}}_n$. For the special case of measurement noise represented by a single Gaussian density, the differential entropy $H(\underline{\boldsymbol{v}}_n)$ can even be evaluated in closed form according to

$$H(\underline{\boldsymbol{z}}_n|\underline{\boldsymbol{x}}_n) = H(\underline{\boldsymbol{v}}_n) = \tfrac{1}{2}\log|2\pi e \cdot \mathbf{C}_n^v| \ .$$

Hence, merely the entropy term $H(\underline{\boldsymbol{z}}_n)$ in (4.11) remains for approximation by means of a bound.

To obtain an upper bound on the conditional entropy in case on non-additive Gaussian mixture noise, one option could be to exploit the relation $H(\underline{\boldsymbol{z}}|\underline{\boldsymbol{x}}) = H(\underline{\boldsymbol{x}},\underline{\boldsymbol{z}}) - H(\underline{\boldsymbol{x}})$. The joint density $f(\underline{x},\underline{z})$ required for the entropy term $H(\underline{\boldsymbol{x}},\underline{\boldsymbol{z}})$ can be efficiently determined by means of the Gaussian mixture estimator proposed in Section 5.5. Alternatively, minimizing the conditional entropy $H(\underline{\boldsymbol{x}}|\underline{\boldsymbol{z}})$ can be considered instead, as discussed in Section 4.1.1. Here, merely one single entropy term needs to be approximated to the disadvantage of merely calculating an upper bound approximation of the cumulative objective function value.

4.2 Entropy Bounds

The Gaussian mixture representation of the state estimates complicates the evaluation of the mutual information objective function. Since mutual information needs to be evaluated for each f_n^p-node of the search tree, an efficient approximation of the mutual information value is inevitable. Here, it is sufficient to focus on approximating the differential entropy terms that form mutual information.

Besides being computationally cheap, an approximation of the entropy values additionally has to ensure meaningful approximation results. Both requirements are typically not fulfilled for direct approximation approaches. In some situation, these approximations may deviate from the true entropy value in an arbitrary fashion and hence, are of limited usefulness. Only a few of the existent approximations, for example computationally expensive random sampling, can be demonstrated to converge to the true value. An overview of common direct approximation approach is given in Section 4.2.1.

On this account, Section 4.2.2 and Section 4.2.3 are concerned with deriving novel tight lower and upper bounds for the differential entropy

$$H(\underline{\boldsymbol{x}}) = -\int_{\mathbb{R}^{n_x}} f(\underline{x}) \cdot \log f(\underline{x}) \, \mathrm{d}\underline{x} \ , \tag{4.12}$$

with $f(\underline{x}) = \sum_i \omega_i \cdot \mathcal{N}(\underline{x}; \underline{\hat{x}}_i, \mathbf{C}_i)$ being a Gaussian mixture with L components. By composing a lower bound of the first entropy term in (4.11) with an upper bound of the second term[1], a meaningful lower bound approximation to mutual information is available. This procedure is reasonable, if the computational demand of calculating the bounds is significantly lower as the computational demand for directly approximating the mutual information value. As it is shown, both bounds can be calculated in closed form, which facilitates an efficient approximation. In addition, refinement methods for the bounds are proposed in Section 4.2.4 in order to approach the true entropy and mutual information values, respectively.

The derived bounds are not restricted to efficiently evaluating objective functions for sensor management. In fact, the bounds are universally valid and thus applicable in further information-theoretic tasks like capacity calculation of communication channels [45], parameter estimation [39], image registration [182], and many others. Furthermore, providing a tight lower and upper bound of the entropy value allows deciding whether a direct approximation is meaningful or not, i.e., some kind of confidence interval is given.

4.2.1 Prior Work

The literature provides many methods for an approximate calculation of the entropy for Gaussian mixture random vectors. A brief overview over this methods is given before starting with the derivation of the entropy bounds.

Exact solutions for the entropy for Gaussian mixtures exists for the special cases of a mixture consisting of merely one component [81] and consisting of two components with equal weights and covariances [127]. Especially the solution

$$H(\underline{\boldsymbol{x}}) = \frac{1}{2} \log |2\pi e \cdot \mathbf{C}| \tag{4.13}$$

for the single Gaussian $f(\underline{x}) = \mathcal{N}(\underline{x}; \underline{\hat{x}}, \mathbf{C})$ is of special importance as it offers one of the most straightforward ways to approximate (4.12) for arbitrary mixtures. Here, $f(\underline{x})$ is replaced by the Gaussian density that exactly captures the first two moments of $f(\underline{x})$. Although this method

1 In case of additive Gaussian measurement noise, merely the lower bound of the first entropy term is necessary (see Section 4.1.5)

is very efficient, it merely provides an (albeit very loose) upper bound approximation to the entropy. A proof of this result can for example be found in [35, 45]. This bound is extensively exploited for the proposed novel lower and upper entropy bounds.

The only entropy approximation methods so far that generally converge to the true entropy value are given by Monte Carlo sampling or by local linearization in combination with repeated splitting of the Gaussian components [213]. In case of Monte Carlo sampling, the first factor in (4.12) is represented by a set of samples drawn i.i.d. from $f(\underline{x})$, which facilitates a point-wise evaluation of the logarithm. According to the strong law of large numbers, this approximation converges to the true entropy value as the number of samples goes to infinity. For the linearization and splitting approach, the components of the first factor in (4.12) are split analogously as described in Section 7.2.2 and the logarithm is then linearized around the mean of each Gaussian. For both approaches, a relatively large number of samples or splits has to be used in order to obtain a good approximation, which in turn is computationally demanding. In case of Monte Carlo sampling, randomization is used and thus, no deterministic approximation is provided, which complicates comparison and precludes classical optimization techniques like gradient descent for entropy minimization.

Deterministic sampling instead allows the use of far less sample points for a specific approximation quality. This is the idea the entropy approximation proposed in [63] is based on. By employing the unscented transform (see e.g. [91]) each Gaussian component of $f(\underline{x})$ in (4.12) is replaced by a set of regression points. However, in contrast to Monte Carlo sampling or component splitting, getting arbitrarily close to the true entropy value is not longer possible as the number of regression points is constant.

In [59, 60], an approximation is developed by replacing (4.12) with the squared integral difference between $f(\underline{x})$ and a uniform density. This method can be regarded as linearizing (4.12) with a second-order Taylor-series expansion around the uniform density. This method turns out to be computationally demanding and often inaccurate. Furthermore, it is only applicable for the special case of Gaussian mixtures with axis-aligned components.

4.2.2 Lower Bound

A lower bound of (4.12) can be obtained by employing an upper bound of the Kullback-Leibler divergence (2.30) between two Gaussian mixtures, which is derived in [81]. In doing so, the logarithm is moved outside the integral since $-\log x$ is convex in x and thus, Jensen's inequality (see e.g. [45]) holds. Only an integral of a product of two Gaussian densities remains, which has a well-known closed-form solution.

Theorem 4.2 (Basic Lower Bound) *A lower bound $H_l(\underline{\boldsymbol{x}})$ of (4.12) is given by*

$$H_l(\underline{\boldsymbol{x}}) = -\sum_{i=1}^{L} \omega_i \cdot \log\left(\sum_{j=1}^{L} \omega_j \cdot z_{i,j}\right) , \tag{4.14}$$

with $z_{i,j} = \mathcal{N}\left(\underline{\hat{x}}_i; \underline{\hat{x}}_j, \mathbf{C}_i + \mathbf{C}_j\right)$.

Proof. With $-\log \mathrm{E}\{f(\underline{\boldsymbol{x}})\} \leq \mathrm{E}\{-\log f(\underline{\boldsymbol{x}})\}$ according to Jensen's inequality it follows

$$\begin{aligned} H(\underline{\boldsymbol{x}}) &\geq -\log\left(\int_{\mathbb{R}^{n_x}} f(\underline{x})^2 \,\mathrm{d}\underline{x}\right) =: H_{l_2}(\underline{\boldsymbol{x}}) \\ &= -\log\left(\sum_{i}^{L} \omega_i \cdot \int_{\mathbb{R}^{n_x}} \mathcal{N}(\underline{x}; \underline{\hat{x}}_i, \mathbf{C}_i) \cdot f(\underline{x}) \,\mathrm{d}\underline{x}\right) . \end{aligned} \tag{4.15}$$

Moving the logarithm inside the sum over i by again applying Jensen's inequality results in the desired lower bound

$$\begin{aligned} H_{l_2}(\underline{\boldsymbol{x}}) &\leq -\sum_{i}^{L} \omega_i \cdot \log\left(\int_{\mathbb{R}^{n_x}} \mathcal{N}(\underline{x};\underline{\hat{x}}_i,\mathbf{C}_i)\cdot f(\underline{x})\,\mathrm{d}\underline{x}\right) = H_l(\underline{\boldsymbol{x}}) \\ &= -\sum_{i}^{L} \omega_i \cdot \log\Bigg(\sum_{j=1}^{L} \omega_j \cdot \underbrace{\int_{\mathbb{R}^{n_x}} \mathcal{N}(\underline{x};\underline{\hat{x}}_i,\mathbf{C}_i)\cdot\mathcal{N}(\underline{x};\underline{\hat{x}}_j,\mathbf{C}_j)\,\mathrm{d}\underline{x}}_{=z_{i,j}}\Bigg)\,. \end{aligned} \tag{4.16}$$

The constant $z_{i,j} = \int_{\mathbb{R}^{n_x}} \mathcal{N}(\underline{x};\underline{\hat{x}}_i,\mathbf{C}_i)\cdot\mathcal{N}(\underline{x};\underline{\hat{x}}_j,\mathbf{C}_j)\,\mathrm{d}\underline{x} = \mathcal{N}\left(\underline{\hat{x}}_i;\underline{\hat{x}}_j,\mathbf{C}_i+\mathbf{C}_j\right)$ follows from the multiplication of two Gaussians (see (A.2) in the Appendix A.1).

A further utilization of Jensen's inequality on (4.16) for moving the logarithm inside the integral according to

$$H_l(\underline{\boldsymbol{x}}) \leq -\sum_{i=1}^{L} \omega_i \cdot \int \mathcal{N}(\underline{x};\underline{\hat{x}}_i,\mathbf{C}_i)\cdot\log f(\underline{x})\,\mathrm{d}\underline{x} = H(\underline{\boldsymbol{x}})$$

concludes the proof. □

The result in (4.14) can be interpreted as applying Jensen's inequality separately to each Gaussian $\mathcal{N}(\underline{x};\underline{\hat{x}}_i,\mathbf{C}_i)$ of $f(\underline{x})$ not being argument of the logarithm. If Jensen's inequality is applied on the entire Gaussian mixture $f(\underline{x})$ instead, the alternative lower bound $H_{l_2}(\underline{\boldsymbol{x}})$ from (4.15) results, which was proposed in [31]. However, as shown in the proof, this bound is always smaller than $H_l(\underline{\boldsymbol{x}})$. The computational complexity of both bounds is quadratic in the number of Gaussians L.

4.2.3 Upper Bound

The upper bound derived next is computationally cheap as its complexity is merely linear in the number of Gaussian components L. It can be interpreted as a weighted sum of discrete entropies and individual entropies (4.13) of Gaussian components.

Theorem 4.3 (Basic Upper Bound) *An upper bound $H_u(\underline{\boldsymbol{x}})$ of (4.12) is given by*

$$H_u(\underline{\boldsymbol{x}}) = \sum_{i=1}^{L} \omega_i \cdot \Big(\underbrace{-\log\omega_i}_{\text{discrete entropies}} + \underbrace{\tfrac{1}{2}\log\left|2\pi e\cdot\mathbf{C}_i\right|}_{\text{Gaussian entropies}}\Big)\,, \tag{4.17}$$

where the individual Gaussian entropies coincide with (4.13).

Proof. By separating the i-th component of the mixture $f(\underline{x})$ that is argument of the logarithm, the entropy for $\underline{\boldsymbol{x}}$ can be written as

$$\begin{aligned} H(\underline{\boldsymbol{x}}) &= -\int_{\mathbb{R}^{n_x}} \sum_{i=1}^{L} \omega_i\cdot\mathcal{N}(\underline{x};\underline{\hat{x}}_i,\mathbf{C}_i)\cdot\log\left(\sum_{j=1}^{L}\omega_j\cdot\mathcal{N}(\underline{x};\underline{\hat{x}}_j,\mathbf{C}_j)\right)\mathrm{d}\underline{x} \\ &= -\sum_{i=1}^{L}\omega_i\int_{\mathbb{R}^{n_x}} \mathcal{N}(\underline{x};\underline{\hat{x}}_i,\mathbf{C}_i)\cdot\log\Big(\omega_i\cdot\mathcal{N}(\underline{x};\underline{\hat{x}}_i,\mathbf{C}_i)\cdot\big(1+\epsilon_i(\underline{x})\big)\Big)\,\mathrm{d}\underline{x} \\ &= -\sum_{i=1}^{L}\omega_i\int_{\mathbb{R}^{n_x}} \mathcal{N}(\underline{x};\underline{\hat{x}}_i,\mathbf{C}_i)\cdot\Big(\log\big(\omega_i\cdot\mathcal{N}(\underline{x};\underline{\hat{x}}_i,\mathbf{C}_i)\big)+\log\big(1+\epsilon_i(\underline{x})\big)\Big)\,\mathrm{d}\underline{x}\,, \end{aligned} \tag{4.18}$$

where

$$\epsilon_i(\underline{x}) = \frac{\sum_{\substack{j=1 \\ i \neq j}}^{L} \omega_j \cdot \mathcal{N}(\underline{x}; \hat{\underline{x}}_j, \mathbf{C}_j)}{\omega_i \cdot \mathcal{N}(\underline{x}; \hat{\underline{x}}_i, \mathbf{C}_i)} \ . \tag{4.19}$$

Since $\log(1 + \epsilon_i(\underline{x}))$ in (4.18) is always non-negative, neglecting it yields the desired upper bound. □

Typically, the upper bound (4.17) is significantly closer to the true entropy value than the well-known bound given by a single Gaussian that matches the first two moments of $f(\underline{x})$. Furthermore, the bound is exact for the single Gaussian case and in cases, where the Gaussian components of $f(\underline{x})$ are well separated, i.e., the shared support of the components in $f(\underline{x})$ becomes negligible[2].

4.2.4 Bound Refinements

In this section, refinement methods for the previously introduced bounds are proposed in order to further approach the true entropy value by spending additional computations.

Upper Bound Refinement

Even if $f(\underline{x})$ has a large number of components, the shape of the Gaussian mixture is often not that complex. For example, a mode of $f(\underline{x})$ that is represented by several Gaussian components might be adequately approximated by a single Gaussian. It is also common to have a Gaussian mixture composed of several (almost) separated clusters of Gaussian components, where each cluster can be adequately represented by a single Gaussian. As shown below, merging several components of $f(\underline{x})$ to a single Gaussian allows calculating a further upper bound of the entropy.

Theorem 4.4 (Upper Bound by Merging Gaussians) *Given a Gaussian mixture random vector $\underline{\boldsymbol{x}} \sim f(\underline{x})$, where the mixture $f(\underline{x})$ is divided into two mixtures according to $f(\underline{x}) = f_1(\underline{x}) + f_2(\underline{x})$. Replacing $f_1(\underline{x})$ by a weighted Gaussian $\tilde{f}_1(\underline{x}) := \omega \cdot \mathcal{N}(\underline{x}; \hat{\underline{x}}_1, \mathbf{C}_1)$ that matches the first two moments of $f_1(\underline{x})$, where $\omega = \int f_1(\underline{x}) \, \mathrm{d}\underline{x}$, yields a new mixture $\tilde{f}(\underline{x}) = \tilde{f}_1(\underline{x}) + f_2(\underline{x})$ with*

$$H(\underline{\boldsymbol{x}}) \leq -\int \tilde{f}(\underline{x}) \cdot \log \tilde{f}(\underline{x}) \, \mathrm{d}\underline{x} =: \tilde{H}_u(\underline{\boldsymbol{x}}) \tag{4.20}$$

and thus, applying Theorem 4.3 on (4.20) *provides an easily computable upper bound for the entropy of $\underline{\boldsymbol{x}}$.*

Proof.

The inequality in (4.20) is equivalent to $\tilde{H}_u(\underline{\boldsymbol{x}}) - H(\underline{\boldsymbol{x}}) \geq 0$. Inserting the components for $f(\underline{x})$ as well as $\tilde{f}(\underline{x})$ and resorting yields

$$\begin{aligned} 0 &\leq -\int_{\mathbb{R}^{n_x}} \tilde{f}(\underline{x}) \cdot \log \tilde{f}(\underline{x}) \, \mathrm{d}\underline{x} + \int_{\mathbb{R}^{n_x}} f(\underline{x}) \cdot \log f(\underline{x}) \, \mathrm{d}\underline{x} \\ &= -\int_{\mathbb{R}^{n_x}} \tilde{f}_1(\underline{x}) \cdot \log \tilde{f}(\underline{x}) \, \mathrm{d}\underline{x} + \int_{\mathbb{R}^{n_x}} f_1(\underline{x}) \cdot \log f(\underline{x}) \, \mathrm{d}\underline{x} + \int_{\mathbb{R}^{n_x}} f_2(\underline{x}) \cdot \underbrace{\left(\log f(\underline{x}) - \log \tilde{f}(\underline{x})\right)}_{=\log \frac{f(\underline{x})}{\tilde{f}(\underline{x})}} \mathrm{d}\underline{x} \ . \end{aligned}$$

2 This corresponds to the case, where ϵ_i in (4.19) approaches zero.

Algorithm 3 $\tilde{H}_u(\underline{\boldsymbol{x}}) \leftarrow \text{RefineUpperBound}(f(\underline{x}))$

1: $\tilde{H}_u(\underline{\boldsymbol{x}}) \leftarrow \text{UpperBound}(f(\underline{x}))$ // According to (4.17)
2: **while** Number of components of $f(\underline{x}) > 1$ **do**
3: $\quad \tilde{f}(\underline{x}) \leftarrow \text{Merge}(f(\underline{x}))$
4: $\quad H_{\text{tmp}} \leftarrow \text{UpperBound}(\tilde{f}(\underline{x}))$ // According to (4.17)
5: $\quad \tilde{H}_u(\underline{\boldsymbol{x}}) \leftarrow \min\{\tilde{H}_u(\underline{\boldsymbol{x}}), H_{\text{tmp}}\}$
6: $\quad f(\underline{x}) \leftarrow \tilde{f}(\underline{x})$
7: **end while**

Adding zero, with $0 = \int f_1(\underline{x}) \log \frac{f(\underline{x})}{\tilde{f}(\underline{x})} \, \mathrm{d}\underline{x} - \int f_1(\underline{x}) \log \frac{f(\underline{x})}{\tilde{f}(\underline{x})} \, \mathrm{d}\underline{x}$ yields

$$0 \leq \int_{\mathbb{R}^{n_x}} \left(f_1(\underline{x}) - \tilde{f}_1(\underline{x}) \right) \cdot \log \tilde{f}(\underline{x}) \, \mathrm{d}\underline{x} + \underbrace{\int_{\mathbb{R}^{n_x}} f(\underline{x}) \log \frac{f(\underline{x})}{\tilde{f}(\underline{x})} \, \mathrm{d}\underline{x}}_{\text{(KLD, constant)} \geq 0} . \tag{4.21}$$

The second summand corresponds to the Kullback-Leibler divergence (2.30), which is non-negative. Hence, it is sufficient to further concentrate on the first summand. Here, separating $\tilde{f}_1(\underline{x})$ and $f_2(\underline{x})$ from $\tilde{f}(\underline{x})$ in the logarithm similarly to the proof of Theorem 4.3 leads to

$$\int_{\mathbb{R}^{n_x}} \left(f_1(\underline{x}) - \tilde{f}_1(\underline{x}) \right) \cdot \log \tilde{f}(\underline{x}) \, \mathrm{d}\underline{x} \geq \int_{\mathbb{R}^{n_x}} \left(f_1(\underline{x}) - \tilde{f}_1(\underline{x}) \right) \cdot \log \tilde{f}_1(\underline{x}) \, \mathrm{d}\underline{x} = 0 .$$

The equality to zero follows from the fact that $f_1(\underline{x})$ and $\tilde{f}_1(\underline{x})$ yield the same first two moments of the quadratic form $\log \tilde{f}_1(\underline{x})$ (see [45]). Hence, both summands in (4.21) are greater or equal to zero, which concludes the proof. □

It is important to note that Theorem 4.4 and (4.20), respectively, provide a family of upper bounds: All possible combinations of merged and unmerged Gaussian components give an upper bound. Obviously, the better the merged components can be represented by a single Gaussian, the tighter the upper bound provided by Theorem 4.4 is. A lowest upper bound will be one that merges clusters of Gaussians that are approximately Gaussian-shaped and does not merge well-separated components. In this case, the entropy value contribution of a merged cluster to the bound (4.17) is close to the entropy of the original (unmerged) mixture and thus potentially lower than the contribution of the individual Gaussians of the original mixture to the bound.

Instead of evaluating the whole family of bounds in a brute-force fashion for obtaining the lowest upper bound, a more efficient algorithm is proposed. According to Algorithm 3, Gaussian components of the mixture $f(\underline{x})$ are successively merged in order to identify Gaussian-shaped clusters (line 3). Afterwards, the corresponding upper bound is calculated (line 4) and compared with the currently lowest upper bound (line 5).

Methods that can be used for merging in line 3 are manifold. They differ in the distance measure used for identifying similar Gaussian components in $f(\underline{x})$ and the number of components merged in one step. Merging-based Gaussian mixture reduction methods like Salmond's clustering algorithm [155] or Runnall's reduction method [153] typically provide a good trade-off between computational complexity and accuracy in identifying Gaussian-shaped clusters. Here, Runnall's method is employed, where at each step two components are merged. The distance measure of this method is based on the Kullback-Leibler divergence (2.30), which is scale invariant and thus is an ideal measure for Gaussian mixture reduction purposes.

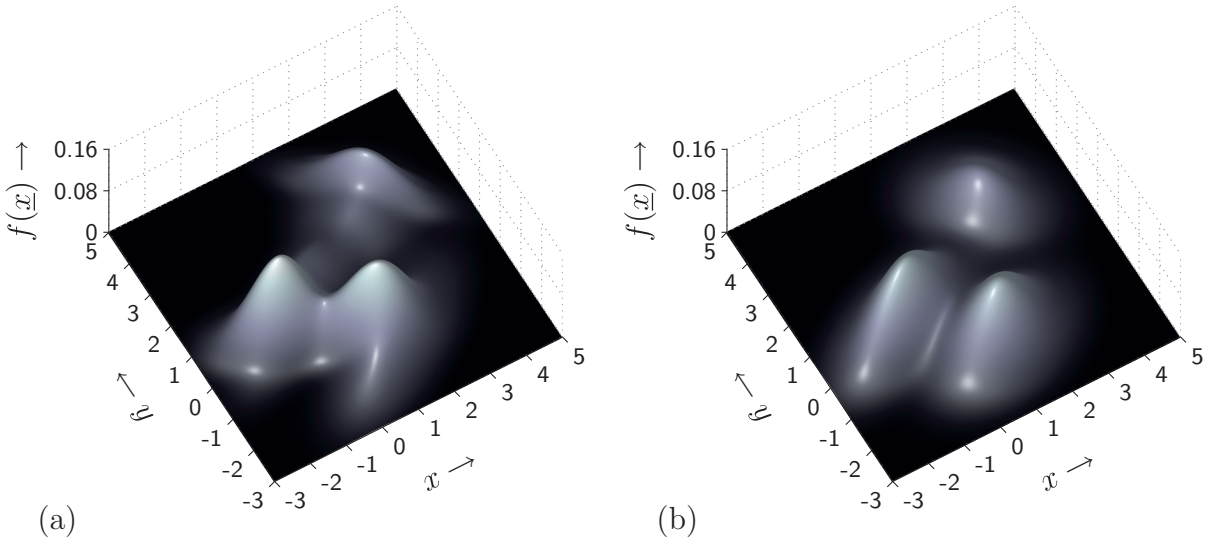


Figure 4.4: (a) Bivariate Gaussian mixture with $L = 6$ components. (b) Gaussian mixture with three components that yields the lowest upper bound by applying Algorithm 3 on the mixture in (a).

Example 4.2: Upper Bound Refinement

In this example, the functionality of Algorithm 3 is demonstrated by applying it to a two-dimensional random vector $\underline{\boldsymbol{x}}$ represented by the six component Gaussian mixture depicted in Figure 4.4 (a). The sequence of upper bounds generated by Algorithm 3 is listed in Table 4.1. Accordingly, the lowest upper bound is calculated at step four, which is also the return value of Algorithm 3. As depicted in Figure 4.4 (b), this bound corresponds to a Gaussian mixture reduced to three components, which is sufficient for representing the three modes of the original mixture. Comparing with the true entropy value $H(\underline{\boldsymbol{x}}) = 3.1220$, it can be stated that the resulting bounding value is tight. ■

Table 4.1: Intermediate steps of Algorithm 3 for refining the upper entropy bound for the Gaussian mixture depicted in Figure 4.4 (a).

step	1	2	3	4	5	6
$\tilde{H}_u(\underline{\boldsymbol{x}})$	3.671	3.601	3.463	3.329	3.467	3.594
# Gaussians	6	5	4	3	2	1

Lower Bound Refinement

By employing the dual operation to merging it is also possible to refine the lower bound given by (4.14). Here, a Gaussian component of the Gaussian mixture is replaced by a mixture of Gaussians, whereas the mean and covariance of the original Gaussian component are preserved. This procedure is referred to as *splitting* a single Gaussian into many. As shown in the following theorem, any moment-preserving splitting of Gaussian components provides a lower bound on the true entropy.

Corollary 4.1 (Lower Bound by Splitting Gaussians) *Given a Gaussian mixture random vector $\underline{\boldsymbol{x}} \sim f(\underline{x})$, where the mixture $f(\underline{x})$ is divided into two mixtures according to $f(\underline{x}) = f_1(\underline{x}) + f_2(\underline{x})$, with $f_1(\underline{x}) = \omega \cdot \mathcal{N}(\underline{x}; \hat{\underline{x}}, \mathbf{C})$ being a weighted single Gaussian. Replacing $f_1(\underline{x})$ by a weighted Gaussian mixture $\tilde{f}_1(\underline{x}) = \sum_{i=1}^{M} \omega_i \cdot \mathcal{N}(\underline{x}; \hat{\underline{x}}_i, \mathbf{C}_i)$ that matches the first two moments*

Algorithm 4 $\tilde{H}_l(\underline{\boldsymbol{x}}) \leftarrow \text{RefineLowerBound}(f(\underline{x}))$

1: $\tilde{H}_l(\underline{\boldsymbol{x}}) \leftarrow \text{LowerBound}(f(\underline{x}))$ // According to (4.14)
2: **repeat**
3: $\quad \tilde{f}(\underline{x}) \leftarrow \text{Split}(f(\underline{x}))$
4: $\quad H_{\text{tmp}} \leftarrow \text{LowerBound}(\tilde{f}(\underline{x}))$ // According to (4.14)
5: $\quad \tilde{H}_l(\underline{\boldsymbol{x}}) \leftarrow \max\{\tilde{H}_l(\underline{\boldsymbol{x}}), H_{\text{tmp}}\}$
6: $\quad f(\underline{x}) \leftarrow \tilde{f}(\underline{x})$
7: **until** termination condition fulfilled // E.g. number of loops

of $f_1(\underline{x})$*, where* $\omega = \sum_i \omega_i$*, yields a new mixture* $\tilde{f}(\underline{x}) = \tilde{f}_1(\underline{x}) + f_2(\underline{x})$ *with*

$$H(\underline{\boldsymbol{x}}) \geq -\int_{\mathbb{R}^{n_x}} \tilde{f}(\underline{x}) \cdot \log \tilde{f}(\underline{x}) \, \mathrm{d}\underline{x} =: \tilde{H}_l(\underline{\boldsymbol{x}}) \ . \tag{4.22}$$

Proof. This result can be shown in analogy to the proof of Theorem 4.4 by switching the role of $f(\underline{x})$ and $\tilde{f}(\underline{x})$. □

Applying Corollary 4.2 on (4.22) yields a new analytic lower bound for the entropy of $\underline{\boldsymbol{x}}$. Algorithm 4 represents a basic procedure for refining the lower bound of the entropy value. Similarly to the upper bound refinement introduced before, selecting appropriate Gaussians for replacement in line 3 is important for improving the lower bound accuracy. Moment-preserving splitting is a more difficult problem than merging, however, since many free parameters, i.e., weights, mean vectors and covariance matrices, have to be calculated. A straightforward but also computationally demanding way is to recursively split any Gaussian along any principal axes of the corresponding covariance ellipsoid. The performance of this procedure is illustrated in the following example. For a detailed description of splitting see for example Section 7.2.2.

Example 4.3: Lower Bound Refinement

Consider again the two-dimensional random vector $\underline{\boldsymbol{x}}$ of Example 4.2. Its Gaussian mixture density comprising six components is depicted in Figure 4.4 (a). For refining the lower bound of the entropy of $\underline{\boldsymbol{x}}$, each Gaussian is recursively split into two Gaussians for both principal axes in line 3 of Algorithm 4. Hence, one Gaussian is replaced by a mixture of four Gaussians. Then, splitting is again applied to any component of the resulting Gaussian mixture and so on. This procedure is repeated four times. The resulting lower bound values according to (4.14) are listed in Table 4.2, where the second column corresponds to the bounding value by applying (4.14) on the original mixture. It can be observed, that for an increased number of components, the bounding value approaches the true entropy value $H(\underline{\boldsymbol{x}}) = 3.1220$. Admittedly, the number of components and thus the computation time grows exponentially. To face this problem, it is important to note that many Gaussians are split unnecessarily as the splitting result does not contribute noticeably. ■

Table 4.2: Refinement of the lower entropy bound by repeated component splitting.

step	0	1	2	3	4
$\tilde{H}_l(\underline{\boldsymbol{x}})$	2.8571	2.9853	3.0129	3.0423	3.0713
# Gaussians	6	24	96	384	1536

Hence, a more directed selection of Gaussians for replacement in the context of lower bound refinement would lead to accurate bounding values, while the computations can be kept on a low level. The derivation of such a directed splitting scheme is content of future work.

4.3 Simulation Results

The effectiveness of the proposed information theoretic sensor management approach is evaluated by means of a simulation from the field of target tracking via a sensor network. The target is assumed to be a vehicle with differential drive kinematics. Hence, the system model is given by

$$\underline{\boldsymbol{x}}_{k+1} = \underline{\boldsymbol{x}}_k + T \cdot \begin{bmatrix} v_k \cdot \cos(\boldsymbol{\phi}_k) \\ v_k \cdot \sin(\boldsymbol{\phi}_k) \\ \alpha_k \end{bmatrix} + \underline{\boldsymbol{w}}_k \ , \tag{4.23}$$

where the three-dimensional system state $\underline{\boldsymbol{x}}_k$ comprises the position $[\boldsymbol{x}_k, \boldsymbol{y}_k]^{\mathrm{T}}$ and the orientation $\boldsymbol{\phi}_k$. The sampling time T is set to be 1 s, the velocity v_k is set to be 5 m/s, and the steering angle α_k is set to be 0 rad. The additive noise $\underline{\boldsymbol{w}}_k$ is zero-mean Gaussian with covariance matrix $\mathbf{C}_k^w = \mathrm{diag}([0.01\ \mathrm{m}^2, 0.01\ \mathrm{m}^2, (5\pi/180)^2\ \mathrm{rad}^2])$. The initial target state $\underline{\boldsymbol{x}}_0$ is characterized by means of Gaussian density with mean $\underline{\boldsymbol{x}}_0 = [0\ \mathrm{m}, 0\ \mathrm{m}, 0\ \mathrm{rad}]^{\mathrm{T}}$ and covariance $\mathbf{C}_0 = \mathrm{diag}([25\ \mathrm{m}^2, 25\ \mathrm{m}^2, (5\pi/180)^2\ \mathrm{rad}^2])$.

The Sensor Network

The vehicle is driving through a small sensor network, similar to the network considered in Section 3.3.5, where only one out of five sensors is allowed to perform a measurement per time step. The position $[x^u, y^u]^{\mathrm{T}}$ of each sensor is depicted in Figure 4.5 (a). For this sensor scheduling problem, the set of configurations $\mathcal{U} = \{1, 2, \ldots, 5\}$ represents the sensor indices. For the three bearing sensors with indices $u \in \{1, 4, 5\}$, the sensor model accords to (3.15), where the measurement noise $\boldsymbol{v}_k^u$ is zero-mean Gaussian with standard deviation $\sigma_k^{v,u} = 5\pi/180$ rad. The sensor network further comprises two signal strength sensors with indices $u \in \{2, 3\}$ and sensor model

$$\boldsymbol{z}_k^u = \frac{c}{d + (\boldsymbol{x}_k - x^u)^2 + (\boldsymbol{y}_k - y^u)^2} + \boldsymbol{v}_k^u \ . \tag{4.24}$$

This sensor model relates the position $[\boldsymbol{x}_k, \boldsymbol{y}_k]^{\mathrm{T}}$ of the vehicle to the relative signal strength $\boldsymbol{z}_k^u$ according to the free-space propagation model [78]. It allows localizing by means of measuring the signal strength of electromagnetic signals emitted by the vehicle. Even for additive noise, the sensor model (4.24) incorporates to some degree distance-dependencies of the measurements, which is not the case for direct distance measurements according to the sensor model (3.16). Here, both constants c and d are set to be 400 and the measurement noise $\boldsymbol{v}_k^u$ is zero-mean Gaussian with standard deviation $\sigma_k^{v,u} = 0.03$.

Thanks to the additive Gaussian noise terms of the sensor models, merely the first entropy term $H(\underline{\boldsymbol{z}}_n)$ of the mutual information step objective (4.11) needs to be bounded from below, while the conditional entropy $H(\underline{\boldsymbol{z}}_n | \underline{\boldsymbol{x}}_n^p)$ can be determined exactly.

Sensor Managers and Tracking Performance

As estimator for inference and planning the HDF is employed, whereas the number of components is $11^2 \cdot 21$ for inference and 11^3 for planning purposes. Hence, less components are used for state estimation during planning in order to reduce the computation load due to the multiple execution of the HDF. Furthermore, three virtual measurements are calculated for measurement anticipation.

For this setup, 75 Monte Carlo simulation runs are performed for three different information theoretic sensor managers (ITSMs) with time horizon length $N \in \{1, 2, 3\}$. The root mean

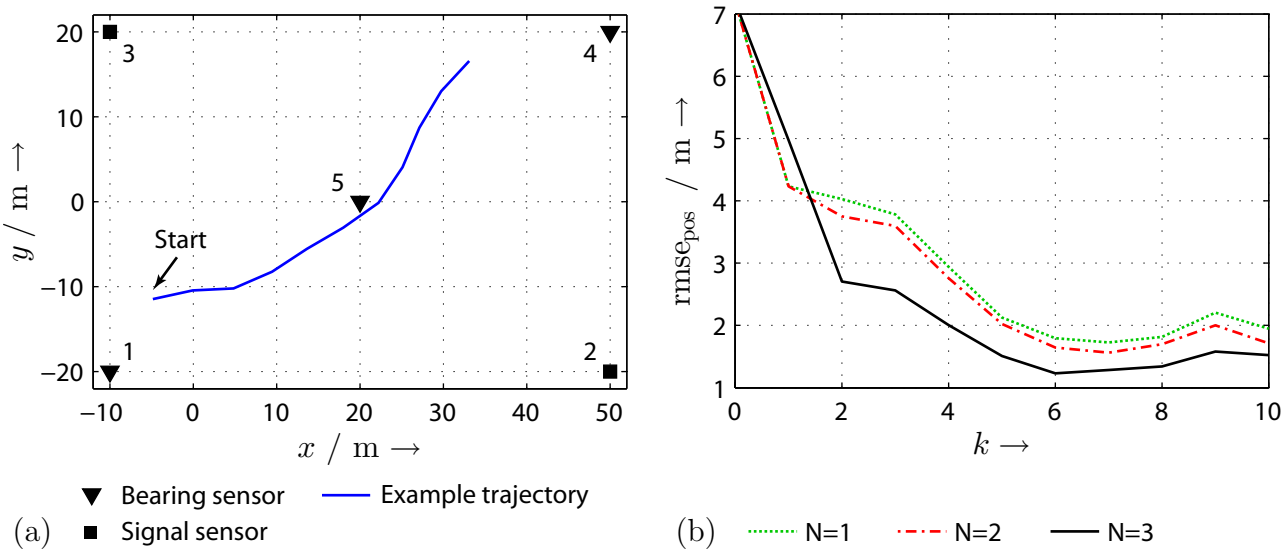


Figure 4.5: (a) Simulation setup and example trajectory of the vehicle. (b) Rmse over 75 Monte Carlo simulation runs for three different information theoretic sensor managers.

square errors (rmse) of the position estimates over all simulation runs for the first ten time steps are depicted in Figure 4.5 (b). As expected, the tracking performance increases with an increasing length of the time horizon, whereas the performance gain is more significant for $N = 3$. For $N = 1$ and $N = 2$, the rmse is similar. At time step one, both sensor managers always select the same sensor for measurement. Then, the manager with $N = 2$ slightly benefits from the longer lookahead. In contrast, the sensor manager with $N = 3$, takes a temporarily loss in performance at time step one by selecting an apparently poor sensor. But in doing so, it can significantly benefit from this decision from time step two on.

Information Theoretic vs. Covariance-based Objective

The effect of taking an information theoretic objective function, namely mutual information, is demonstrated in the following by comparing the performance of the three information theoretic sensor managers with three managers that indeed employ a closed-loop model predictive control scheme but now with an covariance-based objective. More specifically, the covariance-based sensor managers (CBSMs) employ the trace-based objective (2.22). The rmse for $N = 2$ and $N = 3$ is illustrated in Figure 4.6 (a)[3]. It is obvious that the ITSM with $N = 3$ leads to the best estimation accuracy, while the ITSM for $N = 2$ is comparable to the CBSMs. More interesting, there is almost no difference between both CBSMs. The CBSM with $N = 3$ does not benefit from the longer lookahead compared to its ITSM counterpart.

In addition, as Figure 4.6 (b) indicates, the ITSM with $N = 3$ minimizes the trace of the state covariance even stronger, although this is the main objective of the CBSMs. One reason for this is that information theoretic objectives prefer different uncertainty regions, which are indicated by multiple modes in the state density function, while covariance-based objective prefer one single uncertainty region corresponding to a Gaussian-like density. In the considered tracking scenario, the state density is typically non-Gaussian (compare with Example 4.1) and the different modes correspond to different regions of potential vehicle positions. By employing an information theoretic objective in combination with long time horizons, tolerating multimodal densities in the first instance followed by explicitly eliminating modes and thus potential

3 Since the performance of CBMS for $N = 1$ is comparable with the performance of ITSM with $N = 1$, both rmse curves are omitted for clarity.

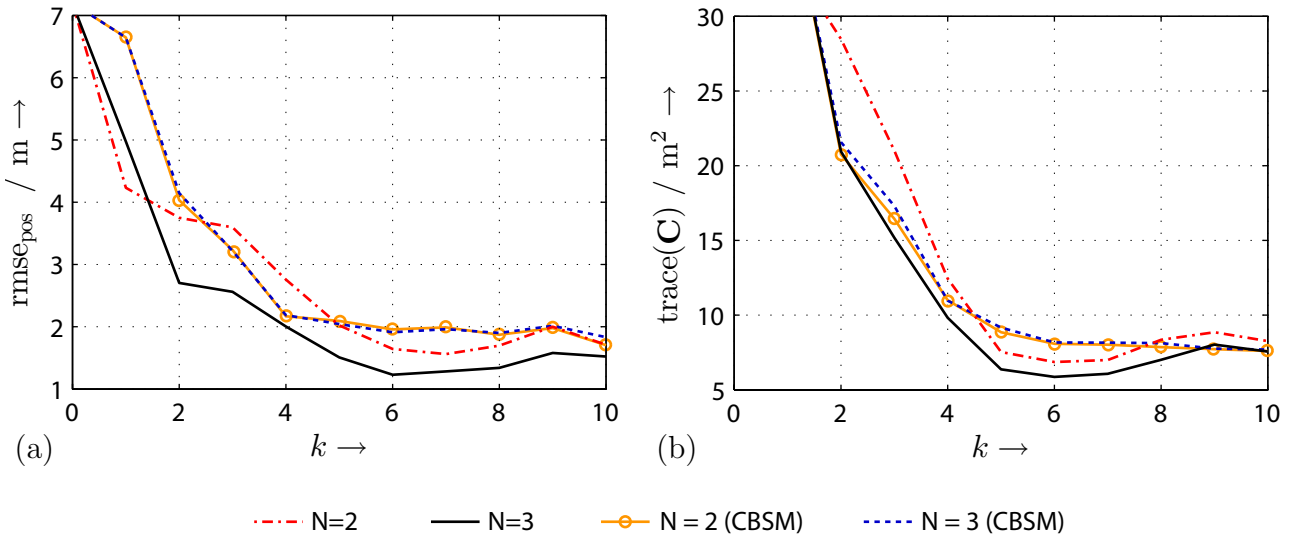


Figure 4.6: Comparison of information theoretic with covariance-based sensor management. (a) Rmse of the position estimates for the time horizons $N = 2$ and $N = 3$. (b) Evolution of the trace of the state covariance, averaged over all simulation runs. Merely the covariance matrix diagonal elements corresponding to the vehicle position are considered for this plot.

vehicle positions is possible. This in turn leads to more certain state estimates as it would be the case for covariance-based objective functions, which are not able to quantify higher-order information of the state density. Consequently, not only the state covariance is minimized but also an integral minimization of the uncertainty incorporated in the state density is achieved. Additionally, mutual information allows comparing different entities of the state vector, i.e., position and orientation, on an equal basis. This is not the case for the employed covariance-based objective function.

4.4 Summary

The information theoretic approach introduced in this chapter provides a feasible closed-loop model predictive control solution to the sensor management problem. The anticipation of future measurement values and thus the incorporation of the influence of uncertainties arising from noise-corrupted measurements on the sensor management decisions is accomplished by means of virtual measurements. In consideration of highly nonlinear system and sensor models, the behavior of the Bayesian estimator is mimicked during planning by providing accurate Gaussian mixture representations of the state estimates within the time horizon. In order to extract meaningful higher-order information from this non-Gaussian state representation, the proposed information theoretic sensor management approach employs mutual information as objective function. As demonstrated in the simulations, configuration sequences determined by means of this type of objective function yields improved estimates of the observed system state compared to the widely-used covariance-based objectives.

To reduce the substantial computational burden, three steps are taken. First, the number of virtual measurements, which has an significant impact on the computational tractability, can be kept on low level by employing Dirac mixture approximation techniques. This allows determining a small number of representative expected measurement values. Second, for an efficient tree search that additionally is capable of dealing with the probabilistic nature of the virtual measurements, the probabilistic branch-and-bound algorithm is employed. And finally, computationally cheap lower and upper bounds to the differential entropy are derived for the

employed Gaussian mixture representation of the system state. These bounds provide the foundation for a tight and meaningful approximation of the mutual information objective.

The proposed information theoretic sensor management approach represents one of the few existing closed-loop model predictive control approaches for continuous states, measurement values, and non-myopic optimization. In [107] for instance, Monte Carlo sampling was employed for both state estimation and measurement value anticipation purposes. Since Monte Carlo sampling results in many non-representative measurements, the amount of computations is large. To face this problem, an application-specific value-to-go approximation is proposed in addition, which significantly reduces the computation time and simultaneously keeps the loss in estimation performance to a minimum. Such an approximation is one option that should be considered for improving the proposed sensor management approach. Here, deriving more generally applicable approximations compared to the one presented in [107, 110] are desirable. Eventually, those approximation could provide a tight lower bound on the value-to-go. These bounds could additionally be employed in the PBAB algorithm for an improved pruning performance.

A further option for future work is to transfer the proposed techniques for calculating representative virtual measurements and tight bounds of information theoretic objectives to open-loop sensor management. Similarly to the approach proposed in [42], this would compromise between open-loop and closed-loop sensor management. In doing so, the virtual measurements would be merely utilized for calculating expected information theoretic objective function values for non-Gaussian state densities, whereas state estimation within the time horizon is still performed in an open-loop control fashion. Hence, the effect of future measurement values is not totally neglected as in a pure open-loop approach, while the computational burden of a pure closed-loop approach is avoided.

CHAPTER 5

Gaussian Estimator based on Deterministic Sampling

The open-loop model predictive control sensor management scheme introduced in Section 3 strongly relies on statistical linearization for converting a nonlinear non-Gaussian management problem into a linear Gaussian one. In principle, every linear regression Kalman filter (LRKF) can be employed within this management scheme. However, typical LRKFs only take the lower-order statistics of the state into account, while the regression point calculation of the Gaussian estimator proposed in this chapter also considers the shape of the distribution function of the prior Gaussian density. In recursive state estimation, this incorporation of shape information automatically leads to a more accurate approximation of higher-order moments, especially in cases of near Gaussian posterior densities. Together with a freely adjustable number of regression points, an improved linearization quality and estimation accuracy are the consequence.

To provide shape approximation under the constraints of exactly capturing mean and covariance, an optimization problem is formulated in Section 5.2 for one-dimensional states, where the regression points are interpreted as an analytic density function, namely a Dirac mixture. This is different from most of the existing LRKFs or even particle filters, where the regression points are not chosen in order to explicitly incorporate shape information and higher-order moments, respectively. The solution of the optimization problem can be calculated off-line and is extended to multivariate densities in Section 5.3. Here, two approaches are proposed, where the *axis-aligned approach* relies on a sparse placement of regression points along the principal axes and thus, the number of required regression points only grows linear with the dimension of the state space. The *grid approach* on the other hand provides a more dense coverage of regression points, which is potentially more accurate but comes at the expense of an exponential increase in complexity. The complete Gaussian estimator and an extension to Gaussian mixtures is presented in Section 5.4 and Section 5.5, respectively.

The Gaussian estimator introduced in this section was published in [217] and its applicability on the real-world localization problem of tracking a human in a telepresence environment was demonstrated in experiments in [209]. Extensions to these publications are the grid approach for the extension to multivariate densities and the extension to Gaussian mixtures.

5.1 Problem Formulation

In order to derive a Gaussian estimator based on the linear regression Kalman filter framework, it is sufficient to restrict the focus on a nonlinear transformation

$$\underline{\boldsymbol{y}} = \underline{g}(\underline{\boldsymbol{x}}) \ , \tag{5.1}$$

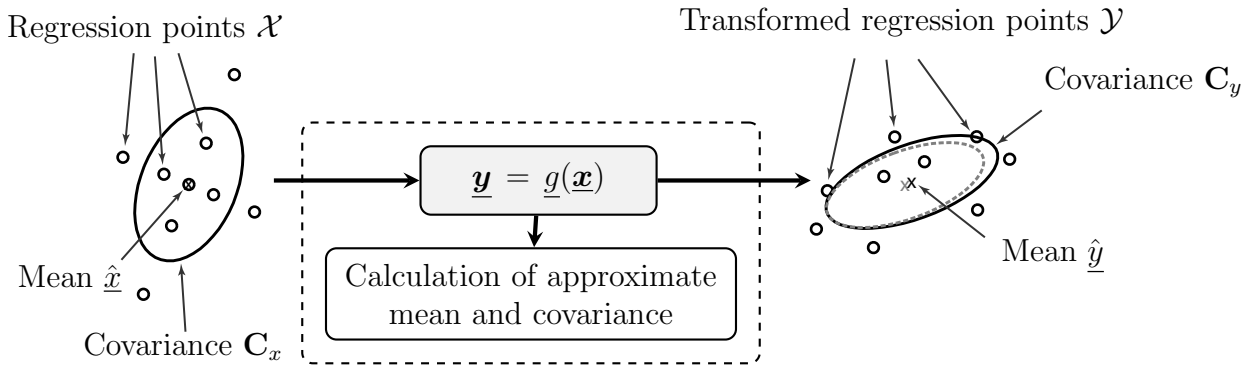


Figure 5.1: Principle of the proposed Gaussian estimator. The set of regression points $\mathcal{X}$ representing the Gaussian density of $\underline{\boldsymbol{x}}$ is propagated through the nonlinear transformation $\underline{\boldsymbol{y}} = \underline{g}(\underline{\boldsymbol{x}})$. By means of the transformed set of regression points $\mathcal{Y}$, mean and covariance matrix for the Gaussian density $\mathcal{N}(\underline{y}; \hat{\underline{y}}, \mathbf{C}_y)$ can be calculated in order to approximate the true mean and covariance of $\underline{\boldsymbol{y}}$ (light gray cross and ellipse, respectively).

where $\underline{\boldsymbol{x}} \in \mathbb{R}^{n_x}$ and $\underline{\boldsymbol{y}} \in \mathbb{R}^{n_y}$ are random vectors and $\underline{g}(\,\cdot\,)$ is an arbitrary nonlinear function. With (5.1), the random vector $\underline{\boldsymbol{x}}$ is nonlinearly mapped to the random vector $\underline{\boldsymbol{y}}$. This transformation is analogous to the prediction of the system state $\underline{\boldsymbol{x}}_k$ and the measurement $\underline{\boldsymbol{z}}_k$ by means of the nonlinear functions $\underline{a}_k(\,\cdot\,)$ and $\underline{h}_k(\,\cdot\,)$ in the nonlinear system model (2.1) and sensor model (2.3), respectively.

5.1.1 Estimation via Regression Points

In general, calculating the density function or the statistics of $\underline{\boldsymbol{y}}$ cannot be carried out in closed form. For arbitrary nonlinear transformations, analytic expressions for the formulas of the Bayesian estimator are not available. Hence, *directly* processing the density or the moments is computationally demanding and imprecise, or even impossible in cases where the nonlinear transformation is not given in an analytic form. By considering a sample representation, processing the density function of $\underline{\boldsymbol{x}}$ can be simplified and efficiently calculating an approximate density function representing $\underline{\boldsymbol{y}}$ is possible.

A specific way to maintain a computationally efficient estimator is to consider a Gaussian density representation $\tilde{f}(\underline{x}) = \mathcal{N}(\underline{x}; \hat{\underline{x}}, \mathbf{C}_x)$ of $\underline{\boldsymbol{x}}$. Gaussian densities are able to represent the most interesting statistical values, namely the mean and the covariance. This is especially the case for tracking applications, where the main interest is on estimating the target position and some kind of uncertainty around the estimated mean. Within the LRKF framework, it is possible to obtain a set of regression points $\mathcal{X}$, i.e., a set of sample points, that exactly captures these first two moments. As illustrated in Figure 5.1, the regression points then can be easily propagated through the nonlinear transformation (5.1) and in turn allow efficiently approximating the true density function of $\underline{\boldsymbol{y}}$ by a Gaussian density $f(\underline{y}) = \mathcal{N}(\underline{y}; \hat{\underline{y}}, \mathbf{C}_y)$. Therefore, only the first two moments, i.e., mean vector $\hat{\underline{y}}$ and covariance matrix $\mathbf{C}_y$, need to be calculated, which can be performed with polynomial complexity with respect to the dimension of $\underline{\boldsymbol{x}}$ and $\underline{\boldsymbol{y}}$.

5.1.2 Contrast to Prior Work

The achievable accuracy of the Gaussian approximation for $\underline{\boldsymbol{y}}$ strongly depends on the strategy of determining the sample representation of $\underline{\boldsymbol{x}}$. Most of existing LRKF approaches like the unscented Kalman filter (UKF, see [91, 92]), the divided differences filter (DDF, [137]), or the central differences filter (CDF, [160]) use a minimal fixed-size set consisting of $2 \cdot n_x + 1$ regression points with corresponding weights that exactly captures the mean $\hat{\underline{x}}$ and the covariance

matrix $\mathbf{C}_x$ of $\underline{\boldsymbol{x}}$. An extension of the UKF employing $6 \cdot n_x + 1$ regression points was proposed in [197]. For any nonlinear transformation, the propagated regression points capture the posterior mean and covariance matrix accurately up to the second order of the corresponding Taylor-series expansion [180].

For further improving the accuracy, the set of regression points used for the Gaussian estimator proposed in this chapter is not restricted to a fixed size. Increasing the number of regression points gives several advantages. Since more regression points are propagated, more information of the nonlinear transformation is captured. This leads to improved and more robust estimates and the Gaussian estimator is well applicable to a larger number of nonlinear transformations. Furthermore, the user is able to adjust the quality as well as the computational demand of the approximation. A LRKF based on a set of regression points of variable size is introduced in [85]. Here, Gauss–Hermite quadrature is employed, which results in exponential complexity with respect to n_x. The complexity of the proposed Gaussian estimator instead remains polynomial with the respect to the dimension of the state space.

To gain the advantages of a larger set of regression points, as much information about $\underline{\boldsymbol{x}}$ as possible has to be incorporated when determining the regression points. While for example the UKF only considers mean and covariance, the regression point selection scheme derived in the following section is based on directly approximating the cumulative distribution function of the prior Gaussian. This is motivated by the fact that an accurate approximation of the distribution function automatically approximates higher-order moments. These moments in turn have an impact on higher-order terms of the Taylor-series expansion of the nonlinear function, which leads to improved estimation results.

For an accurate approximation, the key idea is now to reformulate the approximation problem for determining the regression points as an optimization problem by minimizing a certain distance measure $G(\,\cdot\,)$ between the Gaussian and a convenient analytic representation of the regression points under the constraints that mean $\underline{\hat{x}}$ and covariance $\mathbf{C}_x$ of $\underline{\boldsymbol{x}}$ are captured exactly. This is different from classical LRKFs or even particle filters [9], where no distance measure is employed. The calculation scheme for the regression points presented in the following can be interpreted as *deterministic sampling*, since it deterministically exploits the distribution function of the random vector, while particle filters employ random sampling.

5.2 One-Dimensional Approximation

At first, only one-dimensional transformations $\boldsymbol{y} = g(\boldsymbol{x})$ are considered, where the random variable $\boldsymbol{x}$ is characterized by mean $\hat{x}$ and variance σ_x^2. Without loss of generality, it can be assumed that $\hat{x} = 0$ and $\sigma_x^2 = 1$, which leads to a *standard* Gaussian density $\tilde{f}(x) = \mathcal{N}(x; 0, 1)$ for representing $\boldsymbol{x}$. This restriction is justified, since every Gaussian can be affinely transformed into a standard Gaussian density.

To achieve the goal of calculating a set of regression points $\mathcal{X} = \{\omega_i, \hat{x}_i | i = 1, \ldots, L\}$ that accurately represents both the density function and the first two moments of $\boldsymbol{x}$, constrained optimization

$$\begin{aligned} &\min_{\mathcal{X}} G\left(\tilde{f}(x), \mathcal{X}\right) \\ &\text{w.r.t.} \qquad \hat{x} = 0 \text{ and } \sigma_x^2 = 1 \end{aligned} \tag{5.2}$$

is required. Here, $G(\,\cdot\,)$ is an appropriate distance measure quantifying the deviation between the density of $\boldsymbol{x}$ and the set of regression points $\mathcal{X}$.

5.2.1 Dirac Mixture

To provide the desired regression point representation of $\boldsymbol{x}$ that incorporates information of the density function, an analytic and parametric form for representing the regression points in terms of a so-called *Dirac mixture* density function

$$f(x;\underline{\eta}) = \sum_{i=1}^{L} \omega_i \cdot \delta(x - \hat{x}_i) \ , \tag{5.3}$$

is employed, which is a weighted sum of L Dirac delta distributions $\delta(x - \hat{x}_i)$ representing the weighted regression point at position $\hat{x}_i$. Hence, the parameter vector $\underline{\eta}$ comprises the regression points $\hat{x}_i$ with corresponding weighting coefficients ω_i. A summary of the properties of Dirac delta distributions and Dirac mixtures can be found in Section A.1.3 and Section A.2.2, respectively.

To reduce the number of parameters of $f(x;\underline{\eta})$ to be adjusted for approximation, equal weighting coefficients ω_i are assumed, i.e., $\omega_i = 1/L$. Furthermore, by assuming that the positions $\hat{x}_i$ of the Dirac delta distributions are sorted in ascending order, i.e.,

$$\hat{x}_1 < \hat{x}_2 < \ldots < \hat{x}_L \ ,$$

capturing the mean $\hat{x}$ can easily be guaranteed by placing the Dirac delta distributions symmetrically around $\hat{x}$, i.e.,

$$\hat{x}_{L+1-i} = 2\hat{x} - \hat{x}_i = -\hat{x}_i$$

for $i = 1, 2, \ldots, \bar{L}$ with $\bar{L} := \lceil \frac{L-1}{2} \rceil$. This further reduces the length of $\underline{\eta}$. If L is odd, the center Dirac delta distribution is fixed at the mean $\hat{x}$ by setting $\hat{x}_{\bar{L}+1} = \hat{x} = 0$. Finally, the parameter vector $\underline{\eta}$ is given by $\underline{\eta} = [\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_{\bar{L}}]^{\mathrm{T}}$.

5.2.2 Distance Measure

Typical measures quantifying the distance between densities, like the Kullback-Leibler divergence (2.30) or the squared integral measure [86], cannot be applied directly due to the used Dirac delta distributions in (5.3). Thus, the corresponding cumulative distribution functions are employed instead. The distribution function of the true density $\tilde{f}(x)$ can be written as

$$\tilde{F}(x) = \int_{-\infty}^{x} \tilde{f}(t)\,\mathrm{d}t = \frac{1}{2}\left(1 + \mathrm{erf}\left(\frac{x - \hat{x}}{\sqrt{2}\sigma_x}\right)\right) \ ,$$

where $\mathrm{erf}(\,\cdot\,)$ is the Gaussian error function, while the distribution function corresponding to the Dirac mixture $f(x;\underline{\eta})$ is given by

$$F(x;\underline{\eta}) = \sum_{i=1}^{L} \omega_i \cdot H(x - \hat{x}_i) \ ,$$

where $H(\,\cdot\,)$ is the Heaviside step function

$$H(x - \hat{x}) = \begin{cases} 1 \ , & x > \hat{x} \\ \frac{1}{2} \ , & x = \hat{x} \\ 0 \ , & \text{otherwise} \end{cases} \tag{5.4}$$

at position $\hat{x}$.

By employing the Lagrange multiplier approach [125], the constrained optimization problem (5.2) can be converted into an unconstrained one. For this purpose the distance measure

$$G(\underline{\eta}, \lambda) = \frac{1}{2} \int_{\mathbb{R}} \left(\tilde{F}(x) - F(x; \underline{\eta}) \right)^2 \mathrm{d}x + \lambda \cdot \left(\frac{1}{L} \sum_{i=1}^{L} \hat{x}_i^2 - \sigma_x^2 \right) \tag{5.5}$$

is considered in the following. The first term in (5.5) is the so-called *Cramér-von Mises distance* [24, 162] quantifying the deviation between the distribution functions and λ in the second term is a *Lagrange multiplier*. By means of the Lagrange multiplier approach, exactly capturing the variance σ_x^2 is guaranteed if $L \geq 2$, while exactly capturing the mean is automatically guaranteed thanks to the symmetric positioning of the regression points around the mean $\hat{x} = 0$.

5.2.3 Solution

To minimize the distance measure with respect to $\underline{\eta}$ and λ, the necessary conditions for a minimum $\partial G(\underline{\eta}, \lambda)/\partial \underline{\eta} = \underline{0}$ and $\partial G(\underline{\eta}, \lambda)/\partial \lambda = 0$ have to be satisfied. Utilizing the sifting property of the Dirac delta distribution, the partial derivative of $G(\,\cdot\,)$ with respect to the regression point position $\hat{x}_i$ yields

$$\frac{\partial G(\underline{\eta}, \lambda)}{\partial \hat{x}_i} = \frac{1}{L} \cdot \left(\tilde{F}(\hat{x}_i) - F(\hat{x}_i; \underline{\eta}) + 4\lambda \hat{x}_i - \left(\tilde{F}(-\hat{x}_i) - F(-\hat{x}_i; \underline{\eta}) \right) \right) , \tag{5.6}$$

for $i = 1, \ldots, \bar{L}$. With the identities

$$\begin{aligned} F(\hat{x}_i; \underline{\eta}) &= \tfrac{2i-1}{2L} \ , \\ \tilde{F}(-\hat{x}_i) &= 1 - \tilde{F}(\hat{x}_i) \ , \\ F(-\hat{x}_i; \underline{\eta}) &= 1 - F(\hat{x}_i; \underline{\eta}) \ , \end{aligned}$$

and by setting (5.6) equal zero yields

$$\tilde{F}(\hat{x}_i) - \frac{2i-1}{2L} + 2\lambda \hat{x}_i = 0 \ . \tag{5.7}$$

The partial derivative of $G(\,\cdot\,)$ with respect to λ results in

$$\sum_{i=1}^{\bar{L}} \hat{x}_i^2 - \tfrac{L}{2} \sigma_x^2 = 0 \ . \tag{5.8}$$

The resulting system of nonlinear equations comprising (5.7) and (5.8) is square, i.e., the number of equations equals the number of unknowns. The optimal solution can be determined in closed form for $L \in \{2, 3\}$. Here, (5.8) has the unique solutions

$$\hat{x}_1 = \begin{cases} 1 \ , & \text{if } L = 2 \\ \sqrt{\frac{3}{2}} \ , & \text{if } L = 3 \end{cases} . \tag{5.9}$$

This result coincides with the result of the *unscaled* unscented transformation employed for the UKF. For $L > 3$, iterative root finding algorithms can be applied for determining a root, where the trust-region dogleg method [146] is used throughout this thesis. The required initial solution is defined as

$$\begin{aligned} \hat{x}_i &= \sqrt{2} \cdot \operatorname{erf}^{-1}\left(\tfrac{2i-1-L}{L} \right) \ , \ i = 1, \ldots, \bar{L} \ , \\ \lambda &= 0 \ , \end{aligned}$$

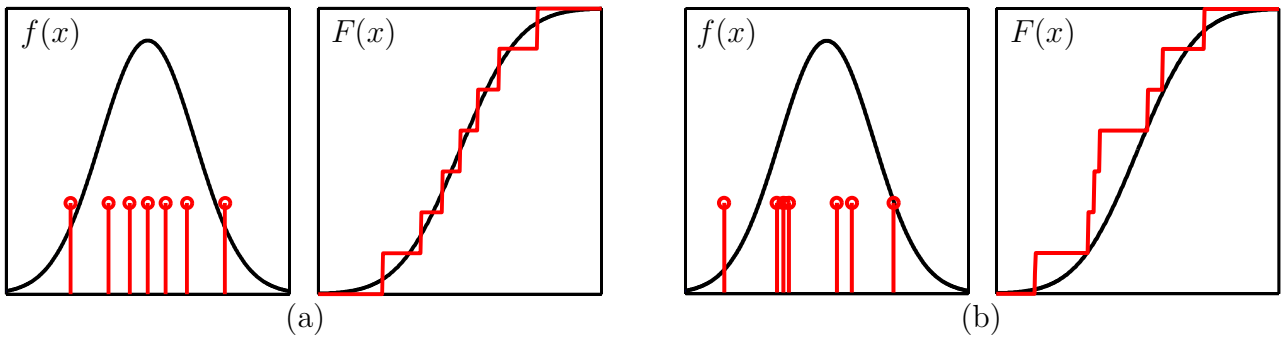


Figure 5.2: (a) Approximation of the standard Gaussian $\tilde{f}(x) = \mathcal{N}(x; 0, 1)$ (black lines) by means of $L = 7$ regression points. (b) Approximation resulting from random sampling with $L = 7$ samples. For both methods, the left figure depicts the approximation in density space, while the right figure depicts the approximation in distribution space.

which is the optimal solution of minimizing $G(\,\cdot\,)$ without considering the constraint on the variance [162].

In Table 5.1, the parameters for sets of regression points with different size are listed. The corresponding approximation of $\tilde{f}(x)$ in the density space as well as in the distribution space is depicted in Figure 5.2 (a) for $L = 7$. As noted in the beginning, the calculation of the regression points by minimizing (5.5) can be considered as deterministic sampling. By the proposed calculation scheme (or deterministic sampling), the regression points (or samples) are placed in a sensible fashion in order to provide an accurate representation of the moments and distribution function. Employing random sampling instead leads to very irregular placements as illustrated in Figure 5.2 (b). Consequently, a much larger set of points is necessary for obtaining comparable approximates of the moments and distribution function of $\boldsymbol{x}$.

Table 5.1: Parameters for several numbers of regression points.

L	x_1	x_2	x_3	x_4	x_5	x_6	x_7
3	-1.2247	0	1.2247	–	–	–	–
5	-1.4795	-0.5578	0	0.5578	1.4795	–	–
7	-1.6346	-0.8275	-0.3788	0	0.3788	0.8275	1.6346

5.2.4 Off-line Approximation

The resulting regression points $\hat{x}_i$ that minimize (5.5) are valid for a standard Gaussian density. For arbitrary Gaussians it is beneficial to split the approximation task into an off-line and an on-line part, instead of solving a similar optimization problem on-line. In doing so, the calculation scheme for the regression points derived before is performed off-line for a desired number of regression points (see Table 5.1 for several approximations). Then, for on-line estimation, these regression points have to be scaled and shifted according to

$$\hat{x} + \sigma_x \cdot \hat{x}_i \ ,$$

where $\hat{x}$ and σ_x are now arbitrary means and standard deviations, respectively. This transformation leads to an on-line approximation of any Gaussian density without impairing the approximation quality. Furthermore, the on-line performance for state estimation is drastically increased.

Table 5.2: Approximate calculation of the moments of the random variable $\boldsymbol{y} = |\boldsymbol{x}|$.

	Moments				
	1	2	3	4	5
MC	0.8955	1.2500	2.2064	4.5616	10.6191
UKF	0.9832	1.2500	1.8788	3.0625	5.1646
GE with $L = 3$	0.9832	1.2500	1.8788	3.0625	5.1646
GE with $L = 7$	0.9177	1.2500	2.0523	3.7419	7.2679
GE with $L = 15$	0.9032	1.2500	2.1266	4.0948	8.5405

Example 5.1: Scalar Transformation
In this example, the improved estimation performance by employing a large set of regression points is demonstrated by means of the scalar transformation

$$\boldsymbol{y} = |\boldsymbol{x}| \ ,$$

where $\boldsymbol{x}$ is Gaussian with mean $\hat{x} = 0.5$ and variance $\sigma_x^2 = 1$, i.e., $\tilde{f}(x) = \mathcal{N}(x; 0.5, 1)$. The resulting random variable $\boldsymbol{y}$ is non-Gaussian and the odd moments are non-zero [176]. A Monte Carlo (MC) simulation with 10 million samples is performed to determine the true (non-central) moments of $\boldsymbol{y}$. The results are shown in Table 5.2 together with the estimates of the UKF[1] and the proposed Gaussian estimator (denoted as GE). For the GE, $L \in \{3, 7, 15\}$ regression points are used. It is obvious that the results of UKF and GE coincide for $L = 3$ (recall (5.9)), while the moment estimates of the GE converge to the MC results when increasing the number of regression points. Especially the moments of order three and higher can be approached, which is not possible for the UKF. The only parameter to be adjusted for an increased estimation quality is the number of regression points L, while the UKF provides three parameters, whose tuning has to be carried out very carefully. ■

5.2.5 Consideration of Higher-order Moments

Thanks to consideration of the distribution function during the calculation of the regression points, information about higher-order moments is implicitly incorporated. The quality of approximating higher-order moments can be further increased, if information about higher-order moments is *explicitly* considered. For this purpose, additional Lagrange multipliers have to be introduced in (5.5). Hence, when capturing the first R moments of $\tilde{f}(x)$, the modified distance measure is given by

$$G(\underline{\eta}, \lambda) = \frac{1}{2} \int_{\mathbb{R}} \Big(\tilde{F}(x) - F(x, \underline{\eta}) \Big)^2 \mathrm{d}x + \sum_{m=2}^{R} \left(\lambda_m \cdot \Big(\frac{1}{L} \sum_{i=1}^{L} \hat{x}_i^m - \mathrm{E}_x\{\boldsymbol{x}^m\} \Big) \right) \ . \tag{5.10}$$

Obviously, by increasing the number of considered moments R, the minimum number of regression points that is necessary for accurately capturing the moments also increases.

In cases where the higher-order moments correspond to the moments of the *Gaussian* $\tilde{f}(x)$, the approximation can still be performed off-line, since the calculation of all higher-order moments of a Gaussian merely depend on the mean and variance. But in some situation it may occur that information about higher-order moments of the true *non-Gaussian* density function is provided on-line during the estimation process. Here, minimizing (5.10) cannot be carried out off-line any longer. However, the benefit of explicitly capturing higher-order moments is demonstrated in the following example.

1 The parameters of the UKF are set to $\alpha = 1$, $\beta = 0$ and $\kappa = 0.5$.

Table 5.3: Approximate moments of $\boldsymbol{y}$ by explicitly considering the forth moment of $\boldsymbol{x}$.

	Moments				
	1	2	3	4	5
MC	0.8955	1.2500	2.2064	4.5616	10.6191
GE with $L = 3$	0.9832	1.2500	1.8788	3.0625	5.1646
GE* with $L = 7$	0.8708	1.2500	2.2498	4.5625	9.8338
GE* with $L = 15$	0.8845	1.2500	2.2270	4.5625	10.2094
GE* with $L = 31$	0.8916	1.2500	2.2161	4.5625	10.3970

Example 5.2: Scalar Transformation (continued)
Consider again the scalar transformation employed in Example 5.1. Now, an additional Lagrange multiplier is used for capturing the forth moment of $\boldsymbol{x}$, which is 3 for the considered Gaussian. Three different sets of regression points are used for approximation, where the set sizes are $L \in \{7, 15, 31\}$. The resulting moment estimates of $\boldsymbol{y}$ are listed in Table 5.3, where the last three rows (indicated by GE*) correspond to the estimates capturing the forth moment of $\boldsymbol{x}$. It can be clearly seen, that not even the forth moment benefits from the explicitly consideration. Also the third and fifth moments of $\boldsymbol{y}$ are much better approximated compared to the result without the additional multiplier (second row). ■

Similar extensions of the regression point calculation procedure of other LRKFs are quite rare. In [89], an extension of the unscented transform, which is employed by the UKF, is presented. This extension allows suboptimally capturing the third-order moment, i.e., the skew. The incorporation of an arbitrary number of moments into the unscented transform is proposed in [176].

5.3 Extensions to Multivariate Case

In the following, vector-valued nonlinear transformations as in (5.1) are considered, i.e., the multivariate random vector $\underline{\boldsymbol{x}} \in \mathbb{R}^{n_x}$ with mean vector $\underline{\hat{x}}$ and covariance matrix $\mathbf{C}_x$ is mapped to the random vector $\underline{\boldsymbol{y}}$.

One way to do so, is to directly extend the optimization problem to multivariate Gaussians. Although this extension would work, it suffers from a computational load increasing with the dimension of $\underline{\boldsymbol{x}}$, where multidimensional integrals have to be evaluated numerically. Instead, the goal is to approximate an arbitrary Gaussian density $\tilde{f}(\underline{x}) = \mathcal{N}(\underline{x}; \underline{\hat{x}}, \mathbf{C}_x)$ by a set of regression points $\mathcal{X}$ under utilization of the previously introduced one-dimensional regression point calculation. This can be achieve by means of a two step procedure: (1) transformation of $\tilde{f}(\underline{x})$ to the multivariate standard Gaussian and then (2) reduction of the multivariate standard Gaussian to the univariate case. Although this procedure is suboptimal compared to directly approximating the arbitrary multivariate Gaussian $\tilde{f}(\underline{x})$, it is much more efficient since the formerly n_x-dimensional optimization problem is decomposed into n_x one-dimensional optimization problems, where the most demanding operations can be performed off-line.

5.3.1 Step 1: Transformation to Multivariate Standard Gaussian

The first step of the approximation procedure is based on the fact that any multivariate Gaussian density $\mathcal{N}(\underline{x}; \underline{\hat{x}}, \mathbf{C}_x)$ can be transformed into a multivariate standard Gaussian $\mathcal{N}(\underline{z}; \underline{0}, \mathbf{I})$ with zero mean and covariance matrix $\mathbf{I}$. Exploiting the eigenvalue decomposition

$$\mathbf{C}_x = \mathbf{V}\mathbf{D}\mathbf{V}^{\mathrm{T}} ,$$

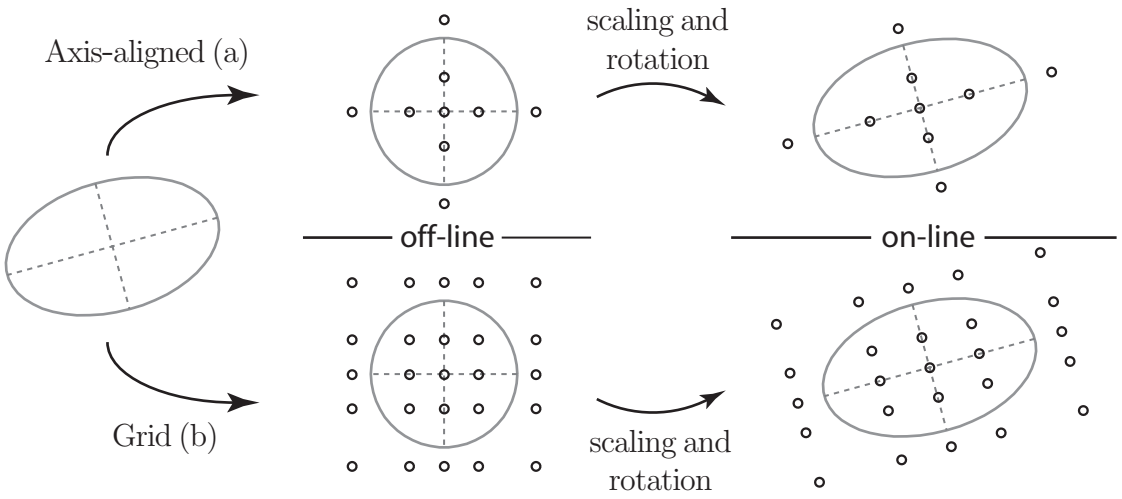


Figure 5.3: Approximating a multivariate Gaussian density by means of (a) the axis-aligned and (b) the grid approach. Determining the set of regression points for a multivariate standard Gaussian can be performed off-line, while adapting (scaling and rotation by means of the eigenvalue decomposition) this set to an arbitrary Gaussian remains an on-line operation.

where $\mathbf{V}$ is the orthogonal matrix of eigenvectors and $\mathbf{D}$ is a diagonal matrix of eigenvalues of $\mathbf{C}_x$, the linear transformation

$$\underline{\boldsymbol{z}} = \left(\mathbf{V}\sqrt{\mathbf{D}}\right)^{-1} \cdot (\underline{\boldsymbol{x}} - \underline{\hat{x}}) \ . \tag{5.11}$$

yields this transformation to the standard Gaussian case. This transformation leads to an identity matrix $\mathbf{C}_z = \mathbf{I}$ and thus, eliminates all correlations of the original Gaussian. Now it is sufficient to determine a set of regression points $\mathcal{Z}$ first for approximating $\mathcal{N}(\underline{z}; \underline{0}, \mathbf{I})$. By inverting the transformation (5.11), this set then can be affinely transformed into the desired set $\mathcal{X}$ for representing the Gaussian $\mathcal{N}(\underline{x}; \underline{\hat{x}}, \mathbf{C}_x)$.

5.3.2 Step 2: Reduction to Univariate Standard Gaussian

For determining the set $\mathcal{Z}$, two possibilities of applying the one-dimensional placement of the regression points as described in Section 5.2 arise:

Axis-aligned Approach Each univariate marginal density of the multivariate standard Gaussian is represented by a set of L regression points. Approximating the multivariate Gaussian is achieved by arranging the marginal sets along the principal axes of the Gaussian density.

Grid Approach As for the axis-aligned approach, the marginal Gaussians are approximated first. But now, the marginal sets are multiplied according to a cartesian product, which corresponds to an irregular grid representation of the multivariate standard Gaussian density.

For both approaches, calculation and arrangement of the marginal sets can be performed off-line, while inverting the transformation (5.11), i.e., scaling and rotation of the set $\mathcal{X}$ are on-line operations. If not explicitly stated elsewhere, the axis-aligned approach is employed throughout this thesis thanks to its computational efficiency. Both approaches are illustrated in Figure 5.3 and are discussed in detail in the following, together with their pros and cons.

Axis-aligned Approach

For a multivariate standard Gaussian $\tilde{f}(\underline{z}) = \mathcal{N}(\underline{z}; \underline{0}, \mathbf{I})$, the univariate marginal density in dimension n is given by

$$\mathcal{N}(z_n; 0, 1) = \int_{\mathbb{R}} \cdots \int_{\mathbb{R}} \mathcal{N}(\underline{z}; \underline{0}, \mathbf{I}) \, \mathrm{d}z_1 \cdots \mathrm{d}z_{n-1} \, \mathrm{d}z_{n+1} \cdots \mathrm{d}z_{n_z} \ ,$$

where $\underline{z} = [z_1, \ldots, z_{n_z}]^{\mathrm{T}}$. To obtain an approximation of the multivariate standard Gaussian, each univariate marginal density can be independently represented by a set of L regression points $\mathcal{Z}_n$. Afterwards, the regression points representing the approximate marginal Gaussians are placed along the principal axes of $\mathcal{N}(\underline{z}; \underline{0}, \mathbf{I})$ as illustrated in Figure 5.3 (a). By employing the inverse transformation of (5.11), the desired approximation of an arbitrary Gaussian density $\tilde{f}(\underline{x}) = \mathcal{N}(\underline{x}; \hat{\underline{x}}, \mathbf{C}_x)$ is given by the set of multivariate regression points $\mathcal{X} = \{\omega_i, \hat{\underline{x}}_i | i = 1, \ldots, n_x \cdot L\}$, where

$$\begin{aligned} \hat{\underline{x}}_i &= \hat{\underline{x}} + \mathbf{V}\sqrt{\mathbf{D}} \cdot \hat{z}_{j_n} \cdot \underline{e}_n \ , \\ \omega_i &= \tfrac{1}{n_x \cdot L} \ . \end{aligned} \tag{5.12}$$

The index j_n of $\hat{z}_{j_n}$ indicates the j-th univariate regression point of the set $\mathcal{Z}_n$ for dimension n and the index $i = (L \cdot (n-1) + j_n) \in \{1, 2, \ldots n_x \cdot L\}$. $\underline{e}_n = [0, \ldots, 0, 1, 0, 0, \ldots, 0]^{\mathrm{T}}$ is the canonical unit vector, where only element n is equal to one. In case of L being odd, the mean vector $\hat{\underline{x}}$ appears n_x times in (5.12). To avoid overestimating the mean, the weights of the concerned regression points have to be adapted according to

$$\omega_i = \tfrac{1}{1 + n_x \cdot (L-1)} \cdot \begin{cases} \frac{1}{n_x} \ , & (i - \bar{L} - 1) \mod L \equiv 0 \\ 1 \ , & \text{otherwise} \end{cases} \ .$$

From the resulting set of weighted regression points $\mathcal{X}$, the mean $\hat{\underline{x}}$ and covariance $\mathbf{C}_x$ of $\tilde{f}(\underline{x})$ can be reconstructed by means of

$$\begin{aligned} \hat{\underline{x}} &= \sum_i \omega_i \cdot \hat{\underline{x}}_i \ , \\ \mathbf{C}_x &= \frac{1}{L} \sum_i (\hat{\underline{x}}_i - \hat{\underline{x}}) \cdot (\hat{\underline{x}}_i - \hat{\underline{x}})^{\mathrm{T}} \ . \end{aligned} \tag{5.13}$$

It is important to note that (5.13) does not coincide with the weighted sample covariance that is typically employed in the context of statistical linearization. Employing the weighted sample covariance would lead to an underestimation of the covariance. Instead, the factor $1/L$ is used for the weights ω_i, which provides an unbiased estimate of $\mathbf{C}_x$ for the employed regression point representation. This characteristic of the axis-aligned approach follows from the fact that the dimension of the state space has no impact on the calculation of the positions $\hat{\underline{x}}_i$ of the regression points in (5.12). Thus, no scaling of the regression points for high-dimensional estimation problems is required. This is contrary to the unscented transform, where the radius of the sphere that bounds the set of regression points increases with the dimension of the state space. Hence, a scaled version of the unscented transform is necessary to avoid capturing non-local effects [90].

Due to the placement of the marginal sets of regression points along the principal axes, the number of required regression points for approximating $\tilde{f}(\underline{x})$ grows only linearly with the dimension of the state space n_x. This facilitates computationally efficient state estimation even for high-dimensional estimation problems. However, for some estimation problems, the sparse placement resulting from the axis-aligned approach leads to an inferior estimation performance compared to the grid approach introduced next.

Grid Approach

The grid approach exploits the separation of the multivariate standard Gaussian into n_z univariate standard Gaussians according to

$$\mathcal{N}(\underline{z}; \underline{0}, \mathbf{I}) = \prod_{n=1}^{n_z} \mathcal{N}(z_n; 0, 1) \ , \ \text{where } \underline{z} = [z_1, \ldots, z_{n_z}]^{\mathrm{T}} \ . \tag{5.14}$$

By replacing each univariate standard Gaussian $\mathcal{N}(z_n; 0, 1)$ in (5.14) with the Dirac mixture representation $\sum_{j=1}^{L_n} \frac{1}{L_n} \cdot \delta(z_n - \hat{z}_{j_n})$ calculated according to Section 5.2, the multivariate standard Gaussian $\mathcal{N}(\underline{z}; \underline{0}, \mathbf{I})$ is approximated by

$$\begin{aligned}
\mathcal{N}(\underline{z}; \underline{0}, \mathbf{I}) &\approx \prod_{n=1}^{n_z} \sum_{j=1}^{L_n} \frac{1}{L_n} \cdot \delta(z_n - \hat{z}_{j_n}) \\
&= \underbrace{\sum_{j_1=1}^{L_1} \cdots \sum_{j_{n_z}=1}^{L_{n_z}}}_{n_z \text{ times}} \underbrace{\prod_{n=1}^{n_z} \frac{1}{L_n} \cdot \delta(z_n - \hat{z}_{j_n})}_{=: \ \omega_i \cdot \delta(\underline{z} - \underline{\hat{z}}_i)} \\
&= \sum_{i=1}^{L} \omega_i \cdot \delta(\underline{z} - \underline{\hat{z}}_i) \ ,
\end{aligned} \tag{5.15}$$

with index $i = (1 + \sum_{n=1}^{n_z} (j_n - 1) \cdot L_n^{n-1}) \in \{1, 2, \ldots L\}$, the number L_n of Dirac components for dimension n, the total number of Dirac components $L = \prod_{n=1}^{n_z} L_n$, and the weights $\omega_i = 1/L$. This procedure corresponds to arranging the regression points in form of an irregular grid as depicted in Figure 5.3 (b).

To approximate an arbitrary multivariate Gaussian, the inverse transformation of (5.11) needs to be applied on (5.15), which leads to a scaling and rotation of the regression points according to

$$\underline{\hat{x}}_i = \underline{\hat{x}} + \mathbf{V}\sqrt{\mathbf{D}} \cdot \underline{\hat{z}}_i$$

and thus, the multivariate Gaussian is approximated according to

$$\mathcal{N}(\underline{x}; \underline{\hat{x}}, \mathbf{C}_x) \approx \sum_{i=1}^{L^{n_z}} \omega_i \cdot \delta(\underline{x} - \underline{\hat{x}}_i) \ .$$

Analogously to the axis-aligned approach, merely scaling and rotation have to be performed online. In contrast to the axis-aligned approach, the covariance matrix of the Gaussian $\mathcal{N}(\underline{x}; \underline{\hat{x}}, \mathbf{C}_x)$ is now calculated by means of the sample covariance, i.e.,

$$\mathbf{C}_x = \sum_i \omega_i \cdot \ (\underline{\hat{x}}_i - \underline{\hat{x}}) \cdot \ (\underline{\hat{x}}_i - \underline{\hat{x}})^{\mathrm{T}} \ . \tag{5.16}$$

Due to the placement of the regression points in form of a grid, the number of necessary grid points grows exponentially with the dimension of the state space, which apparently restricts this approach to low dimensional estimation problems. Fortunately, the grid approach facilitates the use of sets with different size for different dimensions, which attenuates the scaling problem. Less relevant dimensions of the state can be represented by fewer regression points, e.g., in case of an augmented state vector (see next section), where the noise dimensions can be considered as less relevant. Furthermore, the techniques proposed in Section 6.3.2 for decomposing the state vector can also be employed here (see [208]). By this means, only parts of the state vector needs to be represented by a set of regression points, which facilitates the application of the grid approach even for many high-dimensional problems.

5.4 Gaussian Estimator

For applying the proposed deterministic sampling approach to recursive nonlinear estimation for systems and sensors characterized according to (2.1) and (2.3), respectively, the system state has to be augmented with the noise variables. The resulting augmented system state is denoted by $\underline{\boldsymbol{X}}_k = [\underline{\boldsymbol{x}}_k^{\mathrm{T}}, \underline{\boldsymbol{w}}_k^{\mathrm{T}}, \underline{\boldsymbol{v}}_k^{\mathrm{T}}]^{\mathrm{T}}$. The regression points $\mathcal{X}_k = \{\omega_i, \hat{\underline{X}}_{k,i} | i = 1, \ldots, n_X \cdot L, n_X = n_x + n_w + n_v\}$ are now determined for the augmented system state, where $\hat{\underline{X}}_i = \left[(\hat{\underline{x}}_{k,i})^{\mathrm{T}}, (\hat{\underline{w}}_{k,i})^{\mathrm{T}}, (\hat{\underline{v}}_{k,i})^{\mathrm{T}}\right]^{\mathrm{T}}$.

5.4.1 Estimator Equations

In the following, the equations of the Gaussian estimator for nonlinear state estimation based on the novel approximation scheme are listed in analogy to the equations of the UKF presented in [183]. Initially, the augmented system state at time step $k = 0$ is given by $\underline{\boldsymbol{X}}_0$ with mean and covariance matrix

$$\hat{\underline{X}}_0 = [(\hat{\underline{x}}_0^e)^{\mathrm{T}}, \hat{\underline{w}}_0^{\mathrm{T}}, \hat{\underline{v}}_0^{\mathrm{T}}]^{\mathrm{T}} \,, \qquad \mathbf{C}_0^X = \begin{bmatrix} \mathbf{C}_0^e & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_0^w & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{C}_0^v \end{bmatrix} \,.$$

For the time steps $k = 1, 2, \ldots$ do:

1. Determine the set of regression points $\mathcal{X}_{k-1}$ for $\underline{\boldsymbol{X}}_{k-1}$.
2. Prediction step:

$$\begin{aligned}
&\text{(Predicted regression points)} && \hat{\underline{x}}_{k,i} = \underline{a}_{k-1}\left(\hat{\underline{x}}_{k-1,i}, \hat{\underline{w}}_{k-1}\right) \\
&\text{(Predicted mean)} && \hat{\underline{x}}_k^p = \sum_i \omega_i \cdot \hat{\underline{x}}_{k-1,i} && (5.17) \\
&\text{(Predicted covariance)} && \mathbf{C}_k^p = \frac{1}{L} \sum_i \left(\hat{\underline{x}}_{k,i} - \hat{\underline{x}}_k^p\right) \cdot \left(\hat{\underline{x}}_{k,i} - \hat{\underline{x}}_k^p\right)^{\mathrm{T}} && (5.18)
\end{aligned}$$

3. Measurement update step:

$$\begin{aligned}
&\text{(Measurement regression points)} && \hat{\underline{z}}_{k,i} = \underline{h}_k\left(\hat{\underline{x}}_{k,i}, \underline{u}_k, \hat{\underline{v}}_{k,i}\right) \\
&\text{(Predicted measurement value)} && \hat{\underline{z}}_k^p = \sum_i \omega_i \cdot \hat{\underline{z}}_{k,i} && (5.19) \\
&\text{(Measurement covariance)} && \mathbf{C}_k^z = \frac{1}{L} \sum_i \left(\hat{\underline{z}}_{k,i} - \hat{\underline{z}}_k^p\right) \cdot \left(\hat{\underline{z}}_{k,i} - \hat{\underline{z}}_k^p\right)^{\mathrm{T}} && (5.20) \\
&\text{(Cross covariance)} && \mathbf{C}_k^{xz} = \frac{1}{L} \sum_i \left(\hat{\underline{x}}_{k,i} - \hat{\underline{x}}_k^p\right) \cdot \left(\hat{\underline{z}}_{k,i} - \hat{\underline{z}}_k^p\right)^{\mathrm{T}} \\
&\text{(Estimated Kalman gain)} && \mathbf{K}_k = \mathbf{C}_k^{xz} \left(\mathbf{C}_k^z\right)^{-1} \\
&\text{(Posterior mean)} && \hat{\underline{x}}_k^e = \hat{\underline{x}}_k^p + \mathbf{K}_k \cdot \left(\hat{\underline{z}}_k - \hat{\underline{z}}_k^p\right) \\
&\text{(Posterior covariance)} && \mathbf{C}_k^e = \mathbf{C}_k^p - \mathbf{K}_k \mathbf{C}_k^z \mathbf{K}_k^{\mathrm{T}}
\end{aligned}$$

The estimator equations are designed for the axis-aligned approach. The adaptation to the grid approach can be easily achieved by replacing the covariance matrix calculations above with the sample covariance (5.16).

Since merely forward mapping of regression points is required, it is not necessary to have an analytic description of the nonlinear system function $\underline{a}_k(\cdot)$ and the nonlinear measurement

function $\underline{h}_k(\cdot)$. It is additionally assumed for the measurement update step that the predicted system state $\underline{\boldsymbol{x}}_k^p$ and the measurement $\underline{\boldsymbol{z}}_k$ are jointly Gaussian. This assumption is only true for linear relationships between $\underline{\boldsymbol{x}}_k^p$ and $\underline{\boldsymbol{z}}_k$, otherwise it is an approximation. This approximation also appears in other LRKFs and thus, the Gaussian estimator employs the same sequence of operations as typical LRKFs for determining the predicted and posterior state estimate.

For a small number of regression points L, the computational complexity for calculating an estimate is comparable to those LRKFs with a fixed-size set of regression points, since the main effort is spent for solving the optimization problem, which can be carried out off-line. Performing the eigenvalue decomposition of $\mathbf{C}_k^X$ for on-line scaling and rotation has the same complexity as for example the matrix square root required for the UKF.

5.4.2 Simulation Examples

The estimation performance of the novel Gaussian estimator is compared to the popular UKF by means of two simulations from the field of target localization and tracking. In the first simulation example, the target is tracked by a single stationary sensor. The impact of both estimators when employed in the quasi-linear sensor manager as estimator for planning, i.e., when employed for statistical linearization, is demonstrated in the second simulation for the mobile sensor control problem.

Stationary Sensor

In the considered simulation scenario, the target with bicycle kinematics is localized using dead-reckoning and measurements from a single distance sensor located at the origin. Dead-reckoning employs the kinematic model

$$\underline{\boldsymbol{x}}_{k+1} := \begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{y}_{k+1} \\ \boldsymbol{\phi}_{k+1} \end{bmatrix} = \underline{\boldsymbol{x}}_k + (v_k + \boldsymbol{w}_k^v) \cdot \begin{bmatrix} \cos(\boldsymbol{\phi}_k) \\ \sin(\boldsymbol{\phi}_k) \\ \tan(\alpha_k + \boldsymbol{w}_k^\alpha) \end{bmatrix} , \tag{5.21}$$

where the system state $\underline{\boldsymbol{x}}_k$ comprises the position $[\boldsymbol{x}_k, \boldsymbol{y}_k]^\mathrm{T}$ and the orientation $\boldsymbol{\phi}_k$ of the bicycle. This type of kinematics is often also employed for modeling cars. The sensor model accords (2.4), where $[x_k^s, y_k^s]^\mathrm{T} = [0, 0]^\mathrm{T}$. The noise $\underline{\boldsymbol{w}}_k = [\boldsymbol{w}_k^v, \boldsymbol{w}_k^\alpha]^\mathrm{T}$ and $\underline{\boldsymbol{v}}_k$ are zero-mean white Gaussian with covariance matrix $\mathbf{C}_k^w = \mathrm{diag}\left([0.001\ \mathrm{m}^2/\mathrm{s}^2, 0.001\ \mathrm{rad}^2]\right)$ and $\mathbf{C}_k^v = 0.001\ \mathrm{m}^2$, respectively.

For simulation purposes, constant inputs $v_k = 5\ \mathrm{m/s}$, which is the velocity, and $\alpha_k = 0.05\ \mathrm{rad}$, which is the steering angle, are used. The initial system state $\underline{\boldsymbol{x}}_0$ has the mean $\underline{\hat{x}}_0 = [0\ \mathrm{m}, 0\ \mathrm{m}, 0\ \mathrm{rad}]^\mathrm{T}$ and covariance matrix $\mathbf{C}_0 = \mathrm{diag}([0.1\ \mathrm{m}^2, 0.1\ \mathrm{m}^2, 0.01\ \mathrm{rad}^2])$. With this configuration 100 Monte Carlo simulation runs are performed. Each run consists of 50 alternating prediction and measurement update steps. The Gaussian estimator (GE) employs the axis-aligned approach with $L = 5$ regression points per dimension and the parameters of the UKF are set to $\alpha = 1$, $\beta = 0$, and $\kappa = 0$.

In Figure 5.4 (a), the root mean square error (rmse) of the position estimate $[\boldsymbol{x}_k, \boldsymbol{y}_k]^\mathrm{T}$ of the first 50 simulation runs are depicted. Together with Table 5.4, where the average rmse over all simulations runs for each element of the state vector is listed, it can be seen that the GE

Table 5.4: Average rmse over 100 simulation runs.

	rmse_x	rmse_y	rmse_ϕ
UKF	3.5068	2.5991	0.2460
GE	2.5125	1.8655	0.2036

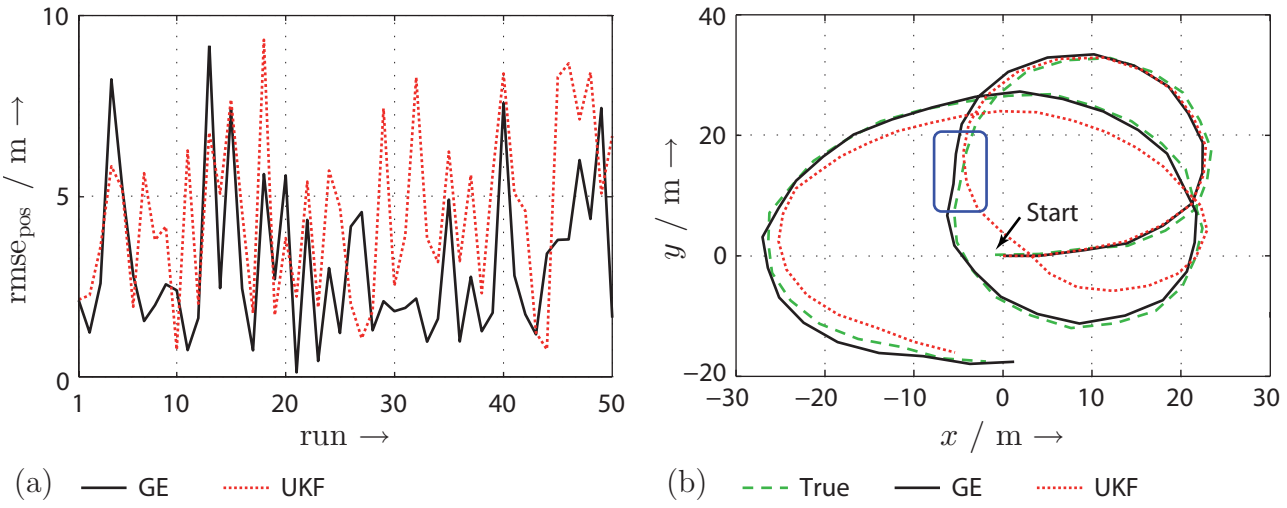


Figure 5.4: (a) Rmse of the position estimates of the GE and the UKF for the first 50 simulation runs. (b) Example simulation run. The true trajectory of the vehicle in $[\hat{x}_k, \hat{y}_k]^{\mathrm{T}}$ (green, dashed) is depicted together with the estimates of the Gaussian estimator (black, solid) and the UKF (red, dotted).

significantly outperforms the estimation results of the UKF, while the computation time is in the same order of magnitude; GE needs 0.084 s per run on average and the UKF needs 0.055 s.[2] For instance in 41 out of the first 50 runs the GE provides better position estimates and in 79 out of all 100 runs the GE provides better orientation estimates ϕ_k than the UKF.

One reason why the results of the UKF are inferior is illustrated in Figure 5.4 (b), where the rmse of both estimators for this particular simulation run is almost identical to the average rmse listed in Table 5.4. Here, the estimated trajectory of the UKF begins to strongly deviate from time step $k = 18$ on (indicated by the blue rectangle in Figure 5.4 (b)), caused by a strong estimation error of the orientation. This error in turn is caused by the statistical linearization of the UKF. As the orientation is not directly observable by means of the sensor model, linearization errors are more severe. The GE instead provides a more sophisticated representation of the state density comprising a larger sample set of regression points. This offers statistical linearization with higher-order accuracy and thus, the GE behaves more robust. The same argument holds for the strong nonlinearity of the system model introduced by the tangent function.

Mobile Sensor Control

In this simulation scenario, the target with bicycle kinematics (5.21) is tracked by means of a mobile distance sensor. Compared to the previous simulation, the system noise covariance of the target is increased in order to force a diverging estimation results for myopic sensor management. Hence, the noise covariance is set to be $\mathbf{C}_k^w = \mathrm{diag}([0.01\ \mathrm{m}^2, 0.0025\ \mathrm{rad}^2])$. These noise terms affect the target velocity and steering angle, which are set to be $v_k = 6$ m/s and $\alpha_k = 0$ rad, respectively. The initial estimate of the pose of the target $\underline{\boldsymbol{x}}_0$ has the mean $\underline{\hat{x}}_0 = [0\ \mathrm{m}, 500\ \mathrm{m}, 5\pi/4\ \mathrm{rad}]^{\mathrm{T}}$ and the covariance $\mathbf{C}_0 = \mathrm{diag}([500\ \mathrm{m}^2, 500\ \mathrm{m}^2, 1\ \mathrm{rad}^2])$.

The sensor behaves analogously to the simulation in Section 3.3.5, i.e., the kinematic model of the sensor accords (2.6), where the sensor moves with a velocity of $v = 20$ m/s after changing its orientation by one of five possible steering angles $u \in \{\frac{i\pi}{8} | i = -2, \ldots, 2\}$. The sensor can further stop its motion for the considered time step. The initial sensor position and the measurement noise are $[x_k^s, y_k^s, \phi_k^s]^{\mathrm{T}} = [-100\ \mathrm{m}, 300\ \mathrm{m}, \pi/2\ \mathrm{rad}]^{\mathrm{T}}$ and $\sigma_k^v = 0.5$ m, respectively.

2 Computation times are based on a Matlab 7.5 implementation running on a Intel Core2 Duo 2×2.4 GHz PC.

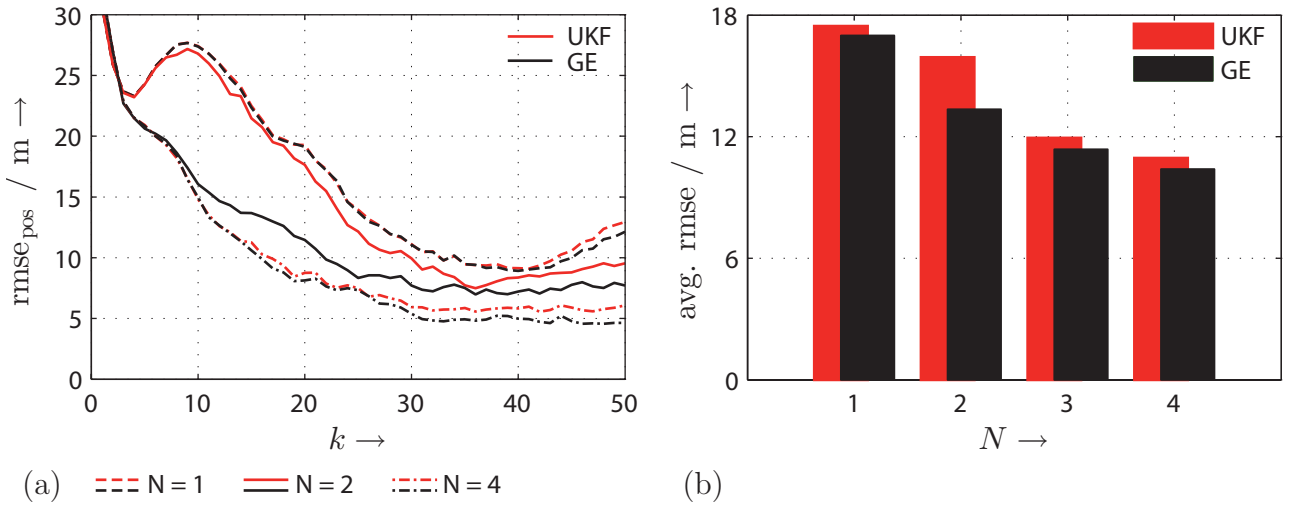


Figure 5.5: (a) Rmse of the position estimates of Gaussian estimator and unscented Kalman filter for time horizons $N \in \{1, 2, 4\}$. (b) Average rmse of both estimators for time horizons $N \in \{1, 2, 3, 4\}$.

This simulation aims at demonstrating the effect of different regression point calculation schemes for statistical linearization on the estimation performance of the quasi-linear sensor manager proposed in Chapter 3. For this purpose, the deterministic sampling scheme of the GE is compared with the corresponding scheme of the UKF. For both estimators, the time horizons $N \in \{1, 2, 3, 4\}$ are investigated in 100 Monte Carlo runs. The rmse of the position estimates over all simulation runs for the first 50 time steps are depicted in Figure 5.5 (a). For all considered time horizons, the quasi-linear sensor manager employing the GE provides better estimates[3]. The strongest deviation of the estimation results occurs for the time horizon $N = 2$. Here, planning with the UKF often leads to sensor trajectories for the time steps five to ten, where the sensor is not appropriately placed for performing informative measurements and thus, for reducing the overall uncertainty of the target position. Hence, the estimation performance is only slightly better as it the case for myopic sensor management, i.e., for $N = 1$. Only by employing a longer lookahead, i.e., for $N > 2$, the UKF allows for a rapid convergence of the estimation results, as indicated by Figure 5.5 (b).

In contrast, planning based on the GE controls the sensor maneuvers in such a way that a rapid convergence is even possible for $N = 2$. Thanks to the improved regression point selection of the GE, the linearization error can be kept on a lower level compared to the UKF. This is of particular importance for open-loop sensor management, where the linearization of the nonlinear system and sensor models is performed in a predictive fashion and thus, the linearization errors accumulate over the time horizon.

5.5 Gaussian Mixture Estimator

In many practical nonlinear estimation problems, representing the state or the noise density by means of a single Gaussian is not adequate, especially in cases of multimodal or highly skewed distributions. These non-Gaussian densities can be represented accurately by means of Gaussian mixtures, i.e., a finite sum of weighted Gaussians (see Section A.2). Due to their universal approximation property [124], Gaussian mixtures are very convenient for that purpose.

In the following, a Gaussian mixture estimator for estimation problems with Gaussian mixture state and noise densities is introduced. Basically, the Gaussian mixture estimator

3 The results for $N = 3$ are not depicted for clarity reasons.

applies the previously derived Gaussian estimator to each Gaussian component of the given Gaussian mixtures.

5.5.1 Prediction Step

Assume that at time step k the density of the estimated state $\underline{\boldsymbol{x}}_k^e$ is represented by the Gaussian mixture

$$f_k^e(\underline{x}_k) = \sum_{i=1}^{L_e} \omega_{k,i}^e \cdot \mathcal{N}\left(\underline{x}_k; \underline{\hat{x}}_{k,i}^e, \mathbf{C}_{k,i}^e\right) \ ,$$

with $\sum_i \omega_{k,i}^e = 1$. Furthermore, the density of the system noise $\underline{\boldsymbol{w}}_k$ is given by

$$f_k^w(\underline{w}_k) = \sum_{j=1}^{L_w} \omega_{k,j}^w \cdot \mathcal{N}\left(\underline{w}_k; \underline{\hat{w}}_{k,j}, \mathbf{C}_{k,j}^w\right) \ ,$$

with $\sum_j \omega_{k,j}^w = 1$. Applying the Chapman-Kolmogorov equation (2.7) of the optimal Bayesian estimator on both mixtures yields the predicted density

$$\begin{aligned} f_{k+1}^p(\underline{x}_{k+1}) &= \int_{\mathbb{R}^{nx}} \int_{\mathbb{R}^{nw}} \delta(\underline{x}_{k+1} - \underline{a}_k(\underline{x}_k, \underline{w}_k)) \cdot f_k^w(\underline{w}_k) \cdot f_k^e(\underline{x}_k) \, \mathrm{d}\underline{x}_k \, \mathrm{d}\underline{w}_k \qquad (5.22) \\ &= \sum_{i=1}^{L_e} \sum_{j=1}^{L_w} \omega_{k,i}^e \cdot \omega_{k,j}^w \cdot \\ &\quad \underbrace{\int_{\mathbb{R}^{nx}} \int_{\mathbb{R}^{nw}} \delta(\underline{x}_{k+1} - \underline{a}_k(\underline{x}_k, \underline{w}_k)) \cdot \mathcal{N}\left(\underline{x}_k; \underline{\hat{x}}_{k,i}^e, \mathbf{C}_{k,i}^e\right) \cdot \mathcal{N}\left(\underline{w}_k; \underline{\hat{w}}_{k,j}, \mathbf{C}_{k,j}^w\right) \mathrm{d}\underline{x}_k \, \mathrm{d}\underline{w}_k}_{\approx \mathcal{N}\left(\underline{x}_{k+1}; \underline{\hat{x}}_{k+1,i,j}^p, \mathbf{C}_{k+1,i,j}^p\right) \text{ (prediction step of Gaussian estimator)}} \ . \end{aligned}$$

The integral on the right hand side corresponds to performing a prediction step for the prior density given by the product of the Gaussians $f_{k,i}^x(\underline{x}_k) := \mathcal{N}\left(\underline{x}_k; \underline{\hat{x}}_{k,i}^e, \mathbf{C}_{k,i}^e\right)$ and $f_{k,j}^w(\underline{w}_k) := \mathcal{N}\left(\underline{w}_k; \underline{\hat{w}}_{k,j}, \mathbf{C}_{k,j}^w\right)$ that represent the state and the noise, respectively. Due to the nonlinearity of the system function $\underline{a}_k(\cdot)$, evaluating the integral in closed form is impossible, in general. However, an approximate solution can be obtained by applying the prediction step of the Gaussian estimator. This corresponds to a local statistical linearization of the nonlinear function $\underline{a}_k(\cdot)$ around the uncertainty regions characterized by $f_{k,i}^x(\underline{x}_k)$ and $f_{k,j}^w(\underline{w}_k)$, respectively. In doing so, the integral is replaced by a predicted Gaussian $\mathcal{N}\left(\underline{x}_{k+1}; \underline{\hat{x}}_{k+1,i,j}^p, \mathbf{C}_{k+1,i,j}^p\right)$ with mean $\underline{\hat{x}}_{k+1,l}^p$ and covariance $\mathbf{C}_{k+1,l}^p$ according to (5.17) and (5.18), respectively. Performing the prediction step of the Gaussian estimator for each combination of Gaussians $f_{k,i}^x(\underline{x}_k)$ and $f_{k,j}^w(\underline{w}_k)$ results in the Gaussian mixture

$$\begin{aligned} f_{k+1}^p(\underline{x}_{k+1}) &\approx \sum_{i=1}^{L_e} \sum_{j=1}^{L_w} \omega_{k+1,i,j}^p \cdot \mathcal{N}\left(\underline{x}_{k+1}; \underline{\hat{x}}_{k+1,i,j}^p, \mathbf{C}_{k+1,i,j}^p\right) \\ &= \sum_{l=1}^{L_p} \omega_{k+1,l}^p \cdot \mathcal{N}\left(\underline{x}_{k+1}; \underline{\hat{x}}_{k+1,l}^p, \mathbf{C}_{k+1,l}^p\right) \qquad (5.23) \end{aligned}$$

for approximating the predicted density, where $l = (i-1) \cdot L_w + j$ and $L_p = L_e \cdot L_w$. It is worth mentioning that all means and covariance matrices of the Gaussian mixture (5.23) can be calculated independently, i.e., the required L_p prediction steps of the Gaussian estimator can be performed in parallel for calculating the means and covariance matrices.

Parallel Weight Calculation

So far, merely calculating the new weights $\omega^p_{k+1,l}$ is left unconsidered. A straightforward way for updating the weights is to do it in parallel as already done for the means and covariances. Here, the predicted weights are given by

$$\omega^p_{k+1,l} = \omega^e_{k,i} \cdot \omega^w_{k,j} \ .$$

This weight update is exact for linear system models and it corresponds to the weight update proposed for the well-known Gaussian sum filter [5], where the prediction step of the EKF is applied on (5.22).

Simultaneous Weight Calculation

However, for the considered nonlinear models, it is more desirable to simultaneously calculate the weights in order to improve the approximation quality of the predicted Gaussian mixture. A simultaneous weight calculation is advantageous in particular when predictions steps have to be performed consecutively without a measurement update in-between or when the measurement noise is strong. For the Gaussian sum filter, a simultaneous weight calculation is derived in [177], which can be adapted for the proposed Gaussian mixture estimator in a straightforward manner.

5.5.2 Measurement Update Step

Analogously to the prediction step, state and noise Gaussian mixtures are processed component-wise in the measurement update step of the Gaussian mixture estimator. Assume that the measurement noise is represented by means of the Gaussian mixture

$$f^v_k(\underline{v}_k) = \sum_{j=1}^{L_v} \omega^v_{k,j} \cdot \mathcal{N}\left(\underline{v}_k; \underline{\hat{v}}_{k,j}, \mathbf{C}^v_{k,j}\right) \ ,$$

with $\sum_j \omega^v_{k,j} = 1$. After receiving the measurement value $\underline{\hat{z}}_k$, the posterior density $f^e_k(\underline{x}_k)$ of the state can be approximated by the Gaussian mixture

$$\begin{aligned} f^e_k(\underline{x}_k) &\approx \sum_{l=1}^{L_p} \sum_{j=1}^{L_v} \omega^e_{k,l,j} \cdot \mathcal{N}\left(\underline{x}_k; \underline{\hat{x}}^e_{k,l,j}, \mathbf{C}^e_{k,i,j}\right) \\ &= \sum_{i=1}^{L_e} \omega^e_{k,i} \cdot \mathcal{N}\left(\underline{x}_k; \underline{\hat{x}}^e_{k,i}, \mathbf{C}^e_{k,i}\right) \ , \end{aligned} \tag{5.24}$$

where $i = (l-1) \cdot L_v + j$ and $L_e = L_p \cdot L_v$. This approximation results from firstly plugging the mixtures $f^p_k(\underline{x}_k)$ and $f^v_k(\underline{v}_k)$ into the Bayesian measurement update equation (2.9) and then from applying the measurement update step of the Gaussian estimator in a parallel manner on each possible combination of the Gaussian components of $f^p_k(\underline{x}_k)$ and $f^v_k(\underline{v}_k)$. This procedure yields the means $\underline{\hat{x}}^e_{k,i}$ and covariances $\mathbf{C}^e_{k,i}$.

The calculation of the weight can be adapted from to the prediction step. Hence, the computationally less demanding parallel weight update is given by

$$\bar{\omega}^e_{k,i} = \omega^p_{k,l} \cdot \omega^v_{k,j} \cdot c_{k,l,j} \ ,$$

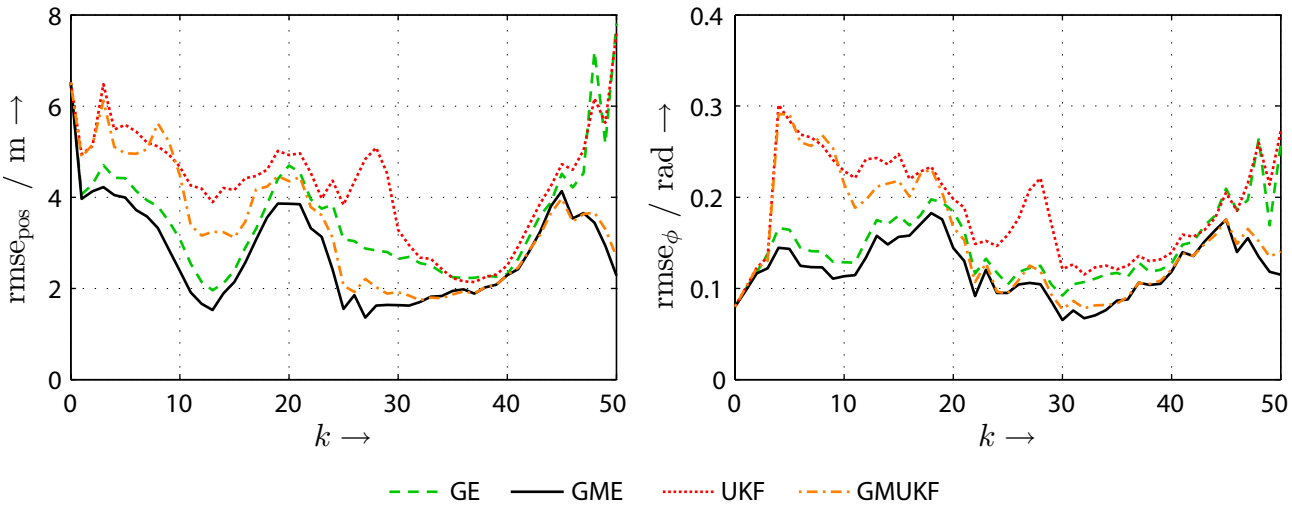


Figure 5.6: Target tracking performance in presence of Gaussian mixture system noise. The rmse in the position (right plot) and in the orientation (left plot) for the Gaussian estimator (GE), the Gaussian mixture estimator (GME), the UKF, and the Gaussian mixture variant of the UKF (GMUKF).

with $c_{k,l,j} = \mathcal{N}\left(\hat{\underline{z}}_k; \hat{\underline{z}}^p_{k,l,j}, \mathbf{C}^z_{k,l,j}\right)$, where $\hat{\underline{z}}^p_{k,l,j}$, $\mathbf{C}^z_{k,l,j}$ are defined by (5.19) and (5.20), respectively. After normalization the desired weights of the posterior Gaussian mixture are

$$\omega^e_{k,i} = \frac{\bar{\omega}^e_{k,i}}{\sum_i \bar{\omega}^e_{k,i}} \ .$$

Again, this kind of parallel weight calculation corresponds to the Gaussian sum filter [5] and is only exact for linear sensors. A more accurate but computationally demanding simultaneous weight calculation for the measurement update step is given in [85].

Remark 5.1 (Exponential Growth) Gaussian mixture extensions exist almost for each LRKF. As noted previously, the Gaussian sum filter extends the EKF, [168] proposes a Gaussian mixture variant for the UKF, and [85] introduces the Gauss–Hermite filter together with its mixture extension. Like the previously introduced Gaussian mixture estimator, all these extensions of existing LRKFs have one problem in common. Examine the Gaussian mixture densities in (5.23) and (5.24), it stands out that the number of mixture components increases with each prediction and measurement update step. More precisely, if at least one of the noise densities is represented by means of a Gaussian mixture, the number of components in $f^p_k(\underline{x}_k)$ and $f^e_k(\underline{x}_k)$ grows exponentially over the time, which reduces the practicability of the Gaussian mixture estimator significantly. In order to keep this growth bounded, Gaussian mixture reduction techniques have to be applied from time to time; at each time step in extreme cases. A very accurate Gaussian mixture reduction algorithm is introduced in Chapter 7.

5.5.3 Simulation Example

The increased estimation performance of the proposed Gaussian mixture extension is demonstrated in this section by revisiting the simulation example exploited in Section 5.4.2. But now, the system noise $\underline{\boldsymbol{w}}_k$ is represented by the bimodal Gaussian mixture

$$f^w_k(\underline{w}_k) = 0.5 \cdot \mathcal{N}\left(\underline{w}_k; [-1, 0]^{\mathrm{T}}, \mathbf{C}\right) + 0.5 \cdot \mathcal{N}\left(\underline{w}_k; [1, 0]^{\mathrm{T}}, \mathbf{C}\right) \ ,$$

with $\mathbf{C} = \mathrm{diag}\left([0.001\ \mathrm{m}^2/\mathrm{s}^2, 0.0001\ \mathrm{rad}^2]\right)$. Together with the input $v_k = 5\ \mathrm{m/s}$, this noise representation reveals that the uncertainty over the target velocity can restricted to variations

around 4 m/s or 6 m/s. The motion of the vehicle is now observed via a bearings sensor with sensor model (2.5). The sensor is located in the origin, i.e., $[x_k^s, y_k^s]^{\mathrm{T}} = [0, 0]^{\mathrm{T}}$, and the measurement noise is zero-mean Gaussian with standard deviation $\sigma_k^v = 0.01$ rad.

For this setup, the proposed Gaussian estimator (GE) together with its Gaussian mixture extension (GME) are compared to the UKF and its Gaussian mixture variant (GMUKF). Due to the Gaussian mixture representation of the system noise, Gaussian mixture reduction is required for remaining the complexity for the GME and GMUKF bounded. For this purpose, the progressive Gaussian mixture reduction algorithm introduced in Section 7 is applied for the mixture estimators whenever the number of Gaussian components exceeds 128 components. The mixtures are then reduced to a maximum of 8 components.

The rmse in the position and orientation are obtained in 100 Monte Carlo simulation runs. In Figure 5.6, the rmse in the position in x direction and in the orientation are depicted. It is observed that the GME provides the best tracking performance. Especially for the first 30 time steps, the GME provides significantly better estimation results than the GMUKF. The GME instead benefits from the superior deterministic sampling scheme of the GE, while the worse statistical linearization results of the UKF also have a strong impact on its mixture extension. This impact is such considerable, that the tracking performance of the GMUKF is even inferior to the performance of the GE for the first 20 time steps. But from time step 40 on, both single Gaussian estimators are not longer capable of dealing with the Gaussian mixture system noise and the tracking accuracy degrades drastically.

5.6 Summary

In this section, the approach of linear regression Kalman filters is extended by the idea of interpreting the regression points as analytic density function, namely a Dirac mixture. In doing so, directly approximating the distribution function of a Gaussian by the regression points is possible and higher-order information of the Gaussian density is implicitly incorporated. This way of approximating a Gaussian can be interpreted as deterministic sampling, where the samples, i.e., the regression points, are placed in a deterministic manner in order to accurately approximate the shape of the distribution function and to exactly capture mean and variance. While computationally demanding parts of this approximation are carried out off-line, adapting the regression points on the current Gaussian density is performed on-line.

For multivariate Gaussians, two approximation techniques are introduced. The grid approach places the regression points in form of a irregular grid and thus, leads to an extensive covering of the state domain. The axis-aligned approach is more scalable as it aims on covering the principal axes of the covariance ellipsoid. Both approaches result in a Gaussian estimator with simple structure. In terms of the computational complexity, especially the estimator based on the axis-aligned approach is comparable to typical LRKFs like the famous unscented Kalman filter. However, in contrast to most of the LRKFs, the number of regression points is adjustable, which allows altering the approximation quality. Thus, by increasing the number of regression points, nonlinearities of the state transformation can be captured more accurately. In case of very high-dimensional estimation problems, the size of the set of regression points can be kept on a minimum for ensuring low computation times. As demonstrated in the simulations, a small number of regression points suffices to outperform the UKF.

The proposed Gaussian estimator can be also applied to non-Gaussian scenarios, where the state and the noise densities are represented by means of Gaussian mixtures. Here, the extension is straightforward, as the Gaussian estimator can be applied component-wise. Merely the calculation of the updated weights is more challenging. Beside the well-known suboptimal parallel weight calculation, more accurate but also more demanding simultaneous calculation

procedures can be employed. The problem of the exponential growth of the number of mixture components is handled in the Chapter 7.

Even if the proposed Gaussian mixture estimator already provides superior estimation performance, further extensions are possible. Currently, merely the Gaussian components of the current state density and the noise densities are exploited. In cases of strong nonlinearities this may not suffice. Here, strong nonlinear regions of the considered probabilistic models could by captured more accurately by the introduction of new components. This requires the systematic and efficient identification of relevant regions. A basic approach towards an efficient identification can be found in [70]. The introduction of new components can be achieved for example by the splitting method employed in Section 7.2.2.

CHAPTER 6

The Hybrid Density Filter

For the considered nonlinear system and sensor models, a closed-form solution of the Bayesian estimator is impossible to obtain in general. However, an accurate inference of the system state after applying the configuration vector to the sensors is essential, especially in case of non-Gaussian multimodal density functions. Besides that, an accurate non-Gaussian density representation is also of paramount importance for state estimation during planning and for measurement prediction, particularly for the information theoretic sensor management scheme introduced in Chapter 4. While a great effort can be spent for state inference after planning in order to achieve the desired accuracy, state estimation during planning necessitates a trade-off between computational burden and accuracy. To meet these demands, the *hybrid density filter* (HDF), an approximate but accurate estimator for nonlinear non-Gaussian state estimation, is introduced in this chapter.

While most of the existing estimators focus on directly approximating the density function representing the system state, the HDF is based on optimally approximating the conditional densities, i.e., the transition density and the likelihood, which are probabilistic representations of the underlying nonlinear system and sensor models. For approximating the conditional density and as inspiration for naming this estimation approach, a so-called *hybrid density* is employed that consists of Dirac delta distributions and Gaussian densities. To optimally adapt the parameters of the hybrid density to the conditional density, the approximation problem is reformulated as an optimization problem for minimizing the Cramér-von Mises distance. This special type of a squared integral measure, which was also utilized for the Gaussian estimator in Chapter 5, allows quantifying the deviation especially in case of the used Dirac delta distributions and thus, is defined over the corresponding cumulative distribution functions of the true and the approximate conditional density. In case of scalar states, this optimization problem can be solved optimally in closed form. An approximate but easy to calculate solution for the multivariate case is presented in Section 6.3.

The hybrid structure of the approximate conditional density facilitates an analytical and efficient evaluation of the prediction step as well as the measurement update step (see Section 6.2). While performing the prediction step is straightforward and results in a Gaussian mixture representation of the predicted density, measurement updating requires an additional, easy to compute interpolation step for retaining a Gaussian mixture representation. However, a very accurate approximation in shape and moments of the resulting complex density function of the system state is achieved. All approximation components of the HDF are depicted in Figure 6.1. As discussed in Section 6.4, predictions or measurement updates by means of the hybrid conditional density approximation can also be interpreted as sampling the prior density deterministically. This deterministic sampling interpretation gives a straightforward way for implementation.

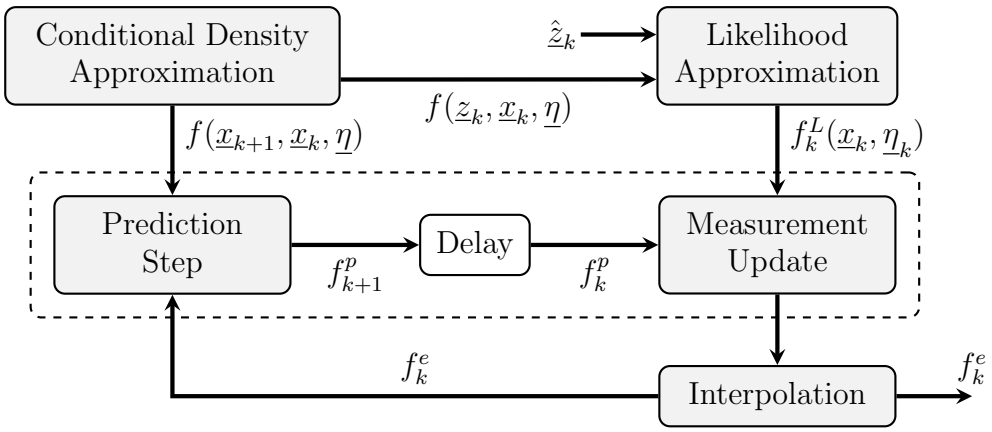


Figure 6.1: Structure of the HDF prediction and measurement update step. The dashed box highlights the components of the Bayesian estimator, while all other components contain the approximations that are necessary for performing estimations by means of the HDF.

The HDF for scalar states was published in [215, 216], while the extension to the multivariate case represents unpublished material. The proposed estimator was applied to stochastic model predictive control problems [226, 230] and parameter identification problems [222].

6.1 Conditional Density Approximation

In this section, only scalar states are considered for brevity and clarity. The extension to multivariate states is content of Section 6.3. Furthermore, it is assumed that the nonlinear system model

$$\boldsymbol{x}_{k+1} = a_k(\boldsymbol{x}_k) + \boldsymbol{w}_k \tag{6.1}$$

and the nonlinear sensor model

$$\boldsymbol{z}_k = h_k(\boldsymbol{x}_k) + \boldsymbol{v}_k \ , \tag{6.2}$$

are affected by additive white Gaussian noise $\boldsymbol{w}_k \sim f_k^w(w_k) = \mathcal{N}(w_k; \hat{w}_k, (\sigma_k^w)^2)$ and $\boldsymbol{v}_k \sim f_k^v(v_k) = \mathcal{N}(v_k; \hat{v}_k, (\sigma_k^v)^2)$, respectively. Here, $a_k(\,\cdot\,)$ and $h_k(\,\cdot\,)$ are nonlinear functions with at most a finite number of points of discontinuities, and $\boldsymbol{x}_k \in \Omega_k$ is the scalar system state at time step k with density $f_k^x(x_k)$ and support

$$\Omega_k := [\alpha_k, \beta_k] \subset \mathbb{R} \ ,$$

where $\forall x_k \in \Omega_k : f_k^x(x_k) > \epsilon$ for constant ϵ with $0 < \epsilon \ll 1$. It is further assumed that $a_k(\,\cdot\,)$ and $h_k(\,\cdot\,)$ are bounded functions on Ω_k.

6.1.1 Conditional Densities in Bayesian Estimation

Thanks to the assumption of additive noise, the transition density (2.8) required for performing the prediction step can be simplified to

$$f_k^T(x_{k+1}|x_k) = f_k^w(x_{k+1} - a_k(x_k)) \ .$$

Similarly, the likelihood, which is employed at the Bayesian measurement update step (2.9) can be written according to

$$f_k^L(\hat{z}_k|x_k) = f_k^v(\hat{z}_k - h_k(x_k)) \ .$$

Both, the likelihood and the transition density are based upon conditional densities, which in turn depend on the noise densities $f_k^v(v_k)$ and $f_k^w(w_k)$ as well as on the structure of the sensor and system model, respectively. Additionally, the likelihood depends on the actual measurement $\hat{z}_k$. Since (6.1) and (6.2) are time-variant and both noise processes are non-stationary, transition density and likelihood are also time-variant, i.e., their shapes change with time index k.

Generally, recursive Bayesian estimation for nonlinear systems is of conceptual value only, since the complex shapes of the conditional densities prevent a closed-form and efficient solution. Furthermore, for the case of nonlinear systems with arbitrarily distributed random variables, there exists no analytic density that can be calculated without changing the type of representation in general. To overcome this problem, an appropriate approximation is inevitable. From now on, true densities will be indicated by a tilde, e.g. $\tilde{f}(\,\cdot\,)$, while the corresponding approximation will be denoted by $f(\,\cdot\,)$.

Instead of directly approximating the densities $f_{k+1}^p(x_{k+1})$ and $f_k^e(x_k)$ resulting from (2.7) and (2.9), respectively, which is computationally demanding, the key idea is to approximate the conditional density

$$\tilde{f}(y|x) = \mathcal{N}(y; \psi(x), \sigma^2) \ .$$

Here, $\psi(\,\cdot\,)$ represents a nonlinear function over the random variable $\boldsymbol{x} \in \Omega = [\alpha, \beta] \subset \mathbb{R}$. To obtain for example the transition density at time step k, one has to substitute y, x, $\psi(\,\cdot\,)$ and σ such that

$$\tilde{f}_k^T(x_{k+1}|x_k) = \tilde{f}(y|x)\Big|_{y=x_{k+1},\, x=x_k,\, \psi(\,\cdot\,)=a_k(\,\cdot\,),\, \sigma=\sigma_k^w} \ .$$

6.1.2 Hybrid Density

As a novel type of density representation for approximating the conditional density $\tilde{f}(y|x)$, the (unnormalized) *hybrid density*

$$f(y, x; \underline{\eta}) = \sum_{i=1}^{L} \omega_i \cdot \delta(x - \hat{x}_i) \cdot \mathcal{N}(y; \hat{y}_i, (\sigma_i^y)^2) \tag{6.3}$$

with parameter vector

$$\underline{\eta} = [\underline{\eta}_1^{\mathrm{T}}, \underline{\eta}_2^{\mathrm{T}}, \ldots, \underline{\eta}_L^{\mathrm{T}}]^{\mathrm{T}}, \text{ where } \underline{\eta}_i^{\mathrm{T}} = [\omega_i, \hat{x}_i, \hat{y}_i, \sigma_i^y] \ .$$

is proposed. Here, L is the number of components, ω_i are weighting coefficients with $\omega_i > 0$, $\hat{x}_i$ and $\hat{y}_i$ are the means, and σ_i^y is the standard deviation of $\mathcal{N}(y; \hat{y}_i, (\sigma_i^y)^2)$. The marginal densities of a hybrid density consist of two different types of analytic densities: the densities of $\boldsymbol{x}$ are Dirac delta distributions $\delta(x_k - \hat{x}_i)$ and the densities of $\boldsymbol{y}$ are Gaussians $\mathcal{N}(y; \hat{y}_i, (\sigma_i^y)^2)$.

Example 6.1: Hybrid Density
The nonlinear system model

$$\boldsymbol{x}_{k+1} = \sin(\boldsymbol{x}_k) + \boldsymbol{x}_k + \boldsymbol{w}_k \tag{6.4}$$

represents parts of the kinematic model (5.21). The density of the system noise $\boldsymbol{w}_k$ is given by $f_k^w(w_k) = \mathcal{N}(w_k; 0, 1)$. Figure 6.2 (a) depicts the system function (black, dashed line) and the corresponding transition density. Furthermore, the red lines in Figure 6.2 (a) and Figure 6.2 (b) illustrate a hybrid density with $L = 4$ components. Due to the Dirac delta distributions, a single hybrid density component can be interpreted as a vertical *slice* of the transition density. ■

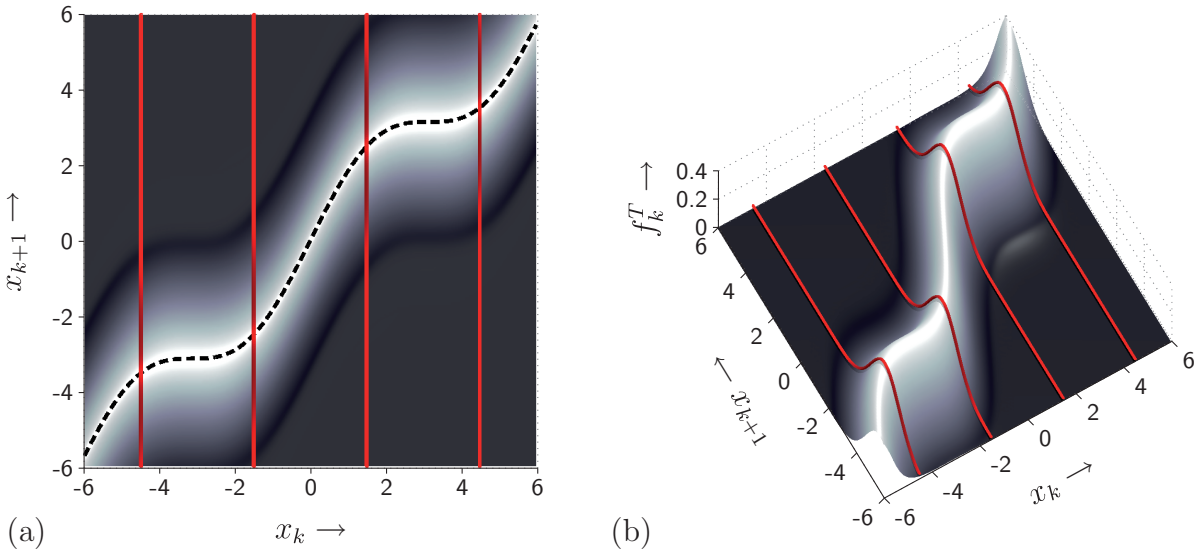


Figure 6.2: Top view (a) and perspective view (b) on the transition density of the system $\boldsymbol{x}_{k+1} = \sin(\boldsymbol{x}_k) + \boldsymbol{x}_k + \boldsymbol{w}_k$. The shape of the transition density strongly depends on the system function (black, dashed line). An approximate hybrid density with $L = 4$ components slices the transition density (red, solid lines) in 5 parts. Each slice is a Gaussian density.

The goal is now to minimize a certain distance measure $G(\underline{\eta})$ between the true conditional density $\tilde{f}(y|x)$ and its approximation $f(y, x; \underline{\eta})$. Generally, the calculation of an appropriate parameter vector $\underline{\eta}$ for a high quality approximation of the transition density is computationally demanding. Since the conditional densities can be time-variant, these calculations are required at every time step. By selecting a hybrid density for approximation purposes, the required computational effort can be drastically reduced and on-line approximation at every time step is possible. With the given conditional density approximation, the prediction and measurement update step can be performed efficiently in closed form, as illustrated in Figure 6.1.

6.1.3 Optimal Approximation

The approximation quality of the HDF strongly depends on the similarity between $\tilde{f}(y|x)$ and its hybrid density approximation $f(y, x; \underline{\eta})$. Hence, the approximation problem is reformulated into an optimization problem

$$\underline{\eta}_{\min} = \arg \min_{\underline{\eta}} G(\underline{\eta}) \ , \tag{6.5}$$

with the objective of minimizing a certain distance measure $G(\underline{\eta})$. The result of this optimization problem yields the parameter vector $\underline{\eta}$ for $f(y, x; \underline{\eta})$, which minimizes the distance to $\tilde{f}(y|x)$.

As for the Gaussian estimator developed in Chapter 5, distance measures defined over the density space cannot be applied directly due to the used Dirac delta distributions in (6.3). Quantifying the deviation between $\tilde{f}(y|x)$ and $f(y, x; \underline{\eta})$ is only possible when considering the corresponding cumulative distribution functions instead. The distribution function of the true conditional density $\tilde{f}(y|x)$ for $x \in \Omega = [\alpha, \beta]$ can be written as

$$\tilde{F}(y|x) = \frac{1}{2} \int_{\alpha}^{x} 1 + \operatorname{erf}\left(\frac{y - \psi(s)}{\sqrt{2}\sigma} \right) \mathrm{d}s \ .$$

The *hybrid distribution function* of $f(y, x; \underline{\eta})$ is given by

$$F(y, x; \underline{\eta}) = \frac{1}{2} \sum_{i=1}^{L} \omega_i \cdot H\left(x - \hat{x}_i\right) \cdot \left(1 + \mathrm{erf}_i(y)\right) \ , \tag{6.6}$$

where

$$\mathrm{erf}_i(y) := \mathrm{erf}\left(\frac{y - \hat{y}_i}{\sqrt{2}\sigma_i^y}\right)$$

is the error function of component i and $H(x - \hat{x})$ is the Heaviside step function at position $\hat{x}$ as defined in (5.4).

As distance measure the so-called *Cramér-von Mises distance* [24]

$$G(\underline{\eta}) = \frac{1}{2} \int_{\mathbb{R}} \int_{\Omega} \left(\tilde{F}(y|x) - F(y, x; \underline{\eta})\right)^2 \mathrm{d}x \, \mathrm{d}y \tag{6.7}$$

is employed. Normally, the underlying nonlinearity complicates solving (6.5), as pointed out for example in [214] for a pure Gaussian mixture representation of the conditional density approximation. Thanks to the special structure of the hybrid distribution (6.6), the optimal solution can easily be derived in closed form.

Theorem 6.1 (Optimal Approximation) *Given the distance measure* (6.7)*, the elements* $\underline{\eta}_i^{\mathrm{T}} = [\omega_i, \hat{x}_i, \hat{y}_i, \sigma_i^y]$ *of the optimal solution* $\underline{\eta}_{\min}$ *of the optimization problem* (6.5) *are*

$$\begin{aligned}
&\textit{(Equal weights)} && \omega_i = \tfrac{\beta - \alpha}{L} \ , \\
&\textit{(Uniformly distributed Diracs)} && \hat{x}_i = \alpha + \omega_i \cdot \tfrac{2i-1}{2} \ , \\
&\textit{(Nonlinearly shifted Gaussians)} && \hat{y}_i = \psi(\hat{x}_i) \ , \\
&\textit{(Noise standard deviation)} && \sigma_i^y = \sigma \ .
\end{aligned}$$

At first, a lemma is developed, which is used to prove Theorem 6.1.

Lemma 6.1 *Given the distance measure*

$$G^*(\underline{\eta}) = \frac{1}{2} \int_{\Omega} \left(\tilde{F}(x) - F(x; \underline{\eta})\right)^2 \mathrm{d}x \tag{6.8}$$

for $\boldsymbol{x} \in \Omega$ *over the distribution functions*

$$\begin{aligned}
\tilde{F}(x) &= \int_{\alpha}^{x} \tilde{f}(s) \, \mathrm{d}s = x - \alpha \ , \\
F(x; \underline{\eta}) &= \int_{\alpha}^{x} f(s, \underline{\eta}) \, \mathrm{d}s = \sum_{i=1}^{L} \omega_i \cdot H(x - \hat{x}_i)
\end{aligned} \tag{6.9}$$

of the marginal densities

$$\begin{aligned}
\tilde{f}(x) &= \int_{\mathbb{R}} \tilde{f}(y|x) \, \mathrm{d}y = 1 \ , \\
f(x; \underline{\eta}) &= \int_{\mathbb{R}} f(y, x; \underline{\eta}) \, \mathrm{d}y = \sum_{i=1}^{L} \omega_i \cdot \delta(x - \hat{x}_i) \ .
\end{aligned} \tag{6.10}$$

Minimizing (6.8) *results in*

$$\hat{x}_i = \alpha + \frac{1}{2}\omega_i + \sum_{j=1}^{i-1} \omega_j \ . \tag{6.11}$$

PROOF. The partial derivative of (6.8) with respect to $\hat{x}_i$ yields

$$\frac{\partial G^*(\underline{\eta})}{\partial \hat{x}_i} = -\omega_i \int_\Omega \left(\tilde{F}(x) - F(x;\underline{\eta})\right) \cdot \delta(x - \hat{x}_i)\,\mathrm{d}x \ . \tag{6.12}$$

Applying the necessary condition of a minimum $\partial G^*(\underline{\eta})/\partial \hat{x}_i = 0$ on (6.12) leads to

$$\int_\Omega \tilde{F}(x) \cdot \delta(x - \hat{x}_i)\,\mathrm{d}x = \int_\Omega F(x;\underline{\eta}) \cdot \delta(x - \hat{x}_i)\,\mathrm{d}x \ .$$

Employing the sifting property of the Dirac delta distribution finally results in (6.11). □

PROOF. [**of Theorem 6.1**] Without loss of generality it is assumed that

$$\alpha \leq \hat{x}_1 < \hat{x}_2 < \cdots < \hat{x}_L \leq \beta \ .$$

The partial derivative of (6.7) with respect to $\hat{x}_i$, $i = 1, \ldots, L$ under incorporation of the necessary condition $\partial G(\underline{\eta})/\partial \hat{x}_i = 0$ for a minimum leads to

$$\int_{\mathbb{R}} \int_\Omega \left(\tilde{F}(y|x) - F(y,x;\underline{\eta})\right) \cdot \frac{\partial F(y,x;\underline{\eta})}{\partial \hat{x}_i}\,\mathrm{d}x\,\mathrm{d}y = 0 \ ,$$

with

$$\frac{\partial F(y,x;\underline{\eta})}{\partial \hat{x}_i} = -\frac{1}{2}\omega_i \cdot \delta(x - \hat{x}_i) \cdot \left(1 + \mathrm{erf}_i(y)\right) \ .$$

Utilizing the sifting property of the Dirac delta distribution yields

$$\int_{\mathbb{R}} \tilde{F}(y|\hat{x}_i)\left(1 + \mathrm{erf}_i(y)\right)\mathrm{d}y = \int_{\mathbb{R}} F(y,\hat{x}_i;\underline{\eta})\left(1 + \mathrm{erf}_i(y)\right)\mathrm{d}y \ .$$

To allow further simplifications, a nonlinear shear is applied by setting

$$\psi(\hat{x}_i) = K \ , \ K \in \mathbb{R} \text{ (constant)} \ . \tag{6.13}$$

This changes only the position of the probability mass along dimension y. The total probability mass and the total marginal probability mass of $\boldsymbol{y}$ remain unchanged. Thus, we get

$$\frac{1}{2}\int_{\mathbb{R}} (\hat{x}_i - \alpha) \cdot \left(1 + \mathrm{erf}\left(\frac{y - K}{\sqrt{2}\sigma}\right)\right) \cdot \left(1 + \mathrm{erf}_i(y)\right)\mathrm{d}y = \int_{\mathbb{R}} F(y,\hat{x}_i;\underline{\eta}) \cdot \left(1 + \mathrm{erf}_i(y)\right)\mathrm{d}y \ .$$

Resubstituting (6.13) and comparing coefficients leads to

$$\hat{x}_i = \alpha + \frac{1}{2}\omega_i + \sum_{j=1}^{i-1}\omega_j \ , \qquad \hat{y}_i = \psi(\hat{x}_i) \ , \qquad \sigma_i^y = \sigma \ .$$

The solution for $\hat{x}_i$ coincides with the result of Lemma 6.1. Thus, minimizing (6.8) is sufficient for obtaining ω_i and $\hat{x}_i$. In consideration of (6.9) und (6.10) it is obvious that $\tilde{F}(x)$ represents the distribution function of an unnormalized uniform distribution over Ω. The optimal approximation of such a distribution by means of Dirac and Heaviside mixtures is well-known [162] and thus, the remaining free parameters ω_i and $\hat{x}_i$ are given by

$$\omega_i = \frac{\beta - \alpha}{L} \ , \qquad \hat{x}_i = \alpha + \omega_i \cdot \frac{(2i-1)}{2} \ .$$

□

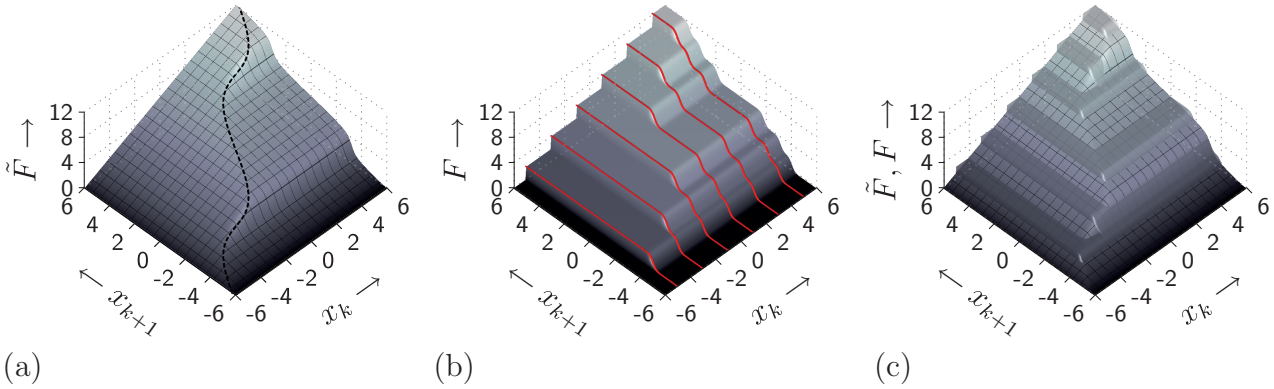


Figure 6.3: Approximation of the conditional distribution function. (a) Conditional distribution function for the system model $\boldsymbol{x}_{k+1} = \sin(\boldsymbol{x}_k) + \boldsymbol{x}_k + \boldsymbol{w}_k$. The black dashed line indicates the system function. (b) Hybrid distribution function with $L = 6$ components. The red lines indicate the optimally placed hybrid components. (c) Superposition of the true and the approximate conditional distribution function.

Summarizing the result of Theorem 6.1, optimally approximating the conditional density is merely a uniform placement of the Dirac delta distributions of the hybrid density within the interval Ω. Thus, the probability masses enclosed between two adjacent components (slices) of the hybrid density are all equal. Furthermore, the Gaussian elements $\mathcal{N}(y; \hat{y}_i, (\sigma_i^y)^2)$ of the hybrid density are displaced duplicates of the noise density $f_k^w(w_k)$ or $f_k^v(v_k)$ that are placed along the nonlinear functions $a_k(\hat{x}_i)$ or $h_k(\hat{x}_i)$, respectively. Hence, each slice of the conditional density illustrated in Figure 6.2 represents a noise density.

For increasing y, the true conditional distribution $\tilde{F}(y|x)$ approaches an unnormalized uniform distribution over Ω. Hence, the approximation error between the true conditional distribution and its hybrid approximation $F(y, x; \underline{\eta})$ is dominated by the deviation between the uniform distribution and its approximation given by the Heaviside step functions, which leads to the uniform placement of the Dirac delta distributions in Ω. This finding is illustrated in the following example.

Example 6.2: Hybrid Distribution

Consider again the nonlinear system model of Example 6.1 with $\boldsymbol{x}_k \in \Omega_k = [-6, 6]$. The corresponding distribution function

$$\tilde{F}(x_{k+1}|x_k) = \frac{1}{2} \int_{-6}^{x_k} 1 + \operatorname{erf}\left(\frac{x_{k+1} - \sin(s) - s}{\sqrt{2}\sigma_k^w} \right) \mathrm{d}s$$

is depicted in Figure 6.3 (a). For $x_{k+1} = 6$, the distribution function $\tilde{F}(x_{k+1}|x_k)$ corresponds to an unnormalized uniform distribution over Ω_k. Figure 6.3 (b) illustrates the approximate hybrid conditional distribution for $L = 6$ components, where the optimal parameters are $\omega_i = 2$, $\sigma_i^y = 1$, $\hat{x}_i = \{-5, -3, -1, 1, 3, 5\}$, and $\hat{y}_i = \{-4.04, -3.14, -1.84, 1.84, 3.14, 4.04\}$, for $i = 1, \ldots, 6$. By superimposing the true and the approximate conditional distribution in Figure 6.3 (c), the similarity between both distributions is revealed. ■

In consideration of the result of Theorem 6.1 and the illustrations of Figure 6.3, it is obvious that the hybrid density converges towards the true conditional density function for an increasing number of components L.

6.1.4 Generalization

Typically, a parametric structure is used for representing the noise, which allows directly setting σ_i^y of $\mathcal{N}(y; \hat{y}_i, (\sigma_i^y)^2)$ to the corresponding parameter of the noise density. This can be generalized

to other parametric noise density representations like Gaussian mixtures, exponential densities [26] or Edgeworth series [36]. There, the non-Dirac mixture density type of the hybrid density has to be chosen according to the current noise density representation [215].

If a non-parametric noise density is available or the density type of the noise differs from the desired type for representing the system state's density, it is possible to first find an appropriate noise density approximation, for example using the method described in [74] for non-parametric noise or the method described in [71] for differing parametric noise, and then to approximate the conditional density afterwards.

6.2 State Estimation

Due to the simplicity of calculating the optimal hybrid density, the conditional density approximation can be performed on-line, i.e., at every time step k. This allows deriving an approximate but efficient computation of the prediction and measurement update step of the Bayesian state estimator. Both steps of the HDF are presented in the following.

6.2.1 HDF Prediction Step

The special structure of the hybrid transition density approximation is very convenient for efficiently performing the prediction step, since it allows calculating a closed-form solution of the Chapman-Kolmogorov equation (2.7).

Theorem 6.2 (Approximate Predicted Density) *Given the density $f_k^x(x_k)$ of the current system state $\boldsymbol{x}_k$ and the hybrid transition density approximation*

$$f(x_{k+1}, x_k; \underline{\eta}) = \sum_{i=1}^{L} \omega_i \cdot \delta(x_k - \hat{x}_i) \cdot \mathcal{N}(x_{k+1}; \hat{y}_i, (\sigma_i^y)^2) \ , \tag{6.14}$$

with parameter vector $\underline{\eta}$ according to Theorem 6.1, the approximate predicted density $f_{k+1}^p(x_{k+1})$ is a Gaussian mixture density with L components that can be calculated analytically.

Proof. Replacing $f_k^T(x_{k+1}|x_k)$ in (2.7) by its approximation (6.14) yields

$$\begin{aligned} f_{k+1}^p(x_{k+1}) &= \int_{\mathbb{R}} f(x_{k+1}, x_k; \underline{\eta}) \cdot f_k^x(x_k) \, \mathrm{d}x_k \\ &= \sum_{i=1}^{L} \omega_i \cdot \mathcal{N}(x_{k+1}; \hat{y}_i, (\sigma_i^y)^2) \cdot \underbrace{\int_{\mathbb{R}} f_k^x(x_k) \, \delta(x_k - \hat{x}_i) \, \mathrm{d}x_k}_{= f_k^x(\hat{x}_i)} \\ &= \sum_{i=1}^{L} \omega_{k+1,i} \cdot \mathcal{N}(x_{k+1}; \hat{y}_i, (\sigma_i^y)^2) \ , \end{aligned} \tag{6.15}$$

with $\omega_{k+1,i} = \omega_i \cdot f_k^x(\hat{x}_i)$. For $i = 1, \ldots, L$, the weighting coefficients ω_i of the hybrid transition density have the same constant value, where the value of ω_i has no impact on the prediction. Thus, setting $\omega_i = 1/\sum_{i=1}^{L} f_k^x(\hat{x}_i)$ leads to a normalized predicted density. □

In a Bayesian setting according to Figure 2.2 and Figure 6.1, the prior $f_k^x(x_k)$ can be any continuous density. Thanks to the point-wise evaluation of the prior density by means of the Dirac delta distributions of the hybrid density, the HDF prediction step is able to handle any continuous density. In case of a Gaussian mixture prior density, the HDF prediction step preserves the density type since the predicted density functions is also Gaussian mixture.

Furthermore, the complexity of the predicted density remains at a constant level, since the number of components representing $f_{k+1}^{p}(x_{k+1})$ only depends on the number L of components of the hybrid density. This systematically prevents an exponential growth of the number of mixture components as it occurs for example for the Gaussian mixture estimator in Section 5.5. Hence, Gaussian mixture reduction is not required in principle. However, due to the point-wise evaluation of the prior density, performing the HDF prediction step is in $O(L^2)$ assuming that $f_k^x(x_k)$ is represented by L components. For large L, it is reasonable to perform Gaussian mixture reduction prior to the prediction step for an overall reduced computational complexity, especially in case of multivariate states (see Section 6.3).

6.2.2 HDF Measurement Update

Performing measurement updates via the HDF differs in two aspects from the prediction step, as illustrated in Figure 6.1. First, instead of directly incorporating the hybrid conditional density, an approximate likelihood has to be generated. Second, the posterior density is not a Gaussian mixture due to the Dirac mixture representation of the likelihood. Thus, an additional interpolation step is needed in order to preserve a continuous density representation. However, the number of components of the approximate density remains constant over all additional steps required for measurement updating and depends, analogously to the HDF prediction step, merely on the number of components used for approximating the conditional density function.

Likelihood Generation and Measurement Updating

For a given measurement $\hat{z}_k$ at time step k, generating the likelihood is straightforward. Plugging $\hat{z}_k$ into the hybrid conditional density approximation $f(z_k, x_k; \underline{\eta})$ yields the likelihood approximation

$$f_k^L(\hat{z}_k|x_k) = f(z_k, x_k; \underline{\eta})\Big|_{z_k=\hat{z}_k} = \sum_{i=1}^{L} \underbrace{\omega_i \cdot \mathcal{N}(\hat{z}_k; \hat{y}_i, (\sigma_i^y)^2)}_{=:\omega_{k,i}^x} \cdot \delta(x_k - \hat{x}_i)$$

$$= \sum_{i=1}^{L} \omega_{k,i}^x \cdot \delta(x_k - \hat{x}_i) \ , \tag{6.16}$$

with $\underline{\eta}_k = [\underline{\eta}_{k,1}^{\mathrm{T}}, \underline{\eta}_{k,2}^{\mathrm{T}}, \ldots, \underline{\eta}_{k,L}^{\mathrm{T}}]^{\mathrm{T}}$, where $\underline{\eta}_{k,i}^{\mathrm{T}} = [\omega_{k,i}^x, \hat{x}_i]$. Thus, the likelihood is represented by a Dirac mixture that in subsequent processing steps is very convenient to efficiently perform the measurement update.

Theorem 6.3 (Dirac Mixture Posterior Density) *Given the predicted density $f_k^p(x_k)$ of the current system state $\boldsymbol{x}_k$ and the approximate likelihood* (6.16)*, the posterior density $\bar{f}_k^e(x_k)$ is a Dirac mixture.*

Proof. Employing Bayes' law (2.9) yields

$$\bar{f}_k^e(x_k) = c_k \cdot f_k^p(x_k) \cdot f_k^L(\hat{z}_k|x_k) = c_k \cdot \sum_{i=1}^{L} \omega_{k,i}^x \cdot \underbrace{\delta(x_k - \hat{x}_i) \cdot f_k^p(x_k)}_{=f_k^p(\hat{x}_i) \cdot \delta(x_k - \hat{x}_i)}$$

$$= c_k \cdot \sum_{i=1}^{L} \omega_{k,i} \cdot \delta(x_k - \hat{x}_i) \ , \tag{6.17}$$

where $\omega_{k,i} = \omega_{k,i}^x \cdot f_k^p(\hat{x}_i)$. The normalization constant $c_k = 1/\sum_{i=1}^{L} \omega_{k,i}$ results from integrating over the sum in (6.17). □

Directly applying this Dirac mixture posterior density to the prediction step would lead to a degeneration of the predicted density, since $\bar{f}_k^e(x_k)$ would be multiplied with the Dirac delta distributions of the hybrid transition density. Instead, the Dirac delta distributions of $\bar{f}_k^e(x_k)$ can be interpolated with arbitrary functions since the point-wise evaluation property of the HDF prediction step allows processing any continuous density representation.

Interpolation Step

In the following, Gaussians are used for interpolation. This leads to a *Gaussian mixture posterior density*

$$f_k^e(x_k) = \sum_{i=1}^{L} \omega_{k,i} \cdot \mathcal{N}(x_k; \hat{x}_i, (\sigma_i^x)^2) \tag{6.18}$$

of $\bar{f}_k^e(x_k)$. This density type representation coincides with the result of the prediction step, i.e., the HDF remains within the same function class for both predictions and measurement updates.

While the parameters $\omega_{k,i}$ and $\hat{x}_i$ in (6.18) can be directly adopted from $\bar{f}_k^e(x_k)$, appropriate standard deviations σ_i^x have to be determined. In general, such an interpolation is computationally demanding. Keeping in mind that the conditional density $\tilde{f}(z_k|x_k)$ has the uniform distribution property when marginalizing along z_k (see Lemma 6.1), the required computational load can be drastically reduced and a suboptimal interpolation can be performed. It is assumed that all Gaussian components in (6.18) have the same standard deviation $\sigma_i^x = \sigma^x$, $i \in \{1, \ldots, L\}$. Then, the interpolation problem can be reduced to the one-dimensional optimization problem

$$\sigma^x = \arg \min_{\sigma} \bar{G}(\sigma) \ , \tag{6.19}$$

where

$$\bar{G}(\sigma) = \int_{\Omega_k} \left(1 - f_k^e(x_k)\right)^2 \mathrm{d}x_k \ .$$

Here, $f_k^e(x_k)$ has to optimally fit with an unnormalized uniform distribution on Ω_k. This optimization problem is convex and thus, σ^x can be derived by means of, e.g. gradient descent. The required computational load can be further reduced by employing the approximate solution [22]

$$\sigma^x = \frac{\beta_k - \alpha_k}{\sqrt{2}L} \tag{6.20}$$

that rapidly convergences to the correct solution of (6.19) for increasing L.

Remark 6.1 (Posterior Mean) The mean of the Dirac mixture posterior density and the Gaussian mixture posterior density are equivalent, as calculating the mean of $f_k^e(x_k)$ is merely based on $\omega_{k,i}$ and $\hat{x}_i$.

The interpolation step can also be considered as a convolution of the Dirac mixture posterior density (6.17) with the zero-mean Gaussian $\mathcal{N}(x_k; 0, (\sigma^x)^2)$ with standard deviation according to (6.20). Thanks to the equally spaced placement of the Dirac delta distributions in $\bar{f}_k^e(x_k)$, this convolution always leads to a smooth density function representation, independent of the number of components L. This is demonstrated in the following example.

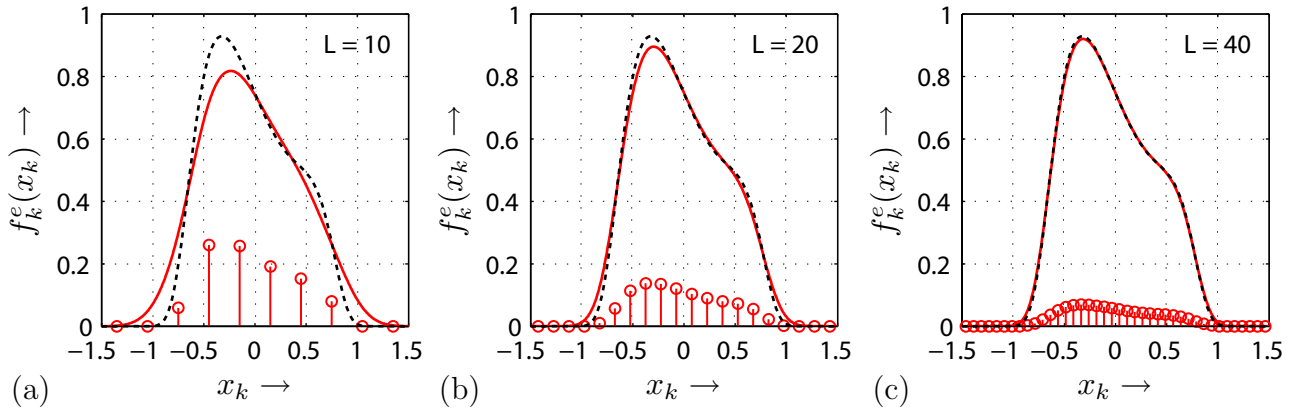


Figure 6.4: Gaussian mixture posterior densities $f_k^e(x_k)$ (red solid lines) and Dirac mixture posterior densities $\bar{f}_k^e(x_k)$ (red stems) resulting from a HDF measurement update with (a) $L = 10$, (b) $L = 20$, and (c) $L = 40$ components representing the approximate conditional density. The true posterior density (black dashed lines) is calculated by means of numerical integration.

Example 6.3: Smoothing via the Interpolation Step

In this example the cubic sensor problem [28] is considered, where the sensor model is given by

$$\boldsymbol{z}_k = \boldsymbol{x}_k^3 + \boldsymbol{v}_k \ . \tag{6.21}$$

The noise is represented by the Gaussian $\boldsymbol{v}_k \sim \mathcal{N}(v_k; 0, 0.3^2)$ and the current state estimate is represented by the Gaussian $\boldsymbol{x}_k \sim \mathcal{N}(x_k; -1.5, 1.2^2)$. Performing the HDF measurement update step with hybrid conditional density approximations with $L \in \{10, 20, 40\}$ components and a given measurement value $\hat{z}_k = 0.2$ leads to the posterior densities depicted in Figure 6.4 (a)-(c). It can be seen that for an increasing number of components L, the approximate Gaussian mixture posterior approaches the true one. For a very small number of components, i.e., for $L = 10$, the Gaussian mixture posterior density is a coarse but smooth approximation, while the deviation between the true posterior and its approximation is marginal for $L = 40$. ■

6.2.3 Combined Prediction and Measurement Update

An alternative to separately performing prediction and measurement update steps is to combine both steps into a single one. For this purpose, the Bayes' formula (2.9) is plugged into the Chapman-Kolmogorov integral (2.7), which yields

$$f_{k+1}^p(x_{k+1}) = c_k \int_{\mathbb{R}} f_k^T(x_{k+1}|x_k) \cdot f_k^L(\hat{z}_k|x_k) \cdot f_k^p(x_k) \, \mathrm{d}x_k \ .$$

Here, it is sufficient to approximate the transition density only and to perform the prediction step as described in Section 6.2.1, while the likelihood can be utilized directly without any approximation. In doing so, the given measurement value $\hat{z}_k$ is automatically incorporated into the state estimate. Furthermore, an additional interpolation step is not required, since the HDF prediction step results in a Gaussian mixture representation for $f_{k+1}^p(x_{k+1})$ (see (6.15)). However, by performing a combined prediction and measurement update it is impossible to achieve a separate posterior density $f_k^e(x_k)$, which is not always preferable, especially in cases where the measurement rate is higher than the time update rate.

6.2.4 Simulation Example

To investigate the performance of the HDF, the nonlinear system model (6.4) introduced in Example 6.1 as well as the cubic sensor model (6.21) of Example 6.3 are considered. A total of 50

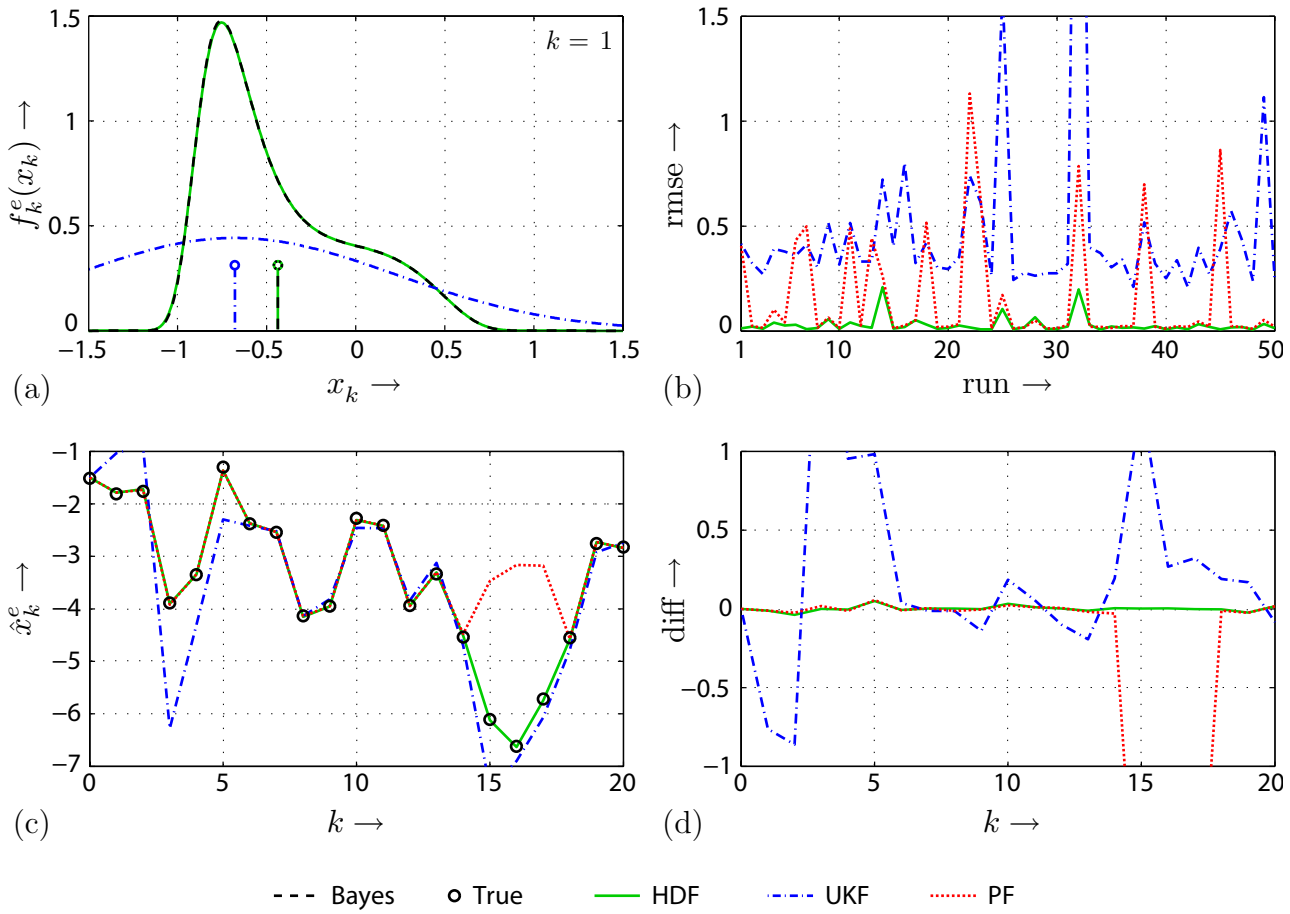


Figure 6.5: (a) Posterior density and mean estimates (stems) after the first measurement update. (b) Root mean square error over 50 Monte Carlo simulation runs. (c) True means (black circles) as well as the mean estimates of the HDF (green solid line), the UKF (blue, dash-dotted), and the PF (red, dotted) for an exemplary simulation run. (d) Difference of the three estimators to the ground truth for the exemplary run.

Monte Carlo simulations are performed with $\sigma_k^w = 0.8$, $\sigma_k^v = 0.3$, $L = 75$, and an initial Gaussian density $f_0^x(x_0) = \mathcal{N}(x_0; -1.5, 1.2^2)$. Each of the 50 Monte Carlo simulation runs consists of 20 alternating prediction and measurement update steps, commencing with a prediction. For comparison, the results of the unscented Kalman filter (UKF) and a particle filter employing 300 particles and systematic resampling (PF) are considered. For an introduction to systematic resampling see [30].

In Figure 6.5 (a), an exemplary result of the first measurement update is depicted.[1] Computationally demanding numerical integration is applied in order to obtain the exact density as reference. It is obvious that there is almost no shape difference between the exact posterior density and the Gaussian mixture density approximation resulting from the HDF. As demonstrated in Example 6.3, the interpolation step of the HDF measurement update can provide an accurate approximation of the true posterior density. In contrast, the Gaussian assumption of the UKF results in a significant difference in shape. This also appears for the mean estimate, while the HDF provides a mean estimate that is almost identical to the ground truth. Also higher-order moments cannot be tracked that accurate by the UKF. In contrast, the shape approximation provided by the HDF allows capturing higher-order moments.

Focusing on the mean estimates, Figure 6.5 (b) depicts the root mean square error (rmse) for each of the 50 simulations runs. The rmse of the HDF is the lowest in 45 simulation runs.

1 The PF provides just a particle representation of the density and thus is omitted here.

For the remaining 5 runs the PF performs best. Consequently, the average rmse of the HDF over all simulation runs is the lowest and is one order of magnitude lower as the average rmse of the PF, although the number of particles of the PF is four times larger than the number of components of the HDF, which also leads to a larger computation time of the PF. Increasing the number of particles of the PF does not improve the estimation accuracy significantly.

Figure 6.5 (c) exemplarily shows the mean estimates of all estimators for one of the 50 Monte Carlo runs, while in Figure 6.5 (d) the mean estimation difference of all estimators to the ground truth is illustrated for this particular run. The relatively poor estimation results of the PF follows from the potential multimodal nature of the predicted density. So, the PF sometimes tracks the wrong modes and the mean estimates strongly differ from the true means. Unlike the PF, the HDF shows almost no deviation from the true means.

6.3 Extension to Multivariate States

So far, the HDF was examined for scalar states. This section is now concerned with an extension to multivariate states together with some hints for a more computationally efficient application of the HDF. For this purpose the nonlinear system and sensor models

$$\begin{aligned} \underline{\boldsymbol{x}}_{k+1} &= \underline{a}_k(\underline{\boldsymbol{x}}_k) + \underline{\boldsymbol{w}}_k \ , \\ \underline{\boldsymbol{z}}_k &= \underline{h}_k(\underline{\boldsymbol{x}}_k) + \underline{\boldsymbol{v}}_k \end{aligned}$$

affected by additive noise are considered, where $\underline{\boldsymbol{w}}_k \sim \mathcal{N}(\underline{w}_k; \hat{\underline{w}}_k, \mathbf{C}_k^w)$ and $\underline{\boldsymbol{v}}_k \sim \mathcal{N}(\underline{v}_k; \hat{\underline{v}}_k, \mathbf{C}_k^v)$.

6.3.1 Suboptimal Conditional Density Approximation

In case of multivariate states, the conditional density

$$\tilde{f}(\underline{y}|\underline{x}) = \mathcal{N}(\underline{y}; \psi(\underline{x}), \mathbf{C})$$

representing the system model or sensor model needs to be approximated by the hybrid density

$$f(\underline{y}, \underline{x}; \underline{\eta}) = \sum_{i=1}^{L} \omega_i \cdot \delta(\underline{x} - \hat{\underline{x}}_i) \cdot \mathcal{N}(\underline{y}; \hat{\underline{y}}_i, \mathbf{C}_i) \ . \tag{6.22}$$

To adjust the parameter vector $\underline{\eta}$ in (6.22), which comprises the weights ω_i, the means $\hat{\underline{x}}_i$ of the Dirac delta distributions, the means $\hat{\underline{y}}_i$ of the Gaussians, and the covariance matrices $\mathbf{C}_i$ of the Gaussians, it is possible to formulate an optimization problem equivalent to the scalar case. Again, the Cramér-von Mises distance (6.7) between the true and the approximate cumulative distribution function is employed. But in contrast to the scalar case, this optimization problem cannot be solved in closed form. While optimally approximating the unnormalized uniform distribution representing the marginal cumulative distribution function $\tilde{F}(x)$ (see (6.9)) by means of the Heaviside step functions is straightforward for the scalar case [162], no closed-form optimal approximation can be found for the corresponding multivariate uniform distribution $\tilde{F}(\underline{x})$.

Instead of determining the optimal approximation of the multivariate uniform distribution by means of numerical optimization, which may be computational demanding or even intractable, a suboptimal approximation is used instead. For this purpose, the mean vectors $\hat{\underline{x}}_i$ of the Dirac delta distributions are arranged in a regular grid. More specifically, assuming that $\underline{\boldsymbol{x}} \in \underline{\Omega}$, i.e., $\underline{\boldsymbol{x}}$ is restricted to its support

$$\underline{\Omega} = [\alpha_1, \beta_1] \times [\alpha_2, \beta_2] \times \cdots \times [\alpha_{n_x}, \beta_{n_x}] \subset \mathbb{R}^{n_x} \ ,$$

the considered multivariate (unnormalized) uniform distribution can be written as

$$\tilde{F}(\underline{x}) = \prod_{n=1}^{n_x} \tilde{F}(x_n) = \prod_{n=1}^{n_x} (x_n - \alpha_n) \ . \tag{6.23}$$

The optimal approximation for each scalar (unnormalized) uniform distribution $\tilde{F}(x_n)$ in (6.23) on the interval $[\alpha_n, \beta_n]$, $n \in \{1, 2, \ldots, n_x\}$, is determined first. Calculating the cross product of all n_x scalar approximations yields to suboptimal grid approximation of (6.23), which is given by

$$\begin{aligned} \omega_i &= \prod_{n=1}^{n_x} \bar{\omega}_n \ , \\ \underline{\hat{x}}_i &= [\alpha_1, \alpha_2, \ldots, \alpha_{n_x}]^{\mathrm{T}} + \left[\bar{\omega}_1 \cdot \tfrac{2 \cdot j_1 - 1}{2}, \ldots, \bar{\omega}_{n_x} \cdot \tfrac{2 \cdot j_{n_x} - 1}{2}\right]^{\mathrm{T}} \ , \end{aligned}$$

with the weighting coefficient for dimension n

$$\bar{\omega}_n = \frac{\beta_n - \alpha_n}{L_n} \ .$$

The index of a component arises from $i = (1 + \sum_{n=1}^{n_x} (j_n - 1) \cdot L_n^{n-1}) \in \{1, 2, \ldots L\}$, where j_n indicates the j-th Dirac component of dimension n. Furthermore, L_n is the number of Dirac components used for approximating the scalar uniform distribution $\tilde{F}(x_n)$ of dimension n. Hence, the total number of components L of the hybrid density (6.22) is $L = \prod_n L_n$. As discussed in [106], this kind of arrangement provides a locally optimal approximation of a multivariate uniform distribution over $\underline{\Omega}$ with respect to the Cramér-von Mises distance measure.

The remaining parameters $\underline{\hat{y}}_i$ and $\mathbf{C}_i$ can be determined analogously to the scalar case. Hence, the Gaussian elements of (6.22) are duplicates of the noise density that are placed along the nonlinear function $\psi(\,\cdot\,)$, i.e., $\underline{\hat{y}}_i = \psi\left(\underline{\hat{x}}_i\right)$ and the covariances $\mathbf{C}_i$ are identical to the corresponding noise covariances $\mathbf{C}_k^w$ and $\mathbf{C}_k^v$, respectively.

By means of the hybrid conditional density (6.22), the prediction and measurement update step coincide to the scalar case. Thanks to the grid arrangement of the Dirac delta distributions, the interpolation step required for measurement updates is also be performed in a straightforward fashion. Assuming that the covariance matrix $\mathbf{C}$ of each component of the Gaussian mixture posterior density

$$f_k^e(\underline{x}_k) = \sum_{i=1}^{L} \omega_{k,i} \cdot \mathcal{N}(\underline{x}_k; \underline{\hat{x}}_i, \mathbf{C}) \ ,$$

is diagonal, i.e., $\mathbf{C} = \mathrm{diag}([\sigma_1^2, \sigma_2^2, \ldots, \sigma_{n_x}^2])$, the variances σ_n^2, $n \in \{1, 2, \ldots, n_x\}$, can be calculated independently by solving the optimization problem (6.19) dimension by dimension.

6.3.2 Improvements in Efficiency

Although the previously introduced extension of the HDF for multivariate state is straightforward and thus easy to implement, it becomes computationally demanding for large state spaces. This is due to the fact that the Dirac delta distributions of the hybrid density are arranged in a grid. The size of the grid grows exponentially with the dimension of the state space. In the following, two techniques are presented, which attenuate this exponential growth by decreasing the number of required Dirac delta distributions per time step. This allows applying the HDF even for high-dimensional states.

Unobserved State Vector

In many applications, e.g., in target tracking scenarios, the sensor model often merely depends on parts of the state vector. Here, the state vector can be decomposed into two substate vectors

$$\underline{\boldsymbol{x}}_k = [(\underline{\boldsymbol{x}}_k^o)^{\mathrm{T}}, (\underline{\boldsymbol{x}}_k^u)^{\mathrm{T}}]^{\mathrm{T}} \ ,$$

where the *observed state vector* $\underline{\boldsymbol{x}}_k^o$ represents the elements of the state vector that occur in the sensor model, while all other elements are subsumed in the *unobserved state vector* $\underline{\boldsymbol{x}}_k^u$.

Example 6.4: Unobserved State Vector in Target Tracking
The system state $\underline{\boldsymbol{x}}_k = [\boldsymbol{x}_k, \dot{\boldsymbol{x}}_k, \boldsymbol{y}_k, \dot{\boldsymbol{y}}_k]^{\mathrm{T}}$ of the target tracking problem considered in Example 2.1 can be decomposed into

$$\begin{aligned} &\text{(position)} && \underline{\boldsymbol{x}}_k^o = [\boldsymbol{x}_k, \boldsymbol{y}_k]^{\mathrm{T}} \ , \\ &\text{(velocity)} && \underline{\boldsymbol{x}}_k^u = [\dot{\boldsymbol{x}}_k, \dot{\boldsymbol{y}}_k]^{\mathrm{T}} \ , \end{aligned}$$

if distance sensors (2.4) or bearing sensors (3.15) are used. For instance, for distance measurements (relative to the origin), the corresponding likelihood $f_k^L(\hat{z}_k|\underline{x}_k) = f_k^L(\hat{z}_k|\underline{x}_k^o) = \mathcal{N}(\hat{z}_k - \sqrt{\boldsymbol{x}_k^2 + \boldsymbol{y}_k^2}; \hat{v}_k, (\sigma_k^v)^2)$ only depends on the observed state vector $\underline{\boldsymbol{x}}_k^o$. ■

The computational demand of the HDF can be significantly reduced if only the observed state vector is considered during the measurement update step. The update of the unobserved part occurs indirectly by using the stochastic dependencies between the observed and unobserved states introduced in the system model. According to this, only the observed state vector has to be considered for approximating the conditional density and likelihood, respectively. Thus, it is not necessary to represent the whole state vector by means of a grid. This finding is summarized in the following theorem.

Theorem 6.4 (Measurement Update for Observed States) *The HDF measurement update for the observed state vector is independent of the unobserved state vector.*

Proof. For the Gaussian mixture prior density $f_k^p(\underline{x}_k)$ representing the current estimate of the state vector $\underline{\boldsymbol{x}}_k$, the mean and covariance of a Gaussian component $\mathcal{N}(\underline{x}_k; \hat{\underline{x}}_k, \mathbf{C}_k)$ of the Gaussian mixture $f_k^p(\underline{x}_k)$ can be written as

$$\hat{\underline{x}}_k = [(\hat{\underline{x}}_k^o)^{\mathrm{T}}, (\hat{\underline{x}}_k^u)^{\mathrm{T}}]^{\mathrm{T}} \ , \ \mathbf{C}_k = \begin{bmatrix} \mathbf{C}_k^o & \mathbf{C}_k^{ou} \\ \mathbf{C}_k^{uo} & \mathbf{C}_k^u \end{bmatrix} \ .$$

Applying Bayes' law to a Gaussian components yields [150]

$$\mathcal{N}(\underline{x}_k; \hat{\underline{x}}_k, \mathbf{C}_k) = \mathcal{N}(\underline{x}_k^{u|o}; \hat{\underline{x}}_k^{u|o}, \mathbf{C}_k^{u|o}) \cdot \mathcal{N}(\underline{x}_k^o; \hat{\underline{x}}_k^o, \mathbf{C}_k^o) \ , \tag{6.24}$$

where

$$\begin{aligned} \hat{\underline{x}}_k^{u|o} &= \hat{\underline{x}}_k^u + \mathbf{C}_k^{uo} \, (\mathbf{C}_k^o)^{-1} \cdot (\underline{x}_k^o - \hat{\underline{x}}_k^o) \ , \\ \mathbf{C}_k^{u|o} &= \mathbf{C}_k^u - \mathbf{C}_k^{uo} \, (\mathbf{C}_k^o)^{-1} \, \mathbf{C}_k^{ou} \ . \end{aligned}$$

Since only the observed state vector is considered for measurement update, the true likelihood $\tilde{f}_k^L(\hat{\underline{z}}_k|\underline{x}_k^o)$ can be approximated by

$$f_k^L(\hat{\underline{z}}_k|\underline{x}_k^o) = \sum_{i=1}^{L} \omega_{k,i} \cdot \delta(\underline{x}_k^o - \hat{\underline{x}}_i^o) \tag{6.25}$$

according to (6.16), where $\hat{\underline{x}}_i^o$ indicates the position of the i-th Dirac component used for approximating the observed state. With (6.24), (6.25), and Bayes' law (2.9), the Dirac mixture posterior density is given by

$$\begin{aligned}\bar{f}_k^e(\underline{x}_k) &= c_k \cdot f_k^L(\hat{\underline{z}}_k|\underline{x}_k^o) \cdot f_k^p(\underline{x}_k) \\ &= c_k \cdot \left(\sum_{i=1}^{L} \omega_{k,i} \cdot \delta(\underline{x}_k^o - \hat{\underline{x}}_i^o)\right) \cdot \left(\sum_{j=1}^{M} \omega_{k,j} \cdot \mathcal{N}\left(\underline{x}_k^{u|o}; \hat{\underline{x}}_{k,j}^{u|o}, \mathbf{C}_{k,j}^{u|o}\right) \cdot \mathcal{N}\left(\underline{x}_k^o; \hat{\underline{x}}_{k,j}^o; \mathbf{C}_{k,j}^o\right)\right) \\ &= c_k \cdot \sum_{i=1}^{L}\sum_{j=1}^{M} \omega_{k,i,j} \cdot \delta(\underline{x}_k^o - \hat{\underline{x}}_i^o) \cdot \mathcal{N}\left(\underline{x}_k^{u|o}; \hat{\underline{x}}_{k,i,j}^{u|o}, \mathbf{C}_{k,j}^{u|o}\right) \end{aligned} \tag{6.26}$$

with

$$\begin{aligned}\omega_{k,i,j} &= \omega_{k,i} \cdot \omega_{k,j} \cdot \mathcal{N}\left(\hat{\underline{x}}_i^o; \hat{\underline{x}}_{k,j}^o, \mathbf{C}_{k,j}^o\right) \\ \hat{\underline{x}}_{k,i,j}^{u|o} &= \hat{\underline{x}}_{k,j}^u + \mathbf{C}_k^{uo} \left(\mathbf{C}_k^o\right)^{-1} \cdot \left(\hat{\underline{x}}_i^o - \hat{\underline{x}}_{k,j}^o\right) \ . \end{aligned} \tag{6.27}$$

In order to obtain the Gaussian mixture posterior density $f_k^e(\underline{x}_k)$, the interpolation step can be applied on the Dirac delta distributions of (6.26). □

Thanks to the way of determining the Gaussian mixtures by the HDF prediction and interpolation step, the covariance matrices $\mathbf{C}_{k,j}^{u|o}$ of (6.26) are identical for all Gaussian components. Thus, they need to be calculated only once per time step, which is computationally negligible. The update of the unobserved state vector and thus, the utilization of stochastic dependencies is carried out in (6.27) by updating the weighting coefficients.

The benefit of approximating the conditional density and the likelihood only under consideration of the observed state vector comes at the expense of an exponential growth of the number of components in the posterior density (see the two sums in (6.26)). However, performing a HDF prediction step automatically leads to a reduction of the posterior Gaussian mixture, since the complexity of the predicted density only depends on the number of components of the hybrid transition density. In cases of repeated measurement updates, a separate Gaussian mixture reduction algorithm as the one proposed in Chapter 7 has to be performed to keep the exponential growth bounded.

Rao-Blackwellization

The computational demand can be further reduced for system and sensor models of the type

$$\begin{aligned}\underline{\boldsymbol{x}}_{k+1}^n &= \mathbf{A}_k^n \cdot \underline{\boldsymbol{x}}_k^l + \underline{a}_k(\underline{\boldsymbol{x}}_k^n) + \underline{\boldsymbol{w}}_k^n \ , \\ \underline{\boldsymbol{x}}_{k+1}^l &= \mathbf{A}_k^l(\underline{\boldsymbol{x}}_k^n) \cdot \underline{\boldsymbol{x}}_k^l + \underline{\boldsymbol{w}}_k^l \ , \\ \hat{\underline{z}}_k &= \underline{h}_k(\underline{\boldsymbol{x}}_k^n) + \mathbf{H}_k(\underline{\boldsymbol{x}}_k^n) \cdot \underline{\boldsymbol{x}}_k^l + \underline{\boldsymbol{v}}_k \ , \end{aligned}$$

where the state vector and the system noise are composed according to

$$\begin{aligned}\underline{\boldsymbol{x}}_k &= [(\underline{\boldsymbol{x}}_k^l)^{\mathrm{T}}, (\underline{\boldsymbol{x}}_k^n)^{\mathrm{T}}]^{\mathrm{T}} \ , \\ \underline{\boldsymbol{w}}_k &= [(\underline{\boldsymbol{w}}_k^l)^{\mathrm{T}}, (\underline{\boldsymbol{w}}_k^n)^{\mathrm{T}}]^{\mathrm{T}} \ . \end{aligned}$$

Here, $\underline{\boldsymbol{x}}_k^l \in \mathbb{R}^{n_x^l}$ denotes the state vector with (conditional) linear dynamics and $\underline{\boldsymbol{x}}_k^n \in \mathbb{R}^{n_x^n}$ denotes the nonlinear parts of the state. It is further assumed that the linear part $\underline{\boldsymbol{w}}_k^l$ and the nonlinear part $\underline{\boldsymbol{w}}_k^n$ of the system noise $\underline{\boldsymbol{w}}_k \sim \mathcal{N}(\underline{w}_k; \hat{\underline{w}}_k, \mathbf{C}_k^w)$ are stochastically independent, i.e.,

$$\mathbf{C}_k^w = \begin{bmatrix} \mathbf{C}_k^l & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_k^n \end{bmatrix} \ . \tag{6.28}$$

In case of stochastic dependencies between the linear and the nonlinear noise parts, techniques like the Gram-Schmidt procedure can be employed in order to decorrelate the noise [8, 95].

Example 6.5: Linear/Nonlinear States in Target Tracking
For target tracking problems (see Example 2.1) with distance or bearing measurements according to the sensor models (2.4) and (2.5), respectively, the state vector $\underline{\boldsymbol{x}}_k = [\boldsymbol{x}_k, \dot{\boldsymbol{x}}_k, \boldsymbol{y}_k, \dot{\boldsymbol{y}}_k]^{\mathrm{T}}$ can be decomposed in linear and nonlinear components according to

$$\begin{aligned} &\text{(velocity)} && \underline{\boldsymbol{x}}_k^l = [\dot{\boldsymbol{x}}_k, \dot{\boldsymbol{y}}_k]^{\mathrm{T}} \,, \\ &\text{(position)} && \underline{\boldsymbol{x}}_k^n = [\boldsymbol{x}_k, \boldsymbol{y}_k]^{\mathrm{T}} \,, \end{aligned}$$

with $\mathbf{A}_k^n = T \cdot \mathbf{I}$, $\mathbf{A}_k^l = \mathbf{I}$, $\underline{a}_k(\underline{\boldsymbol{x}}_k^n) = \underline{\boldsymbol{x}}_k^n$, and $\mathbf{H}_k(\underline{\boldsymbol{x}}_k^n) = \mathbf{0}$. Even if the system model is purely linear, the positions act as nonlinear states as they are nonlinearly mapped to the measurements by the function $\underline{h}_k(\underline{\boldsymbol{x}}_k^n)$. ■

For such systems, the prediction of the nonlinear part of the state vector can be carried out independently from the linear state. To see this, the density function $f_k^e(\underline{x}_k)$ representing the current state estimate can be written as

$$f_k^e(\underline{x}_k) = \underbrace{f_k^e(\underline{x}_k^l|\underline{x}_k^n)}_{\text{Optimal KF}} \cdot \underbrace{f_k^e(\underline{x}_k^n)}_{\text{Approximate HDF}} \tag{6.29}$$

by exploiting Bayes' law. Under consideration of the stochastically independent parts $\underline{\boldsymbol{w}}_k^l$ and $\underline{\boldsymbol{w}}_k^n$ of the system noise, performing the transition density approximation and HDF prediction is merely required for the nonlinear state. More specifically, with (6.28) and (6.29), the prediction step accords to

$$\begin{aligned} f_k^p(\underline{x}_k) &= \int_{\mathbb{R}^{n_x}} f_k^T(\underline{x}_{k+1}|\underline{x}_k) \cdot f_k^e(\underline{x}_k) \,\mathrm{d}\underline{x}_k \\ &= \int_{\mathbb{R}^{n_x^n}} \Bigg(\underbrace{\int_{\mathbb{R}^{n_x^l}} f_k^T(\underline{x}_{k+1}^l|\underline{x}_k) \cdot f_k^e(\underline{x}_k^l|\underline{x}_k^n) \,\mathrm{d}\underline{x}_k^l}_{\text{Kalman filter prediction}} \Bigg) \cdot \underbrace{f_k^T(\underline{x}_{k+1}^n|\underline{x}_k^n) \cdot f_k^e(\underline{x}_k^n) \,\mathrm{d}\underline{x}_k^n}_{\text{HDF prediction}} \,. \end{aligned}$$

Thanks to the reduced dimensionality of $\underline{\boldsymbol{x}}_k^n$, performing this part of the prediction can be performed much more efficiently compared to the HDF prediction of the whole state vector $\underline{\boldsymbol{x}}_k$. Furthermore, approximate prediction of the linear part is avoided, since the optimal Kalman filter prediction step can be applied to the linear state variables after processing the nonlinear part. Overall, this procedure, which is often referred to as Rao-Blackwellization [55, 161], leads to more accurate and computationally tractable estimates.

For incorporating measurement values into the Rao-Blackwellization scheme, the combined prediction and measurement update step described in Section 6.2.3 has to be applied. In doing so, the estimation of the linear parts of the state vector additionally involves the Kalman filter measurement update step.

6.3.3 Simulation Example

By means of the target localization and tracking scenario introduced in Example 2.1, the estimation performance of the proposed multivariate extension of the HDF is compared with particle filters (PF) using systematic resampling. The sampling interval T is set to be 1 s and the diffusion strength q is set to be 0.5. The initial estimate $\underline{\boldsymbol{x}}_0^e$ is assumed to be Gaussian with mean $\hat{\underline{x}}_0^e = [50\ \mathrm{m}, 1\ \mathrm{m/s}, 50\ \mathrm{m}, 1\ \mathrm{m/s}]^{\mathrm{T}}$ and covariance $\mathbf{C}_0^e = \mathrm{diag}([100\ \mathrm{m}^2, 1\ \mathrm{m}^2/\mathrm{s}^2, 100\ \mathrm{m}^2, 1\ \mathrm{m}^2/\mathrm{s}^2])$.

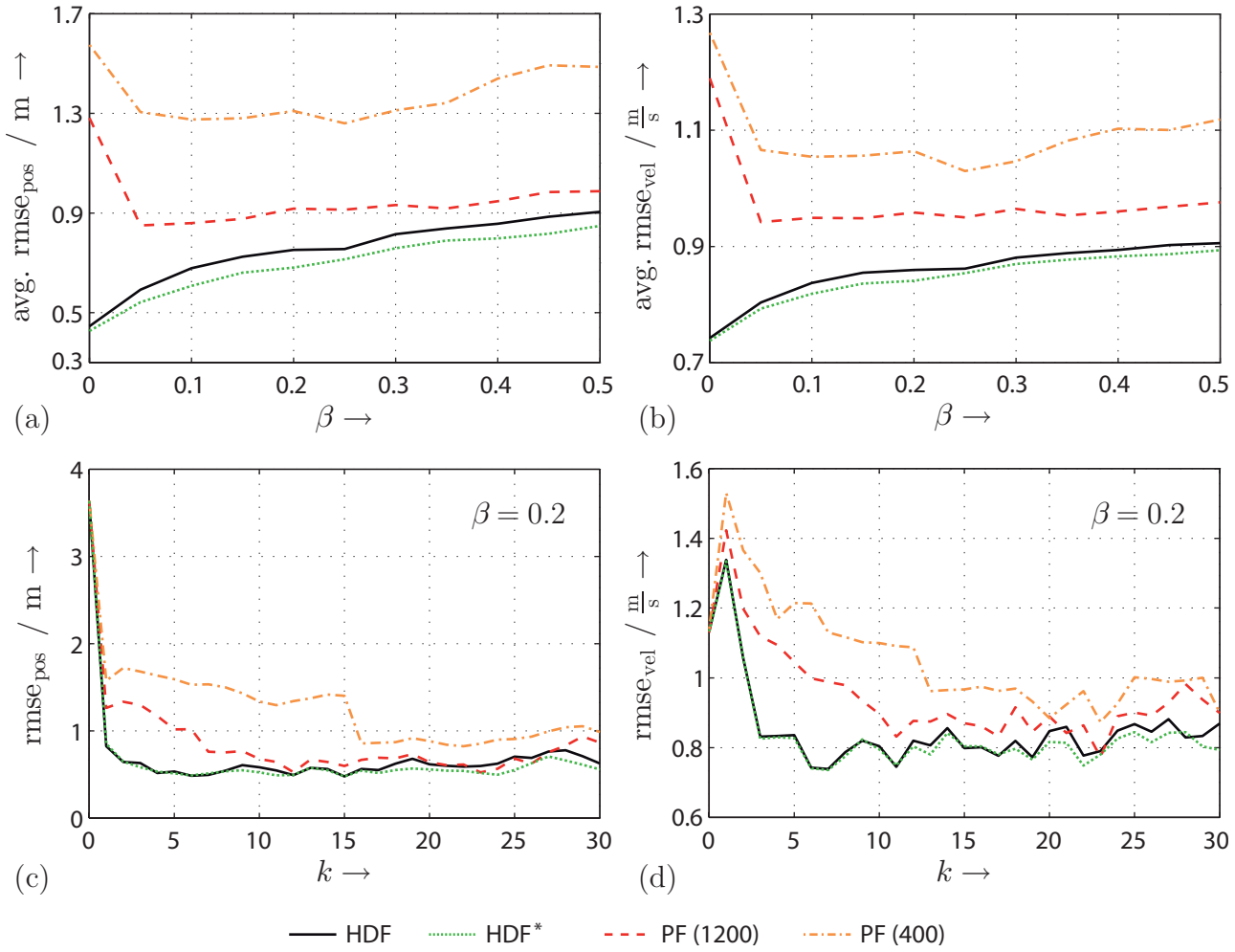


Figure 6.6: Tracking performance for radar target tracking with glint noise. Depending on the glint noise probability β, the average root mean square error (rmse) of (a) the position estimation and (b) the velocity estimation are depicted. (c) The rmse of the position estimates for $\beta = 0.2$. (d) The rmse of the velocity estimates for $\beta = 0.2$.

For this simulation example, radar tracking with glint noise[2] is considered. Here, the radar sensor model is given by

$$\underline{\hat{z}}_k = \begin{bmatrix} \sqrt{\boldsymbol{x}_k^2 + \boldsymbol{y}_k^2} \\ \arctan\left(\frac{\boldsymbol{y}_k}{\boldsymbol{x}_k}\right) \end{bmatrix} + \underline{\boldsymbol{v}}_k \ , \tag{6.30}$$

where the glint measurement noise $\underline{\boldsymbol{v}}_k$ is modeled by the Gaussian mixture

$$f_k^v(\underline{v}_k) = (1-\beta) \cdot \mathcal{N}(\underline{v}_k; \underline{0}, \mathbf{C}_k^{v,1}) + \beta \cdot \mathcal{N}(\underline{v}_k; \underline{0}, \mathbf{C}_k^{v,2}) \ , \tag{6.31}$$

with covariances $\mathbf{C}_k^{v,1} = \mathrm{diag}([0.01^2\ \mathrm{m}^2, 1^2\ \mathrm{mrad}^2])$ and $\mathbf{C}_k^{v,2} = \mathrm{diag}([0.1^2\ \mathrm{m}^2, 10^2\ \mathrm{mrad}^2])$. The parameter β refers to the glint noise probability. Eleven probability values are used for simulation; the probability β is taken from the set $\{0, 0.05, \ldots, 0.5\}$. By increasing the glint probability β it is possible to investigate the performance of the estimators for stronger and more heavily tailored noise.

For the given simulation setup the following four estimators are considered:

2 For tracking with glint noise see e.g. [195].

HDF A HDF exploiting that merely the position $[\boldsymbol{x}_k, \boldsymbol{y}_k]^{\mathrm{T}}$ of the state vector is observed in (6.30). While 1200 Dirac delta distributions are employed for the prediction step, the number of Dirac delta distributions is reduced to 400 for the measurement update.

HDF* Additionally to considering only the position during the measurement update, this HDF further exploits the linearity of the system model. Hence, the prediction is performed via a bank of KFs, while the measurement update of the HDF still employs 400 Dirac delta distributions. Admittedly, the number of Gaussian mixture components cannot be longer kept on a constant level for this particular HDF. The PGMR algorithm described in Chapter 7 is executed after each prediction step in order to reduce the number of components below 10.

PF For comparison reasons, two PFs are used. One with 400 samples (like the HDF*) and one with 1200 (like the HDF).

For each probability value 50 Monte Carlo simulation runs are performed, where the target is tracked over 30 time steps. In Figure 6.6 (a) and (b) the average rmse with respect to the target position $[\boldsymbol{x}_k, \boldsymbol{y}_k]^{\mathrm{T}}$ and the target velocity $[\dot{\boldsymbol{x}}_k, \dot{\boldsymbol{y}}_k]^{\mathrm{T}}$ for all glint probabilities are depicted. It is observed that both HDFs outperform the PFs for each probability value. The main reason for this can be seen in Figure 6.6 (c) and (d), where the rmse of the position and velocity estimates over all simulation runs for the glint noise probability $\beta = 0.2$ is depicted. Compared to the HDFs, the estimates of PFs take much longer time to converge. This effect is more severe for the PF with 400 samples. Even if both the HDF* and the PF with 400 samples employ the same number of components, the performance of the HDF* is significantly better for three reasons. First, the Diracs of the HDF are more systematically placed. Second, since only the position needs to be considered during measurement updates, no approximate processing of the velocity states is required. Due to the correlation between velocity and position, this in turn improves the estimation accuracy for both parts of the state vector. For PFs, considering observed states only during measurement updates cannot be achieved in a straightforward manner. Third and finally, the prediction can be performed in an optimal fashion due the exploitation of the linearity of the system model. It is important to note, that this exploitation does *not* coincide with the Rao-Blackwellization described in Section 6.3.2. It comes naturally as the HDF processes Gaussian mixture densities. By not performing KF prediction and employing HDF prediction instead, the number of Diracs has to be increased in order to achieve a comparable estimation performance. To improve the performance of both PFs, Rao-Blackwellization could be employed. But compared to the HDF*, merely the velocity states could be represented by means of Gaussians, while the position states still have to be represented by particles in order to allow for efficient measurement updates.

From Figure 6.6 (a) and (b) a further observation can be made. For an increasing glint noise probability, the tracking performance of all estimators degrades since the glint measurement noise becomes more and more heavy tailored. Thereby, the degradation of the PF with 1200 is not that strong compared to all other estimators. This follows from the fact, that PFs typically favor wide likelihood functions, since more peaked likelihoods (corresponding to more accurate sensors) force sample degeneration. Hence, for $\beta = 0$ only the first Gaussian component of the measurement noise (6.31) is active and thus, both PFs provide poor results as the measurement noise is low. The performance already becomes significantly better for $\beta = 0.05$, as the second Gaussian component of the noise leads to a likelihood with larger support. The HDFs instead behave more naturally, i.e., for the most accurate sensor, the performance is superior. For all estimator holds that increasing the number of components/particles would attenuate the degradation for an increasing β and thus would improve the tracking performance. However, that would be at the expense of increased computation time.

6.4 Contrast to Prior Work

Applying the HDF prediction and measurement update step yields a parametric and continuous representation of the current state estimate in terms of a Gaussian mixture density. This is worth mentioning, since the HDF estimation steps can be also interpreted as a *deterministic sampling* of the prior density. Analogously to the Gaussian estimator in Section 5, the sampling is called deterministic due to the systematic placement of the Dirac delta distributions. In the specific case of the HDF, these distributions are always arranged as a grid over $\underline{\Omega}_k$. The sampling character of the HDF is illustrated in the following example on the basis of the prediction step.

Example 6.6: Sampling Interpretation of the HDF Prediction Step
Except for the constant factor ω_i, the weights $\omega_{k+1,i}$ in (6.15) coincide with the function values of $f_k^x(\hat{\underline{x}}_i)$. Thus, $f_k^x(\underline{x}_k)$ can be replaced with the (deterministic) sample representation

$$f_k^x(\underline{x}_k) = \sum_{i=1}^{L} \omega_{k+1,i} \cdot \delta(\underline{x}_k - \hat{\underline{x}}_i) \ .$$

Using the true transition density $\tilde{f}_k^T(\underline{x}_{k+1}|\underline{x}_k)$ in the Chapman-Kolmogorov equation (2.7) leads to

$$\begin{aligned} f_{k+1}^p(\underline{x}_{k+1}) &= \int_{\mathbb{R}^{n_x}} \tilde{f}_k^T(\underline{x}_{k+1}|\underline{x}_k) \cdot f_k^x(\underline{x}_k) \, \mathrm{d}\underline{x}_k \\ &= \sum_{i=1}^{L} \omega_{k+1,i} \underbrace{\int_{\mathbb{R}^{n_x}} \tilde{f}_k^T(\underline{x}_{k+1}|\underline{x}_k) \cdot \delta(\underline{x}_k - \hat{\underline{x}}_i) \, \mathrm{d}\underline{x}_k}_{= f_k^w\left(\underline{x}_{k+1} - \underline{a}_k(\hat{\underline{x}}_i)\right)} \\ &= \sum_{i=1}^{L} \omega_{k+1,i} \cdot \mathcal{N}(\underline{x}_{k+1}; \hat{\underline{y}}_i, \mathbf{C}_k^w) \ , \end{aligned}$$

which is identical to (6.15) (for multivariate states). Thus, the HDF prediction step can be interpreted as an approximation of the underlying nonlinear system as a result of approximating the transition density. Alternatively, it can be considered as a sample approximation of the prior density. This interpretation holds also for the measurement update step and offers a very convenient way for implementing the HDF. ■

Random sampling based estimators like particle filters typically do not generate a continuous representation of the predicted or posterior density. Since they use Monte Carlo techniques, a sample representation is generated instead. Exceptions are the Gaussian (sum) particle filter [100, 101] and regularized particle filter [133]. Instead of resampling, these particle filters convert the samples into a continuous density representation, for instance into a Gaussian density in case of the Gaussian particle filter. However, still random sampling is applied, i.e., the estimation results are not deterministic and reproducible. Furthermore and in contrast to the HDF, particle filters are often incapable of maintaining multimodality during the entire estimation process. Especially due to the sample degeneration problem, it may happen that entire modes get lost. The HDF instead allows capturing the shape of the density function and thus information about higher-order moments more accurately.

As one of the major strength of particle filters it has been asserted that they avoid the *curse of dimensionality*, i.e., their complexity does not grow exponentially with the dimension of the state space. According to [49], this statement is wrong in general. Due to the unsystematic (random) sample placement, most of the samples are wasted, as they are not concentrated

on the relevant volume of the state space, whereas the volume growth exponentially with the dimension [48].

As an alternative to random sampling, Quasi-Monte Carlo estimators use deterministically drawn samples [136, 145]. The techniques used for generating these samples are often very complex and thus scalability is a critical problem [139]. A likewise computationally demanding but optimal approximation of arbitrary prior densities with deterministically drawn samples is proposed in [164]. In contrast to the HDF, this estimator solves an optimization problem on the prior density. Due to the optimal placement, few samples are sufficient to achieve precise estimation results. For an improved on-line performance, a suboptimal version, where the samples are placed in a greedy fashion, is also available [73, 97].

A further class of related nonlinear estimators are grid-based methods [102], where the grid points, i.e., the Dirac delta distributions, represent probability regions of the continuous state space[3]. This is different to the HDF, where the grid arrangement for the Dirac delta distributions is merely used for approximating the conditional densities. Since no continuous density representation is used in grid-based methods, the grid must be sufficiently dense to get a good approximation of the state space. Furthermore, the discretization of the state space must be predefined, while the grid part of the hybrid density can be determined on-line. Both methods suffer from increasing computational costs as the dimensionality of the state space increases. But in case of the HDF, this effect can be attenuated with the techniques proposed in Section 6.3.2.

6.5 Summary

The hybrid density filter is based on approximating the conditional densities involved in prediction and measurement update steps. A hybrid density consisting of Dirac delta distributions and Gaussian densities is used for approximation purposes. To achieve a high quality approximation of the conditional density and thus of the estimation results, the approximation is formulated as an optimization problem. For scalar states, this optimization problem can be solved analytically thanks to the special structure of the hybrid density, while for multivariate states, an optimal solution is computationally intractable since no closed-form approximation of the occurring multivariate uniform distributions exists so far. Instead, a suboptimal but practical grid-based approximation of the uniform distributions is employed.

Given the hybrid conditional density approximation, the prediction and measurement update step for the nonlinear system and sensor models can be performed in closed form. The HDF state estimation process can also be interpreted as deterministic sampling due to the Dirac delta distributions. For low-dimensional states, predictions and measurement updates can be performed with low computational effort. Compared to particle filters, which utilize random sampling, the HDF has the advantage of a lower number of required samples as well as a continuous density representation. Also, a random number generator is not required, which leads to a simple implementation. With increasing dimensionality of the state space, the number of required hybrid density components grows exponentially. To face this problem, methods for improving the efficiency have been proposed, namely measurement updates for observed states only and the integration of the well-known Rao-Blackwellization technique into the HDF.

Considering the fact that the characteristics of the nonlinear system and measurement functions have no impact on the placement of the Dirac delta distributions of the hybrid density, it is intended to investigate more elaborate distance measures, e.g., by employing a generalized version of the Cramér-von Mises distance based on localized cumulative distribution functions [72]. This could also counter the problem of the exponential growth of the number of components.

3 If the state space is discrete and finite, grid-based methods provide optimal estimates [9].

Furthermore, information about the prior density function, besides its support, should be incorporated into the conditional density approximation in order to arrange the hybrid density components even more systematically.

CHAPTER 7

Progressive Gaussian Mixture Reduction: A Constructive Approach

Thanks to their universal approximation property, Gaussian mixtures are a very convenient function system for representing probability densities. Besides their usage as result of the proposed state estimators (Gaussian mixture estimator and HDF), Gaussian mixtures are also employed for accurately representing multimodalities in tasks like multi-target tracking [14], density estimation [76, 142], or machine learning [44], just to name a few. However, recursive processing of Gaussian mixtures generally leads to an exponential growth of the number of mixture components (see for example Remark 5.1). But also the point-wise evaluation of Gaussian mixtures with a large number of components, as required for example for the HDF, is computationally demanding. In order to keep the computational and memory requirements bounded, it is inevitable to approximate a Gaussian mixture by one with fewer components.

Several methods were developed in the recent years for reducing the number of Gaussian components. Typically, the reduction is achieved by deleting components with low contribution to the overall mixture or by successively merging components with strong similarity. For a brief introduction on state-of-the-art reduction algorithm see Section 7.4. To fully exploit the approximation potential of reduced-order Gaussian mixtures, the *progressive Gaussian mixture reduction* (PGMR) approach introduced in this chapter employs a principle dual to existing algorithms. Instead of repeatedly removing mixture components, a Gaussian mixture is successively constructed to approximate the original mixture with far less components. However, a priori determining the optimal number of components for maintaining a specific deviation to the original mixture is impossible. Thus, a homotopy continuation[1] approach is employed, which starts with a single Gaussian density (see Section 7.1). During the continuation towards the original mixture, new Gaussian components are added to the approximate mixture for providing better approximation capabilities. To control this growth in components, the deviation is constantly tracked by a squared integral distance measure and new components are only added in regions of emerging strong deviations when necessary.

For obtaining accurate results in an efficient way, the progress of the continuation is controlled by a predictor-corrector scheme (see Section 7.2). The continuation progresses faster whenever the changes generated by the gradually incorporated original mixture are marginal. On the other hand, strong changes slow down the progression such that accurately adapting the approximation is possible. Furthermore, for enabling an efficient implementation of the proposed reduction method, closed-form solutions for all necessary calculations are derived.

The fundamentals of the proposed progressive Gaussian mixture reduction algorithm were published in [219] for univariate Gaussian mixtures. This chapter extends the results of this paper to the multivariate case.

1 For an introduction to homotopy continuation see for example [3].

7.1 Gaussian Mixture Reduction via Homotopy Continuation

In the following, it is assumed that the true density function of the random vector $\underline{\boldsymbol{x}}$ is represented by the Gaussian mixture

$$\tilde{f}(\underline{x}) = \sum_{i=1}^{M} \omega_i \cdot \mathcal{N}\left(\underline{x}; \underline{\hat{x}}_i, \mathbf{C}_i\right) \ .$$

In typical state estimation tasks, for example by applying the Gaussian mixture estimator introduced Section 5.5, the number of mixture components M increases exponentially over time. Due to computational and memory limitations, this growing mixture cannot be processed for any significant time span. Even if $\tilde{f}(\underline{x})$ has a large number of components, the shape of the Gaussian mixture is often not that complex, e.g., a mode of the true density is represented by several Gaussians, but a single component would be adequate for approximating the mode. Thus, a Gaussian mixture with a considerable smaller number of components can typically be found by fusing locally shared information and by removing redundancy in $\tilde{f}(\underline{x})$.

7.1.1 Problem Formulation

The goal is now to determine a *reduced Gaussian mixture* $f(\underline{x}; \underline{\eta})$ consisting of $L \ll M$ components, that is close to the original mixture $\tilde{f}(\underline{x})$. For representing the reduced Gaussian mixture, a special case of a Gaussian mixture with *axis-aligned* Gaussian components (short: *axis-aligned Gaussian mixture*) is employed. Here, the covariance matrix $\mathbf{C}_j$ of each Gaussian component j is diagonal, i.e., each component is separable in every dimension (see Appendix A.1) and thus, the reduced Gaussian mixture can be written according to

$$f(\underline{x}; \underline{\eta}) = \sum_{j=1}^{L} \omega_j^2 \cdot \mathcal{N}\left(\underline{x}; \underline{\hat{x}}_j, \mathbf{C}_j\right) = \sum_{j=1}^{L} \omega_j^2 \cdot \prod_{n=1}^{n_x} \mathcal{N}\left(x_n; \hat{x}_{j,n}, \sigma_{j,n}^2\right) \tag{7.1}$$

with parameter vector

$$\underline{\eta} = [\underline{\eta}_1^{\mathrm{T}}, \underline{\eta}_2^{\mathrm{T}}, \ldots, \underline{\eta}_L^{\mathrm{T}}]^{\mathrm{T}} \quad \text{and} \quad \underline{\eta}_j^{\mathrm{T}} = [\omega_j, \underline{\hat{x}}_j^{\mathrm{T}}, \underline{\sigma}_j^{\mathrm{T}}] \ .$$

The vector $\underline{\sigma}_j$ comprises the main diagonal elements, i.e., the standard deviations of the covariance matrix $\mathbf{C}_j$. Please note that squared weighting coefficients ω_j^2 are used to ensure that $f(\underline{x}; \underline{\eta})$ remains a valid density function during the reduction process, i.e., the weighting coefficients do not become negative.

Using an axis-aligned Gaussian mixture for representing the reduced mixture seems to be disadvantageous at the first glance, since an axis-aligned Gaussian mixture has minor approximation capabilities compared to a general one. Hence, an axis-aligned reduced Gaussian mixture typically requires more components to achieve a comparable approximation quality. But in exchange, less parameters for a single component have to be adjusted since the covariance matrices are diagonal. Furthermore, the necessary determination of the derivatives with respect to the parameter vector $\underline{\eta}$ proves to be easier. Altogether, representing $f(\underline{x}; \underline{\eta})$ as in (7.1) lowers the algorithmic complexity of the proposed Gaussian mixture reduction algorithm.

In order to determine the reduced Gaussian mixture, the Gaussian mixture reduction problem is formulated as an optimization problem

$$\begin{aligned} \underline{\eta}_{\min} &= \arg \min_{\underline{\eta}} G\big(\tilde{f}(\underline{x}), f(\underline{x}; \underline{\eta})\big) \\ \text{w.r.t.} \quad & G\big(\tilde{f}(\underline{x}), f(\underline{x}; \underline{\eta}_{\min})\big) \leq G_{\max} \end{aligned} \tag{7.2}$$

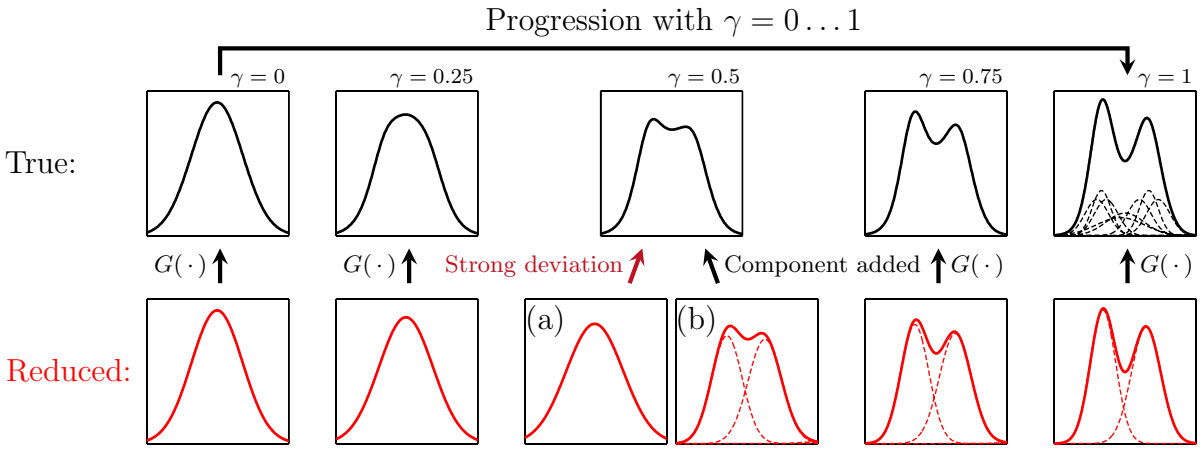


Figure 7.1: Principle of the progressive Gaussian mixture reduction. The reduction starts with a single Gaussian density for $\gamma = 0$, which is progressively transformed towards the true mixture by increasing γ. The progression and thus the deviation between the true and the reduced mixture is continuously tracked by means of the distance measure $G(\cdot)$. (a) Whenever the maximum approximation error $G_{\max}$ is violated, as it is the case here for $\gamma = 0.5$, (b) a new components is added to the reduced mixture in order to increase the approximation capabilities. For $\gamma = 1$, the true mixture is reached and the reduction process ends up with the desired reduced Gaussian mixture with parameter vector $\underline{\eta}_{\min}$.

by minimizing a certain distance measure $G\big(\tilde{f}(\underline{x}), f(\underline{x};\underline{\eta})\big)$, which quantifies the deviation or the similarity between both mixtures. This in turn allows adapting the parameters in $\underline{\eta}$, i.e., the weights, means, and variances of $f(\underline{x};\underline{\eta})$ in order to minimize the deviation under the constraint that the deviation between $\tilde{f}(\underline{x})$ and $f(\underline{x};\underline{\eta})$ is less than an user-defined maximum value $G_{\max}$. Besides defining a maximum deviation it is also possible to additionally constrain the number of used components for $f(\underline{x};\underline{\eta})$.[2] Thus, the user is able to adjust the quality as well as the computational demand of the reduction by giving a limit on the allowed deviation and/or the number of components.

7.1.2 Progressive Processing

The optimization problem (7.2) is generally not convex, so that directly minimizing the deviation between both mixture densities results in getting trapped in an unappropriate local optimum. Furthermore, the optimal number of mixture components L is not known a priori for maintaining a deviation less than $G_{\max}$. To overcome these problems, the proposed reduction approach makes use of the Progressive Bayes framework introduced in [71], i.e., a specific type of homotopy continuation is applied in order to find the solution of (7.2) progressively.

In doing so, a so-called *progression parameter* $\gamma \in [0, 1]$ is used for parameterizing the original Gaussian mixture $\tilde{f}(\underline{x})$ in such a way that for $\gamma = 0$ the Gaussian mixture can be reduced directly, i.e., the exact solution of the optimization problem is known without deviation from $\tilde{f}(\underline{x})$. As illustrated in Figure 7.1, the effect of the original mixture is then introduced gradually by incrementing the progression parameter. This ensures a continuous transformation of the optimal solution of the initial optimization problem towards the desired original Gaussian mixture $\tilde{f}(\underline{x})$, by progressively adjusting the parameters $\underline{\eta}$ of $f(\underline{x};\underline{\eta})$ to keep $G\big(\tilde{f}(\underline{x}), f(\underline{x};\underline{\eta})\big)$ at a minimum.

2 Obviously, in case of an additional component constraint, the maximum deviation is not guaranteed to be maintained.

7.1.3 Parameterization and Initialization

For that purpose, the *parameterized Gaussian mixture* $\tilde{f}(\underline{x};\gamma)$ is introduced according to

$$\tilde{f}(\underline{x};\gamma) = \gamma \cdot \tilde{f}(\underline{x}) + (1-\gamma)\cdot \hat{f}(\underline{x}) \ ,$$

which satisfies the identities

$$\tilde{f}(\underline{x};0) = \hat{f}(\underline{x}) \ \text{ and } \ \tilde{f}(\underline{x};1) = \tilde{f}(\underline{x}) \ .$$

Here, the density $\hat{f}(\underline{x})$ should be chosen in such a way that performing its reduction is straightforward. A very natural choice is a single Gaussian $\hat{f}(\underline{x}) = \mathcal{N}(\underline{x};\hat{\underline{x}},\tilde{\mathbf{C}})$, whose mean $\hat{\underline{x}}$ and covariance matrix $\tilde{\mathbf{C}}$ correspond to the mean and covariance of the original mixture $\tilde{f}(\underline{x})$ (see (A.3) and (A.4), respectively). Admittedly, the covariance of an arbitrary Gaussian mixture is not diagonal and thus, cannot be directly employed for initializing the progression. Instead, the diagonal covariance matrix $\mathbf{C} = \operatorname{diag}([\sigma_1^2, \sigma_2^2, \ldots, \sigma_{n_x}^2])$ with the variances

$$\sigma_i^2 = \sum_{j=1}^{n_x} |\tilde{c}(i,j)| \ , \tag{7.3}$$

is proposed for the initial Gaussian $\hat{f}(\underline{x})$, where $\tilde{c}(i,j)$ is the element of the covariance matrix $\tilde{\mathbf{C}}$ at row i and column j. By means of the row sum criterion it can be shown, that the diagonal covariance matrix $\mathbf{C}$ with elements according to (7.3) fulfills

$$\mathbf{C} \succeq \tilde{\mathbf{C}} \ .$$

In a concrete manner, the covariance ellipsoid corresponding to $\mathbf{C}$ fully contains the ellipsoid of $\tilde{\mathbf{C}}$. Moreover, for a scalar random variable $\boldsymbol{x}$, $\mathbf{C}$ coincides with $\tilde{\mathbf{C}}$ and for the bivariate case the covariance ellipse of $\mathbf{C}$ is the tightest possible axis-aligned ellipse.

By this choice for an initial density, covering the support of the true Gaussian mixture is ensured. Starting the progression with this "simple" density allows directly determining the optimal solution, i.e., $f(\underline{x};\underline{\eta}) = \hat{f}(\underline{x})$ with $\underline{\eta} = [1, \hat{\underline{x}}^{\mathrm{T}}, \sigma_1^2, \sigma_2^2, \ldots, \sigma_{n_x}^2]^{\mathrm{T}}$ for $\gamma = 0$. This solution, i.e., the parameter vector $\underline{\eta}$, then tracks the original Gaussian mixture that is progressively modified by increasing γ as depicted in Figure 7.1.

As the initialization of the continuation indicates, the way the proposed mixture reduction approach operates is dual to existing algorithms. Instead of beginning with the complete original mixture, at first a less complex reduction or approximation task is solved. As it is illustrated in Figure 7.1 (a)–(b) and explained in more detail in Section 7.2.2, splitting or insertion operations are used in order to add new Gaussian components to the initial single Gaussian if required. This ensures to achieve the maximum deviation value $G_{\max}$.

7.1.4 Distance Measure

For quantifying the deviation between $\tilde{f}(\underline{x};\gamma)$ and $f(\underline{x};\underline{\eta})$, several measures $G(\,\cdot\,)$ can be used. For convenience, the squared integral distance measure [86]

$$G\big(\tilde{f}(\underline{x};\gamma), f(\underline{x};\underline{\eta})\big) = \frac{1}{2}\int_{\mathbb{R}^{n_x}} \Big(\tilde{f}(\underline{x};\gamma) - f(\underline{x};\underline{\eta})\Big)^2 \,\mathrm{d}\underline{x} \tag{7.4}$$

is chosen, since it can be evaluated analytically for Gaussian mixtures. However, the proposed approach is not restricted to this specific deviation measure. For instance, the Kullback-Leibler divergence (2.30) can also be used, especially as it is the ideal deviation measure for mixture

reduction in a maximum likelihood sense [153, 193]. Due to the fact that it is impossible to evaluate this measure in closed form for Gaussian mixtures, numerical integration schemes have to be employed, which leads to increased computational costs.

In the following, $G(\underline{\eta}, \gamma)$ is used as shorthand term for $G\big(\tilde{f}(\underline{x}; \gamma), f(\underline{x}; \underline{\eta})\big)$.

7.1.5 Progressive Minimization

To perform the progression of γ from 0 to 1, while keeping the distance measure at its minimum, the differential relation between γ and the parameter vector $\underline{\eta}$, i.e., the variation of $\underline{\eta}$ depending on the variation of γ, is required. Hence, the optimization problem (7.2) is transformed into a system of ordinary differential equations (ODE). In order to obtain these differential equations, the necessary condition of a minimum of $G(\underline{\eta}, \gamma)$ has to be satisfied. Thus, derivatives of $G(\underline{\eta}, \gamma)$ with respect to γ and $\underline{\eta}$ have to be zero, as $G(\underline{\eta}, \gamma)$ is a function over γ and $\underline{\eta}$. Taking the partial derivative of $G(\underline{\eta}, \gamma)$ with respect to the parameter vector $\underline{\eta}$ yields

$$\frac{\partial G(\underline{\eta}, \gamma)}{\partial \underline{\eta}} = -\int_{\mathbb{R}^{n_x}} \Big(\tilde{f}(\underline{x}; \gamma) - f(\underline{x}; \underline{\eta})\Big)\, \underline{F}(\underline{x}; \underline{\eta})\, \mathrm{d}\underline{x} \;, \tag{7.5}$$

where

$$\underline{F}(\underline{x}; \underline{\eta}) = \frac{\partial f(\underline{x}; \underline{\eta})}{\partial \underline{\eta}} \;.$$

Setting (7.5) to zero results in

$$\int_{\mathbb{R}^{n_x}} \tilde{f}(\underline{x}; \gamma) \cdot \underline{F}(\underline{x}; \underline{\eta})\, \mathrm{d}\underline{x} = \int_{\mathbb{R}^{n_x}} f(\underline{x}; \underline{\eta}) \cdot \underline{F}(\underline{x}; \underline{\eta})\, \mathrm{d}\underline{x} \;.$$

The partial derivative with respect to γ gives the desired system of ordinary first-order differential equations

$$\int_{\mathbb{R}^{n_x}} \underline{F}(\underline{x}; \underline{\eta}) \cdot \frac{\partial \tilde{f}(\underline{x}; \gamma)}{\partial \gamma}\, \mathrm{d}\underline{x} = \left(\underbrace{\int_{\mathbb{R}^{n_x}} \underline{F}(\underline{x}; \underline{\eta}) \cdot \underline{F}(\underline{x}; \underline{\eta})^{\mathrm{T}}\, \mathrm{d}\underline{x}}_{=:\mathbf{P}'(\underline{\eta})} + \underbrace{\int_{\mathbb{R}^{n_x}} \Big(f(\underline{x}; \underline{\eta}) - \tilde{f}(\underline{x}; \gamma)\Big) \cdot \mathbf{M}(\underline{x}, \underline{\eta})\, \mathrm{d}\underline{x}}_{=:\Delta\mathbf{P}(\underline{\eta}, \gamma)} \right) \cdot \frac{\partial \underline{\eta}}{\partial \gamma} \;,$$

where

$$\mathbf{M}(\underline{x}, \underline{\eta}) = \frac{\partial^2 f(\underline{x}; \underline{\eta})}{\partial \underline{\eta}\, \partial \underline{\eta}^{\mathrm{T}}} \;.$$

This can be written as

$$\mathbf{P}(\underline{\eta}, \gamma) \cdot \dot{\underline{\eta}} = \underline{b}(\underline{\eta}, \gamma) \;, \tag{7.6}$$

where the coefficients are given by

$$\mathbf{P}(\underline{\eta}, \gamma) = \mathbf{P}'(\underline{\eta}) + \Delta\mathbf{P}(\underline{\eta}, \gamma) \;, \tag{7.7}$$

$$\underline{b}(\underline{\eta}, \gamma) = \int_{\mathbb{R}^{n_x}} \underline{F}(\underline{x}; \underline{\eta}) \cdot \frac{\partial \tilde{f}(\underline{x}; \gamma)}{\partial \gamma}\, \mathrm{d}\underline{x} \;. \tag{7.8}$$

Algorithm 5 Pseudo-code of the numerical solver for (7.6)

1: $\gamma \leftarrow 0$
2: $\underline{\eta} \leftarrow \underline{\eta}(\gamma = 0)$
3: $\Delta\gamma \leftarrow \gamma_{\min}$
4: **repeat**
5: $\quad \gamma \leftarrow \gamma + \Delta\gamma$
6: $\quad \dot{\underline{\eta}} \leftarrow \text{solve}\left(\mathbf{P}(\underline{\eta}, \gamma), \underline{b}(\underline{\eta}, \gamma)\right)$
7: $\quad \underline{\eta}_{\text{tmp}} \leftarrow \underline{\eta} + \Delta\gamma \cdot \dot{\underline{\eta}}$ // Predictor
8: $\quad \left[\underline{\eta},\ \gamma,\ \Delta\gamma\right] \leftarrow \text{Adaptation}\left(\underline{\eta}_{\text{tmp}}, \gamma, \Delta\gamma, G_{\max}\right)$ // Corrector
9: **until** $\gamma = 1$

Closed-form expressions for (7.7) and (7.8) are given in Appendix B.1 and Appendix B.2, respectively. It is worth mentioning that the term $\Delta\mathbf{P}(\underline{\eta}, \gamma)$ can be omitted. This can be justified by the fact that $\Delta\mathbf{P}(\underline{\eta}, \gamma)$ comprises second-order derivatives of the reduced mixture as well as the difference between the reduced and the complex true mixture. Hence, the complexity of evaluating this term is large, while one the other hand, the resulting values are negligible compared to $\mathbf{P}'(\underline{\eta})$. As shown in [71], omitting $\Delta\mathbf{P}(\underline{\eta}, \gamma)$ coincides with replacing the reduced Gaussian mixture by its first-order Taylor-series expansion around a given nominal parameter vector.

7.1.6 Solving the System of Ordinary Differential Equations

The system of ODEs (7.6) cannot be solved analytically in general. Thus, a numerical solution scheme has to be used. One option is to employ well-known ODE solvers like Runge-Kutta. However, for this specific case these methods often turned out to be numerically unstable and the integration of new mixture components is not supported.

Instead, the numerical solver listed in Algorithm 5 is proposed. As aforementioned, the algorithm starts with $\gamma = 0$ and thus with an optimal choice of the parameter vector $\underline{\eta}$ (see line 1-2). During the solution process, γ is gradually increased while $\underline{\eta}$ is simultaneously adjusted (line 5-8). Please note that solving the ODE in line 6 can be carried out directly, as γ is a fixed value and thus, merely a system of linear equations $\mathbf{P} \cdot \dot{\underline{\eta}} = \underline{b}$ has to be solved, e.g., by employing LU factorization.

With the solution vector $\dot{\underline{\eta}}$ for a specific γ, a so-called *predictor-corrector scheme* can be realized, which is quite common in homotopy continuation [3, 163]. Here, the predictor is represented by line 7. By means of extrapolation, the predictor generates an approximate parameter vector $\underline{\eta}_{\text{tmp}}$ further along the solution curve of $\partial^2 G(\underline{\eta}, \gamma) / \left(\partial\underline{\eta}\partial\gamma\right) = \underline{0}$. For this purpose, $\dot{\underline{\eta}}$ gives the direction for predicting $\underline{\eta}$, while the step size $\Delta\gamma$ gives the increment in prediction direction.

Typically, this prediction step causes an error governed by the current step size. For reducing the introduced error under the user-defined error bound $G_{\max}$, a correction or adaptation step is applied subsequently (line 8). In this thesis, the term adaptation is used instead of correction, as not only a correction of $\underline{\eta}$ is performed after the prediction. In fact, new Gaussian components are introduced by splitting existing Gaussians or by inserting new Gaussians, if the deviation between $\tilde{f}(\underline{x}; \gamma)$ and $f(\underline{x}; \underline{\eta})$ is still larger than $G_{\max}$. This procedure facilitates *adapting* $f(\underline{x}; \underline{\eta})$ to emerging structural changes in $\tilde{f}(\underline{x}; \gamma)$ during the progression. The methods used for adaptation are described in detail in the following section.

7.2 Adaptation Methods

A straightforward way to realize the adaptation is to keep the step size always at the minimum value $\gamma_{\min}$. This leads to a linear increment of γ. But then, choosing an appropriate $\gamma_{\min}$ is critical, since one has to balance between a compensation of even marginal changes in $f(\underline{x};\gamma)$ and a fast progression, leading to a coarse error reduction at some parts of the progression.

7.2.1 Parameter and Step Size Adaptation

Since the distance measure has to be minimized for a specific γ, a Newton approach for determining the roots of (7.5) is applied [163] instead. This allows correcting $\underline{\eta}$ in order to compensate the introduced error. Furthermore, γ and the step size $\Delta\gamma$ are adjusted for controlling the speed of the progression. This can be done due to the fact that a fast convergence of the Newton approach indicates only a small error introduced by the prediction. Hence, if the variation of the true mixture is mild, the step size can be increased for the next progression step, which leads to an acceleration of the progression. The opposite case, where the Newton approach does not converge, indicates a large error. Thus, the prediction step can be reverted by setting γ to its former value and the step size can be decreased. The progression slows down in cases of strong variations and an adequate adaptation is permitted.

For obtaining this adaptation, the Newton approach

$$\mathbf{H}(\underline{\eta}_k,\gamma)\cdot\Delta\underline{\eta} = -\frac{\partial G(\underline{\eta},\gamma)}{\partial\underline{\eta}}\bigg|_{\underline{\eta}=\underline{\eta}_k} = -\,\underline{g}(\underline{\eta}_k,\gamma)\ , \tag{7.9}$$

has to be applied. A closed-form expression of the gradient $\underline{g}(\underline{\eta}_k,\gamma)$ of the distance measure is given in Appendix B.3, while the Hessian

$$\mathbf{H}(\underline{\eta}_k,\gamma) = \frac{\partial^2 G(\underline{\eta},\gamma)}{\partial\underline{\eta}\,\partial\underline{\eta}^{\mathrm{T}}}\bigg|_{\underline{\eta}=\underline{\eta}_k}\ ,$$

is identical to the matrix $\mathbf{P}$ in (7.7).[3] $\Delta\underline{\eta} = \underline{\eta}_{k+1} - \underline{\eta}_k$ is determined by solving the system of linear equations (7.9), which yields the recursion

$$\underline{\eta}_{k+1} = \underline{\eta}_k + \Delta\underline{\eta}\ .$$

This recursion is initialized with $\underline{\eta}_0 = \underline{\eta}_{\mathrm{tmp}}$ (obtained at line 7 of Algorithm 5). In cases where this initial value is close to the true parameter vector, the method quickly converges, which can be detected by $\Delta\underline{\eta} \to \underline{0}$.

Algorithm 6 summarizes the correction method for $\underline{\eta}$, γ, and $\Delta\gamma$. Again, the system of linear equations in line 3 can be solved efficiently using LU factorization. In addition to controlling the convergence of the Newton approach, adapting $\underline{\eta}$ is aborted after a maximum number of steps $k_{\max}$ (see line 5). The structural adaptation performed in line 8 is described in detail in the following section.

7.2.2 Structural Adaptation

Performing the correction step does not guarantee that the maximum deviation $G_{\max}$ is maintained. This is especially the case when new modes emerge due to gradually incorporating the true Gaussian mixture. Here, the current number of components of the reduced mixture may

3 As motivated in Section 7.1.5, the term $\Delta\mathbf{P}$ can be omitted here for decreasing the computational demand.

Algorithm 6 $[\underline{\eta}, \gamma, \Delta\gamma] \leftarrow \text{Adaptation}\big(\underline{\eta}_{\text{tmp}}, \gamma, \Delta\gamma, G_{\max}\big)$

$\underline{\eta}_0 \leftarrow \underline{\eta}_{\text{tmp}}$
repeat
  $\Delta\underline{\eta} \leftarrow \text{solve}\left(\mathbf{H}(\underline{\eta}_k, \gamma), \underline{g}(\underline{\eta}_k, \gamma)\right)$
  $\underline{\eta}_{k+1} \leftarrow \underline{\eta}_k + \Delta\underline{\eta}$
until $k+1 = k_{\max}$ or $\Delta\underline{\eta} \to 0$
if $\Delta\underline{\eta} \to 0$ **then** // Newton method converged
  $\Delta\gamma \leftarrow \text{Increase}(\Delta\gamma)$ // Accelerate progression
  $\underline{\eta} \leftarrow \text{StructuralAdaptation}(\underline{\eta}_{k+1}, G_{\max})$
else
  $\underline{\eta} \leftarrow \underline{\eta}_{\text{tmp}}$
  $\gamma \leftarrow \gamma - \Delta\gamma$
  $\Delta\gamma \leftarrow \text{Decrease}(\Delta\gamma)$ // Decelerate progression
end if

not suffice to capture this structural change and new components have to be added to the reduced mixture in order to improve its approximation capabilities. Therefore, two possible ways for adding new components are introduced in the following: *component splitting* and *component insertion*.

Normalized Distance Measure

Independent of the adding method, at first it has to be identified whether the approximation provided by the reduced mixture suffices or not. For enabling a scale-invariant check of the deviation between $\tilde{f}(\underline{x}; \gamma)$ and $f(\underline{x}; \underline{\eta})$, the *normalized distance measure*

$$G_N(\underline{\eta}, \gamma) = \sqrt{\frac{\int_{\mathbb{R}^{n_x}} \left(\tilde{f}(\underline{x}; \gamma) - f(\underline{x}; \underline{\eta})\right)^2 \mathrm{d}\underline{x}}{\int_{\mathbb{R}^{n_x}} \tilde{f}(\underline{x}; \gamma)^2 \,\mathrm{d}\underline{x} + \int_{\mathbb{R}^{n_x}} f(\underline{x}; \underline{\eta})^2 \,\mathrm{d}\underline{x}}} \tag{7.10}$$

is employed. Compared to the distance (7.4), this measure is more convenient for specifying limits on the maximum allowed deviation [71]. It ranges between zero, which is the case if $\tilde{f}(\underline{x}; \gamma)$ and $f(\underline{x}; \underline{\eta})$ are identical, and one, if both mixtures are absolutely non-overlapping.

Component Splitting

Once $G_N(\underline{\eta}, \gamma)$ is larger than $G_{\max}$, the progression is stopped and the number of components is increased. A straightforward way to introduce new mixture components is to *split* existing ones. In doing so, the most critical component, i.e., the component that is mainly responsible for the deviation, has to be identified by evaluating L individual distances

$$G_i(\underline{\eta}, \gamma) = \int_{\mathbb{R}^{n_x}} \left(\tilde{f}(\underline{x}; \gamma) - f(\underline{x}; \underline{\eta})\right)^2 \cdot f_i(\underline{x}; \underline{\eta}_i) \,\mathrm{d}\underline{x} \,, \tag{7.11}$$

where $i = 1, \ldots, L$ and $f_i(\underline{x}; \underline{\eta}_i) = \omega_i^2 \cdot \mathcal{N}(\underline{x}; \hat{\underline{x}}_i, \mathbf{C}_i)$. These individual distances can be evaluated in closed form and the component with maximum distance is selected for splitting. The distance measure (7.11) represents a weighted version of (7.4), which favours components that are close to regions of strong deviation between the original and reduced mixture.

Several possibilities arise for performing a split. They differ, e.g., in number of new components or the parameters of the new components. Simply reproducing the original component

is not sufficient since the symmetry has to be broken to facilitate approximating the critical region of the true Gaussian mixture in different ways [15].

In this thesis, splitting a component into two new Gaussians is preferred, since for two Gaussians a moment-preserving replacement with respect to mean and covariance can be easily guaranteed [153]. Furthermore, splitting is only performed symmetrically around the mean vector *and* along a principal axis of the selected Gaussian component. Especially the latter eases splitting, because due to the used axis-aligned Gaussian the principal axes are parallel to the axes of the state space. In doing so, the selected component $\omega^2 \cdot \mathcal{N}(\underline{x}; \underline{\hat{x}}, \mathbf{C})$ can be replaced by $\frac{\omega^2}{2} \cdot \mathcal{N}(\underline{x}; \underline{\hat{x}}_1, \mathbf{C}_1)$ and $\frac{\omega^2}{2} \cdot \mathcal{N}(\underline{x}; \underline{\hat{x}}_2, \mathbf{C}_2)$. Assuming that the axis or dimension n is selected for splitting, the mean vectors $\underline{\hat{x}}_1$ and $\underline{\hat{x}}_2$ as well as the covariance matrices $\mathbf{C}_1$ and $\mathbf{C}_2$ coincide with the mean vector $\underline{\hat{x}}$ and covariance $\mathbf{C}$, respectively, except for the splitting dimension n. For this particular dimension, the corresponding mean vector elements and covariance matrix elements of $\underline{\hat{x}}_1, \underline{\hat{x}}_2, \mathbf{C}_1$ and $\mathbf{C}_2$ have to satisfy

$$\begin{aligned} \hat{x}_n &= \tfrac{1}{2} \cdot \hat{x}_{1,n} + \tfrac{1}{2} \cdot \hat{x}_{2,n} \ , \\ \sigma_n^2 &= \tfrac{1}{2} \cdot \sigma_{1,n}^2 + \tfrac{1}{2} \cdot \sigma_{2,n}^2 + \tfrac{1}{4} \cdot \left(\hat{x}_{1,n} - \hat{x}_{2,n}\right)^2 \ , \end{aligned}$$

which ensures a moment-preserving split of the reduced Gaussian mixture. For all simulations in this thesis the parameters

$$\hat{x}_{1,n} = \tfrac{1}{2} \cdot \sigma_n + \hat{x}_n \ , \qquad \hat{x}_{2,n} = -\tfrac{1}{2} \cdot \sigma_n + \hat{x}_n \ , \qquad \sigma_{1,n}^2 = \sigma_{2,n}^2 = \tfrac{3}{4} \cdot \sigma_n^2$$

are used.

Even if splitting is only performed along a principal axis, the possibilities of selecting an appropriate axis for splitting grows with the number of dimensions of the state space. To select an appropriate axis, the following procedure is proposed:

1. The selected Gaussian is (virtually) split along each principal axis. Thus, in total n_x splits have to be performed.
2. The individual distance (7.11) is evaluated for each of the n_x Gaussian pairs that result from step 1. For each pair, the distance values of both Gaussians are cumulated.
3. The split with the maximum (cumulative) distance value of step 2 is finally undertaken.

This procedure turned out to be a good trade-off between computational demand and identifying the split that will result in an adequate covering of the region of strong deviation between the original and reduced mixture by the added components. In doing so, the number of improper splits and their required adaptation to the true mixture can be kept on a minimum level. The drawback of this way of component splitting is that determining the principal axis for split becomes computationally demanding for high-dimensional state spaces as the number of virtual splits created in step 1 and evaluated in step 2 grows with the dimension of the state space.

Component Insertion

An alternative procedure for adding new components to the reduced mixture is the *insertion* of a completely new component. This procedure relies on the determination of the point of strongest deviation between the true and the reduced mixture. By directly inserting components at this point, the approximation capabilities of the reduced mixture are improved effectively and thus, emerging deviations during the progression can be resolved in situ. Furthermore, the demanding determination of the principal axis for splitting can be avoided, which makes the insertion procedure more scalable than component splitting.

For an effective component insertion, the so-called *deviation Gaussian mixture*

$$\bar{f}(\underline{x}) = \tilde{f}(\underline{x};\gamma) - f(\underline{x};\underline{\eta}) \tag{7.12}$$

is introduced, which consists of $L+M$ components. Since (7.12) represents the difference between the true and the reduced Gaussian mixture, it can be negative. Hence, $\bar{f}(\underline{x})$ describes no valid probability density function.

Extrema of the deviation Gaussian mixture indicate points of strong deviation, albeit negative minima are irrelevant for insertion, since the reduced Gaussian mixture already possesses more probability mass than the true mixture at the corresponding regions. Modes instead indicate a lack of approximation capability that can be eliminated by component insertion, whereas the highest mode is of special interest for an effective approximation error reduction.

Certainly, accurately determining all modes or even the highest one of a Gaussian mixture corresponds to a demanding optimization problem, whose solution generally requires iterative search methods [32]. Since being sufficiently close to the highest mode is adequate for component insertion, the high computational burden of an exact mode finding can be avoided by a heuristic procedure. It exploits that except for some pathologic cases, the positions of the modes almost coincide with a subset of the mean vectors of the mixture components. Hence, evaluating the deviation Gaussian mixture (7.12) at all mean vectors of its Gaussian components and taking the maximum value results in the point

$$\hat{\underline{x}}_{\max} = \arg \max_{\hat{\underline{x}}_i} \left\{ \bar{f}(\hat{\underline{x}}_i) \right\} \tag{7.13}$$

which is typically very close to the highest mode of $\bar{f}(\underline{x})$. Here, $\hat{\underline{x}}_i$, $i = 1, \ldots, L+M$ are the Gaussian mean vectors of $\tilde{f}(\underline{x};\gamma)$ and $f(\underline{x};\underline{\eta})$.

Given the point (7.13), a new Gaussian component $\omega \cdot \mathcal{N}(\underline{x}; \hat{\underline{x}}_{\max}, \mathbf{C})$ can be inserted into the reduced mixture. The remaining parameters ω and $\mathbf{C}$ of this Gaussian should by chosen in such a way that the reduced mixture significantly overvalues the true mixture at the considered region, e.g., by using small values for the elements of $\mathbf{C}$. In doing so, a strong artificial error is introduced, which forces an adaptation by the additional corrector step (see Example 7.1 on the next page.).

Component Deletion

During the progression it also occurs that components of the reduced mixture become negligible and thus, contribute almost nothing to the approximation of $\tilde{f}(\underline{x};\gamma)$. These components can be identified by the ratio $\omega_i^2 / \operatorname{trace}(\mathbf{C}_i)$ being close to zero. Deleting then reduces the complexity of the reduced Gaussian mixture, which in turn avoids overfitting effects. Furthermore, in cases where a maximum number of components is specified by the user, deleting components creates space for splitting or insertion operations, especially when the current number of components in $f(\underline{x};\underline{\eta})$ is close to the maximum.

Additional Correction Step

Structural adaptations by performing splitting, insertion or deletion of mixture components improves approximation capabilities and/or introduces an additional error. To adapt the parameters of newly added components and to reduce the introduced error, an additional correction step is performed by reapplying the Newton approach derived in Section 7.2.1. In the following example, the systematic differences between splitting and insertion as well as the effect of the additional correction step is demonstrated.

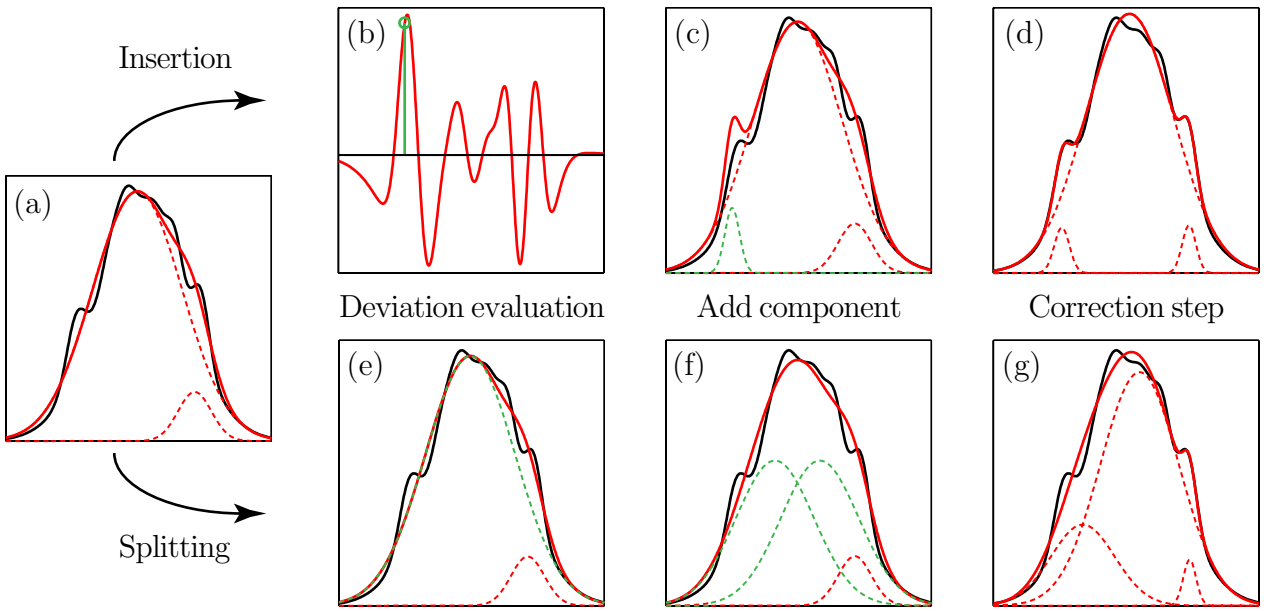


Figure 7.2: Splitting vs. insertion. (a) The true Gaussian mixture (black) and its reduced version (red) at a particular stage of the progression. Component insertion: (b) approximate determination of the highest mode of the deviation Gaussian mixture (red), (c) insertion of the new component (green), and (d) adaptation to the shape of the true mixture via the additional correction step. Component splitting: (e) determination of the Gaussian component (green) causing strong approximation errors, (f) splitting into two new Gaussians (green), and (g) imperfect adaptation by means of the additional correction step.

Example 7.1: Splitting vs. Insertion

Consider the scenario depicted in Figure 7.2 (a), where for a particular value of γ the true Gaussian mixture is approximated by a reduced mixture with two components. Applying insertion for adding a new component leads to the steps illustrated in Figure 7.2 (b)-(d). At first, the highest mode of the deviation Gaussian mixture (7.12) is determined approximately. At the position of this mode (indicated by the green stem in Figure 7.2 (b)) a new Gaussian component is added to the reduced mixture. It can be clearly seen that the new component (green Gaussian in Figure 7.2 (c)) is quite peaked in order that its adaptation is forced in the following correction step. Due to the precise placement of the newly added Gaussian, the correction step converges rapidly and the strongest deviation is corrected properly (see Figure 7.2 (d)). Furthermore, adding the new component facilitates to represent the right mode of the true mixture by the right Gaussian component thanks to the increased approximation capabilities.

This result is now compared with the splitting procedure. The first step here is to identify the Gaussian that maximizes the individual distance measure (7.11). Since the left Gaussian covers the whole support, it is mostly responsible for deviations between the true and reduced mixture. Accordingly, it is selected for splitting. The two Gaussians resulting from the split are depicted in Figure 7.2 (f). Even if the left of the two new Gaussians is relatively close to the region of strongest deviation, it is not properly shaped for capturing the left mode of true mixture. Hence, convergence of the additional correction step is much slower compared to the insertion step. More drastically, the remaining deviation between true mixture and its reduced version is still considerable (see Figure 7.2 (g)), if the correction is performed with the same number of steps that was necessary after the insertion procedure for obtaining the result depicted in Figure 7.2 (d). Only by significantly increasing the number of steps for correction, the same approximation result can be obtained. Consequently, the computation time of the splitting-based mixture reduction is typically larger than the computation time of the reduction based on insertion, while the approximation quality of both techniques is almost equivalent (see also the simulation results in the following section). ■

7.3 Simulation Results

For demonstrating the effectiveness of the proposed progressive Gaussian mixture reduction (PGMR) algorithm, three different simulations are conducted. First, the effect of the deviation bound $G_{\max}$ on the reduction quality is highlighted. Additionally, PGMR is compared to state-of-the-art reduction methods by means of reducing randomly generated Gaussian mixtures for scalar and two-dimensional random variables. For improved readability, all deviation values and bounds are multiplied by a factor 100.

7.3.1 Deviation Bound

The true Gaussian mixture $\tilde{f}(x)$ representing the scalar random variable $\boldsymbol{x}$ consists of $M = 10$ components, where the single Gaussians have weighting coefficients, means, and standard deviations according to

$$\begin{aligned}\omega &\in \{0.1\ 0.1\ 0.1\ 0.1\ 0.1\ 0.1\ 0.1\ 0.1\ 0.1\ 0.1\}\ ,\\ \hat{x} &\in \{-3.5\ \ -3\ \ -1\ 0\ 0.5\ 2\ 3\ 3.5\ 5\ 5.5\}\ ,\\ \sigma &\in \{0.6\ 0.6\ 0.6\ 0.6\ 0.7\ 0.7\ 1\ 0.5\ 0.5\ 0.5\}\ ,\end{aligned}$$

is reduced by PGMR with varying deviation bound $G_{\max} \in \{0.5, 0.75, 2, 4\}$. Both techniques for adding new components to the reduced mixtures are applied. In Table 7.1, the used number of components L as well as the deviation between the true Gaussian mixture and its reduced version are listed. The normalized distance measure (7.10) is used for quantifying the deviation. It can be seen that splitting and insertion result in almost the same reduced mixtures in this particular example.

By increasing the maximum deviation value $G_{\max}$, the number of used components decreases as expected. This comes along with a reduced computation time since less structural adaptation operations have to be performed for more relaxed deviation limits. In Figure 7.3, the true Gaussian mixture (black, dashed) is depicted together with the reduced Gaussian mixture (red, solid) for $G_{\max} = 2$. Furthermore, the individual Gaussian components of $f(x;\underline{\eta})$ are also shown. Considering the five modes of the true mixture, one might expect that using also five mixture components would result into a precise approximation. This is almost true except for the second mode at $x \approx 0$, which cannot be fitted appropriately by a single Gaussian. Thus, a considerable improvement of the reduction quality is gained for $L = 6$. At this point, spending more components only gives marginal quality improvements.

As the last two rows in Table 7.1 indicates, the bound $G_{\max}$ is always maintained. The bound can be violated, when in addition to $G_{\max}$ a limit on L is imposed. For example, not allowing more than $L = 5$ for the bound $G_{\max} = 0.5$, results in a deviation $G_N(\underline{\eta}, \gamma) = 0.917$ for both component adding procedures. This deviation value is indeed larger than the bound. However, in many practical applications, keeping L below a given maximum number of components is of paramount importance for assuring worst-case computation time. Thus, with PGMR the user can set preferences on either a maximum deviation or a maximum number of components.

Table 7.1: Required number of components and reduction quality for different maximum deviation values $G_{\max}$.

$G_{\max}$	0.5	0.75	2	4
Number of components (splitting & insertion)	7	6	5	4
Deviation $G_N(\underline{\eta}, \gamma)$ (splitting)	0.217	0.234	0.917	3.228
Deviation $G_N(\underline{\eta}, \gamma)$ (insertion)	0.228	0.236	0.917	3.228

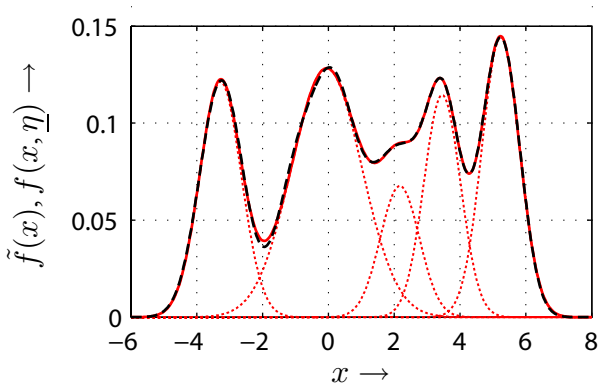


Figure 7.3: True Gaussian mixture (black, dashed) and reduced Gaussian mixture (red, solid) consisting of 5 components (red, dotted).

7.3.2 Comparison with State-of-the-art Methods

Now, the PGMR algorithm is compared with three established reduction methods. All three algorithms employ the common greedy approach of successively merging pairs of components to maintain the desired number of components. The first is Williams' reduction algorithm [193], which can be considered as global reduction approach. The second is a local reduction algorithm proposed by M. West [188], while the method of A. Runnalls [153] represents a compromise between local and global approaches. For a more detailed introduction to these algorithms and their classification into local and global reduction methods see Section 7.4.

Univariate Gaussian Mixture

At first, the Gaussian mixture of a scalar random variable $\boldsymbol{x}$ is considered for comparison purposes. The true mixture consists of $M \in \{40, 80, 120, 160, 200\}$, where the parameters are drawn i.i.d. from uniform distributions over the intervals

$$\omega \in [0.05, 0.5] \ , \qquad \hat{x} \in [0, 3] \ , \qquad \sigma \in [0.09, 0.5] \ .$$

For each number of components M, 20 Monte Carlo simulation runs are performed, where all reduction algorithms are forced to use $L = 10$ or less components. For PGMR, the bound $G_{\max} = 1$ is selected.

In Figure 7.4 (a) and (b), the average deviations and average computation times for all M are depicted.[4] Neither in deviation nor in computation time there is a significant difference between PGMR with splitting and PGMR with insertion. For both adding methods, PGMR provides the best average deviation for each M. This is notable, as PGMR on average uses between six and eight components, while Williams', West's, and Runnalls' algorithm always result in reduced mixtures with 10 components. In Figure 7.4 (c), the reduction results for a true mixture with $M = 200$ components are depicted. Thanks to the progressive processing, PGMR is capable of almost exactly capturing the shape of the true mixture, while Williams' algorithm fails in accurately approximating details of the shape as it can be clearly seen for the second mode. In this specific example, the reduction result of Runnalls' algorithms is comparable to Williams' and thus, is not depicted here for clarity reasons. The grossness of West's method is even more significant, as it does not incorporate any shape information when merging components. However, in contrast to PGMR, the three state-of-the-art algorithms preserve the mean and variance of the original mixture (see Section 7.4).

4 The computation times depend on a Matlab 7.5 implementation running on a PC with an Intel Core2 Duo 2.4 GHz processor.

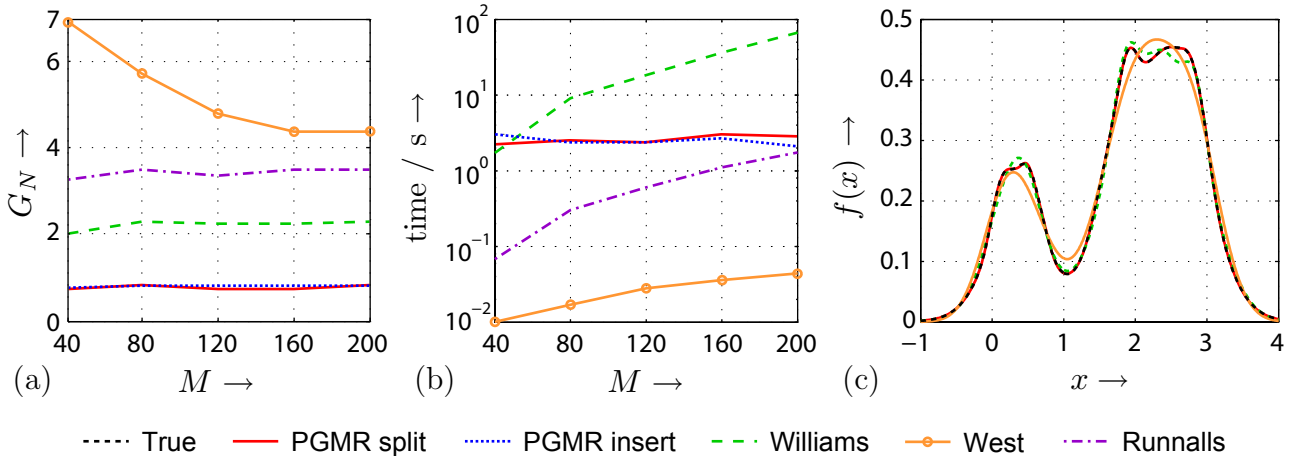


Figure 7.4: Reduction results for univariate Gaussian mixtures over 20 Monte Carlo simulation runs. (a) Average deviation, (b) average computation time, and (c) an exemplary 200 component Gaussian mixture and its reduced versions.

Figure 7.4 (b) indicates that the computation time of PGMR is approximately constant for all M, while it grows with M for the other algorithms. In case of West's algorithm, this growth is negligible, as the algorithm is generally computationally very efficient due to its local reduction characteristic.

The constant computation time of PGMR originates from the different way a mixture is reduced. Regardless of the number of components of $\tilde{f}(x)$, PGMR always starts with a single Gaussian. The computationally most expensive operations of PGMR are structural adaptations, i.e., extending the number of components. However, these operations are only performed if required and handling many components in $f(x;\underline{\eta})$ is systematically avoided. On the other side, all three state-of-the-art algorithms start the reduction with the complete original mixture and perform a greedy search involving all remaining components for identifying the next merging operation in each reduction step. This search basically has a quadratic complexity in case of Williams'[5] as well as Runnalls' algorithm and a linear complexity for West's method.

Bivariate Gaussian Mixture

In this simulation, randomly generated Gaussian mixtures representing a two-dimensional random vector $\underline{\boldsymbol{x}} \in \mathbb{R}^2$ are considered. The weighting coefficients ω and the elements c of the (non-diagonal) covariance matrices of the true mixtures are drawn from the intervals

$$\omega \in [0.05, 0.5] \,, \qquad\qquad c \in [0.1, 1] \,,$$

while 25% of the mean vectors are drawn from $\underline{\hat{x}} \in [0, 0.75] \times [0, 1.5]$ and the remaining mean vectors originate from $\underline{\hat{x}} \in [1.5, 2] \times [0, 1.5]$. Due to this placement of the Gaussian components, bimodality is forced in the true mixture. An exemplary true mixture with $M = 120$ components is depicted in Figure 7.5 (a). Again, 20 Monte Carlo simulation runs are performed for each number of components $M \in \{40, 80, 120, 160, 200\}$. The maximum number of components of the reduced mixture is set to $L = 10$, and the maximum allowed deviation for PGMR is $G_{\max} = 2$.

The average deviation is depicted in Figure 7.5 (b). As in the univariate case, PGMR provides the best reduction quality, independent of the used adding technique; splitting and

5 As suggested in [193], the used implementation of Williams' reduction method exploits that all terms for calculating the measure (7.4) can be pre-computed and stored. Because only a few terms change between several reduction steps, partially updating the stored terms leads to significant computational savings.

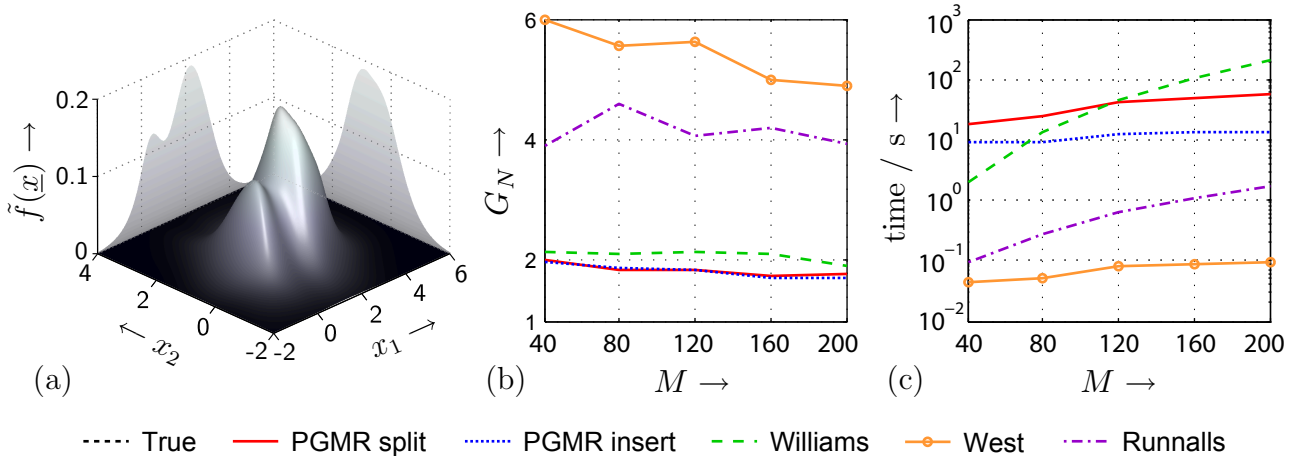


Figure 7.5: Reduction results for bivariate Gaussian mixtures. (a) Exemplary true Gaussian mixture with 120 components. (b) Average deviation and (c) average computation time over 20 Monte Carlo simulation runs.

insertion perform almost equivalent. The results obtained from Williams' reduction algorithm are close to PGMR. But again, Williams always ends up with 10 components for the reduced mixture, while both PGMR methods merely require between seven and nine components on average. It is further worth mentioning that PGMR performs best even though axis-aligned mixtures are used for approximation purposes.

More severe are the differences between both PGMR methods with regard to the computation times, as shown in Figure 7.5 (c). Due to the computationally more demanding distance measure evaluations for multivariate Gaussian mixtures, the computation time is larger for all reduction methods compared to the univariate case. However, the characteristics of the growth with regard to the number of components M of the true mixture are still the same, except for PGMR with splitting. Now, the time grows linearly with M, mainly for two reasons. First, splitting scales with the dimension of the state-space and is conceptually more complex than performing an insertion due to the necessity of determining the principal axis for splitting. Second, as discussed in Example 7.1, splitting typically leads to a slower convergence of the additional correction step, which in turn results in a longer computation time. This effect is exacerbated in case of multivariate mixtures. Since splitting is only performed along principal axes, the newly added component in some cases may be relatively far away from the point of strongest deviation. This in turn leads to more extensive corrections. In extreme cases, additional splittings have to be performed, if the additional correction step cannot fully adjust the new Gaussian towards the strongest deviation.

7.4 Contrast to Prior Work

State-of-the-art mixture reduction algorithms and PGMR differ in many aspects. In Table 7.2, the most important ones are summarized. For a detailed discussion, a short overview of existing reduction algorithms is given at first. Therefore, these algorithms are classified into two groups, namely into *local* and *global* algorithms, depending on the optimization approach and the information on the true mixture that is incorporated for reducing a Gaussian mixture.

Local Mixture Reduction

Local algorithms only consider lower-order statistics of the mixture like mean and variance or completely disregard the overall effect on the true mixture, e.g., by evaluating only the

Table 7.2: Comparison of classical Gaussian mixture reduction algorithms with PGMR with respect to the optimization approach for approximation, reduction principle, computational complexity, user constraints, type of the reduced mixture, and preservation of moments.

	Classical approaches	*PGMR*
Optimization	mainly local	global
Principle	top-down	bottom-up
Complexity	depends on components	depends mainly on shape
Constraints	number of components	number of components or maximum deviation
Reduced mixture	general	axis-aligned
Moment-preserving	yes	no

similarity between components. Salmond's joining and clustering algorithm [155, 157] as well as West's algorithm [188] are part of this group. These algorithms involve finding the pair (Salmond's joining and West's algorithm) or the group (Salmond's clustering algorithm) of mixture components that are closest based on the weighted form of the Mahalanobis distance measure [120]. These Gaussians are merged and the process is repeated until the reduced mixture contains the desired number of components. Instead of the Mahalanobis distance measure, the so-called iterative pairwise replacement algorithm (IPRA) [166, 174] employs a weighted form of the Hellinger distance [19] for evaluating the similarity between components.

Global Mixture Reduction

On the other hand, the measure employed in *global* methods considers all available information, i.e., shape information of the mixture, for reduction purposes. Williams' reduction algorithm [193] employs the squared integral measure (7.4) to evaluate at each reduction step which particular deletion of a component or merge of a pair of components yields the smallest dissimilarity from the true Gaussian mixture. Compared to local methods, the reduction results of global methods are typically more accurate, at the expense of a higher computational effort for reduction. One way to benefit from both reduction approaches is to evaluate a localized version of a global measure as done in Runnalls' algorithms [153], where a computationally cheap upper bound of the Kullback-Leibler divergence (2.30) is minimized.

Reduction Principle

All classical approaches operate with the same top-down principle. They start the reduction with the complex Gaussian mixture and reduce the number of components by successively merging pairs or groups of Gaussians. For obtaining a specific approximation quality, merging approaches often end up with a number of components that is still too large, as the inherent redundancy of the original mixture is exploited only in a greedy fashion. PGMR instead employs a principle dual to existing algorithms, where an approximate mixture is generated in a bottom-up fashion by progressively incorporating the true Gaussian mixture. For tracking the deviation between the true and the reduced mixture during the progression, the same distance measure as used for Williams' algorithm is employed. Accordingly, PGMR is also part of the group of global reduction algorithms. But in contrast to Williams' algorithm, which only tries to minimize the growth of the approximation error, PGMR exploits the distance measure for improving the approximation capabilities of the reduced mixture. Furthermore, new components are only added when necessary in order to keep the number of components in the reduced mixture as small as possible.

Complexity

Due to the constructive nature of PGMR, where new components are added at points of strong deviations, its computational complexity mainly depends on the complexity of the true mixture's shape. Hence, the more complex the shape of the true mixture is, e.g., the more modes the mixture has, the more computationally expensive operations for structural adaptation have to be performed in order to adapt the reduced mixture to emerging structural changes during the progression. The number of components of the true mixture instead has only a minor effect on the overall runtime, as demonstrated in the simulations in Section 7.3.2. This is different from the classical approaches. Due to the successive merging of components, the shape of the true mixture has almost no effect on the computation time.

The behavior of PGMR is particularly beneficial in scenarios, where mixtures with simple shapes are represented by many Gaussian components. For instance, target tracking applications as considered for example in Section 5.5.3 or Section 6.3.3 typically lead to unimodal density functions and thus cause low runtimes.

One drawback of PGMR compared to classical approaches is a stronger dependence on the dimension of the state space. This is due to the fact that the number of parameters required for describing the reduced mixture grows linearly with the dimension. The matrices that are involved in performing the predictor and correction steps in turn depend quadratically on the number of mixture parameters. This dependence is more severe for PGMR with splitting, as shown in the simulations. Attenuating this dependence will further increase the practicability of the PGMR approach and is part of future work.

Constraints

Classical approaches always stop the reduction process once the desired number of components is achieved, regardless of the resulting reduced mixture is a good approximation of the true mixture or not. This drawback is abolished by PGMR. Here, maintaining an user-defined maximum deviation value is guaranteed during the progression. Additionally, a maximum number of components can be defined by the user. In contrast to classical approaches, the user can balance between reduction quality and computation time.

Reduced Mixture Type

Despite that PGMR merely employs axis-aligned Gaussians for representing the reduced mixture, it is able to outperform classical reduction algorithms, which utilize general Gaussian mixtures for approximation. But the more complex the shape of the true Gaussian mixture, e.g., the more irregular and rough the shape is, the more the approximation quality of all reduction algorithms degrades. This loss in performance is currently more severe for PGMR, because of the employed axis-aligned Gaussians, which offer less approximation capabilities. Hence, for accurately representing specific regions of the true mixture, more components for the reduced mixture are necessary as it would be the case if general Gaussian components were used. In some pathological cases no reduction of the number of mixture components is possible. The extension of PGMR to utilize general Gaussian mixtures for approximation is part of future work.

Moment-preserving Reduction

Merging of Gaussians in classical reduction algorithms is always performed under the preservation of the mean and covariance of the original mixture. PGMR instead does not provide an exact preservation of these moments. However, thanks to the shape approximation, the

deviation between the moments of the original mixture and the reduced mixture is typically small.

7.5 Summary

To achieve the goal of replacing a complex Gaussian mixture by one consisting of a minimal number of components with respect to a desired maximum reduction error, the classical approach of successively merging components is often inappropriate. In comparison, the novel progressive Gaussian mixture reduction algorithm introduced in this chapter provides significantly better reduction results. It was demonstrated that gradually incorporating the effect of the complex true Gaussian mixture during the progression facilitates accurate approximations as the reduced Gaussian mixture can be constantly adapted and, if required, its approximation capability in specific regions can be improved by adding new components. Compared to local reduction methods, the resulting reduced mixture is very close to the original since adapting the approximation is accomplished by a global optimization. Compared to other global approaches, redundancy in the original mixture is better exploited. Thus, the used number of components and the computational demand is significantly smaller.

Future work is devoted to increase the practicability of PGMR. Currently, the complexity of PGMR grows with the dimension of the state space as well as with the number of components of the reduced mixture due to the Newton method utilized in the correction steps. Explicitly determining the occurring matrices can for example be avoided by employing quasi-Newton methods like BFGS (see e.g. [61]) or by employing matrix-free Newton methods like Jacobian-free Newton-GMRES [98]. Employing these techniques would also mark an important step towards the extension of PGMR for constructing a general instead of an axis-aligned Gaussian mixture for reduction purposes.

Furthermore, an exact preservation of mean and covariance of the original mixture is currently not possible. It is intended to incorporate the true moments as further constraints. By applying the Lagrangian multiplier approach, especially the mean constraint can then be maintained during the progression.

CHAPTER 8

Conclusions and Future Work

In this thesis, various techniques for nonlinear non-Gaussian sensor management and state estimation are developed in order to form a versatilely applicable probabilistic framework for sensor management. The main challenges arise from the circumstances that both theories building the foundation of the framework, i.e., stochastic control and Bayesian state estimation, are of conceptual value only for the assumed properties of the considered system and sensor models. More precisely, to maximize the utility of the given sensors and their sensing modalities, optimization over multiple time steps ahead needs to be performed. This requires to anticipate, quantify, and incorporate the effect of future and thus unavailable information about sensor measurements onto the sensing decisions, whereas the number of potential decisions growth exponentially with the length of the time horizon. In addition, the consideration of potential decisions repeatedly results in recursively estimating and inferring the temporal evolution of the observed nonlinear system. Coping with these challenges necessitates the development of systematic, accurate, and efficient approximations for feasible sensor management. Applications for the proposed framework especially arise in scenarios, where the information gathering process needs to be controlled in order to gain accurate estimates even for hardware constrained sensors, as it is often the case in sensor networks (processing power, energy) and for mobile sensor platforms (maneuverability).

8.1 Summary of Contributions

As aforementioned, sensor management can be considered as extended Bayesian state estimation task, where the sensors and their sensing modalities are carefully configured in order to maximize the information about the observed object. For both aspects, sensor management and state estimation, theoretical contributions are presented in this thesis.

Sensor Management

For sensor management purposes, two algorithms with different scope have been proposed. The *quasi-linear sensor manager* realizes an open-loop model predictive control scheme and is aimed at computationally restricted applications with mild nonlinearities. For strongly nonlinear and non-Gaussian scenarios on the other hand, the *information theoretic sensor manager* calculates the sensor configurations in a closed-loop model predictive control fashion. The theoretical contributions for realizing both sensor management algorithms are:

- Conversion of the formerly nonlinear non-Gaussian sensor management problem into a linear Gaussian one by means of statistical linearization. This kind of linearization allows incorporating uncertainty over the system state, which is of paramount importance due to the predictive application of the linearization for conversion.

- Efficient calculation of the optimal configuration sequence for the linear Gaussian sensor management problem by means of the novel *information-based pruning* algorithm. This optimal pruning algorithm exploits the sensor information matrix for an early exclusion of suboptimal configuration sequences from the tree search.

- Determination of so-called *virtual measurement* values for incorporating and anticipating future measurements. On the one hand, generating few but meaningful virtual measurement values reduces the computational complexity of the information theoretic sensor manager, while on the other hand specific characteristics of the measurement density, such as skewness or multimodality can be exploited.

- Backward recursion scheme employing optimal pruning for information theoretic sensor management. The novel probabilistic branch-and-bound pruning algorithm is aware of the probabilistic nature of the virtual measurement values.

- Tight and computationally cheap lower and upper bounds on the differential entropy for Gaussian mixture random vectors. These bounds allow efficiently approximating the mutual information, which is utilized as objective function for information theoretic sensor management.

The effectiveness of the proposed sensor management algorithms has been demonstrated in simulations concerning the sensor scheduling problem and the mobile sensor control problem for target localization and tracking. Due to the canonical structure of these problems, similar results can be expected for other sensor management problems.

State Estimation

The foundation of the proposed probabilistic sensor management framework is built by three approximate but computationally efficient state estimation techniques: the *Gaussian estimator*, the *hybrid density filter* (HDF), and the *progressive Gaussian mixture reduction* (PGMR) algorithm. The main theoretical contributions concerning these estimation techniques are:

- Deterministic sampling of Gaussian densities under consideration of the first two moments as well as the shape of the corresponding cumulative distribution function. This sampling scheme is employed in the quasi-linear sensor management approach for statistical linearization, whereas the number of regression points can be adapted depending on the desired accuracy and computational complexity.

- Approximating the nonlinear system and sensor models by considering the corresponding probabilistic representation in terms of conditional densities, i.e, transition density and likelihood. For the scalar case, the optimal approximation can be obtained in closed-form, while the multivariate case requires suboptimal approximation. The "curse of dimensionality", i.e., exponential growth with increasing dimension of the state space can be attenuated by decomposition in linear/nonlinear states as well as by decomposition in observed/unobserved states.

- Constructive (bottom-up) approach for Gaussian mixture reduction based on homotopy continuation for an improved exploitation of redundancies in Gaussian mixtures with many components. The continuation is controlled by a predictor-corrector scheme, where new components are added to the reduced mixture only when necessary in order to improve the approximation capabilities.

The application of the state estimation techniques is not only restricted to the proposed sensor management framework. The necessity of Bayesian state estimation arises in many practical applications, for instance in signal processing, machine learning, or, as demonstrated in many simulations within this thesis, in localization and tracking tasks.

8.2 Outlook to Future Work

Potential extensions to the various algorithms and approaches introduced in this thesis have been discussed in the summary sections at the end of each chapter. In the following, some long term developments in sensor management are identified that are worthy for further investigation.

Distributed Sensor Management

Even if not explicitly mentioned, the methods proposed in this thesis are restricted to sensor systems, where the sensor manager is a central component, i.e., the sensor configurations are always transmitted from and all measurement values are send back to a single sink. In large-scale multi-sensor systems such as sensor networks, it is desirable to perform sensor management in a distributed fashion. Distributed sensor management is advantageous over a central control and processing structure for many reasons. It allows reducing communication overheads and improves the flexibility, reliability, and scalability of the sensor system. The main challenge here is that determining optimal sensor configurations requires global knowledge, while the nodes of a sensor network often only possess local knowledge. Furthermore, due to its unreliability, the (wireless) communication channel has to be taken into account when calculating the sensor configuration, since packets containing the configuration vector, state estimate, or measurement values may be dropped. Only few approaches like those in [58, 122, 148] have been presented so far that deal with this problem, where admittedly many strongly simplifying assumptions are made. One step toward distributed sensor management could be to employ submodular objective functions [104, 192], which provide a tight bound on the estimation performance in case of greedy/myopic sensor management. As demonstrated in [220], myopic sensor management in turn simplifies a distributed implementation as only local knowledge is necessary.

Constrained Sensor Management

The objective functions employed in this thesis, namely covariance-based and information theoretic functions, are aimed in minimizing the uncertainty over the state estimate. In many applications, the sensor system is constrained for example with respect to energy consumption or communication radius. By extending the objective function and the optimization, trading off estimation performance and *explicitly* maintaining the constraints is possible. The main challenge herein is to still determine the optimal configuration sequence in a tractable way. Especially with regard to a versatilely applicable management framework, the solution techniques, e.g., pruning methods, further have to conserve the special structure of the given sensor management application.

Continuous-valued Configurations

For many sensor management problems like sensor scheduling, the set of possible configurations is discrete and finite. In case of continuous-valued configurations, employing discretization may lead to a finite set that suffices for adapting the given sensor system. However, the obtained

discretized set may be too large such that determining the next configuration becomes computationally intractable. Instead, directly manipulating the continuous-valued configurations could be advantageous, whereas simple enumeration as in the discrete case is now not longer possible. One solution approach is to combine gradient descent and potential field techniques [128, 173] with sensor management techniques as in [111, 122]. However, such an approach suffers from neglecting long-term effects resulting in local minima, which becomes even more severe in case of constrained sensor systems. Thus, further efforts have to be made for sensor management with continuous configurations over long time horizons.

Management of Dynamic Sensors

In many sensor management applications, the sensors possess an internal sensor state as stated in Section 2.1.3. Depending on the sensor dynamics affecting the sensor state, the performance of the proposed information-based pruning algorithm degrades. But also the bounds on the estimation performance that can be stated for myopic sensor management with submodular objective functions are not longer valid. The problem of sensor management with dynamic sensors becomes even more severe in cases where the sensor state is uncertain as well, i.e., evolves according to a stochastic process. Such a situation for example arises for the mobile sensor control problem when the position of the mobile is not known for certain. Here, extensions of the proposed pruning algorithm and the derivation of new bounds for myopic management in case of dynamic sensors are worth for further research.

Simultaneous Dynamic System Control and Sensor Management

Another interesting extension of the proposed sensor management framework would be to not only configure the sensors, but also to control the dynamic behavior of the observed system. An exemplary application for such a simultaneous dynamic system control and sensor management arises in cooperative mobile robotics [29], where a team or a swarm of mobile robots cooperatively performs environmental exploration [151] or object transportation [83]. Here, a robot typically has uncertain information about the positions of its team members. The robots need to adapt their motion in order to obtain or provide valuable position information, which is essential for accomplishing the given task. By combining the proposed framework with the model-predictive control techniques proposed in [187], a simultaneous treatment of both stochastic control tasks is possible. Due to the fact that additional decision layers need to be considered, the main challenge is to achieve a computationally feasible combined manager and controller component.

APPENDIX A

Density Function Representations

In the following, the definition and some important properties of a Gaussian density function is given. Furthermore, two special cases, namely axis-aligned Gaussians and Dirac Delta distributions are also described. Gaussian mixtures, which are convex combinations of Gaussian densities are subsequently introduced in Section A.2. Gaussian mixtures play an important role for precise density function approximation and state estimation, since all estimators described in Section 2.3 can also be applied on Gaussian mixtures by component-wise evaluation of the corresponding estimator equations.

A.1 Gaussian Density

For a Gaussian random vector $\underline{\boldsymbol{x}} \in \mathbb{R}^n$, the corresponding multivariate density function is defined according to

$$f(\underline{x}) = \mathcal{N}(\underline{x}; \underline{\hat{x}}, \mathbf{C}) = \frac{1}{\sqrt{|2\pi\mathbf{C}|}}\, e^{-\frac{1}{2}(\underline{x}-\underline{\hat{x}})^{\mathrm{T}}\mathbf{C}^{-1}(\underline{x}-\underline{\hat{x}})} \,, \tag{A.1}$$

with mean vector $\underline{\hat{x}}$ and symmetric, positive semi-definite covariance matrix $\mathbf{C}$.

A.1.1 Properties

Thus, a Gaussian density is uniquely defined by its first two moments. All higher-order moments can be calculated on basis of mean and covariance, whereby uncorrelatedness of elements of $\underline{x}$ automatically implies stochastical independence of these elements. In case of uncorrelated elements, the corresponding entries of the covariance matrix are zero.

Since the covariance matrix is symmetric and positive semi-definite, the amount of uncertainty characterized by means of the matrix can be interpreted geometrically as *covariance ellipsoid*. The exponent $d = (\underline{x} - \underline{\hat{x}})^{\mathrm{T}}\mathbf{C}^{-1}(\underline{x} - \underline{\hat{x}})$ of (A.1), which is also called the *Mahalanobis distance* [120], coincides with the general equation of an ellipsoid. The eigenvalues of $\mathbf{C}$ define the length of the principal axes and the eigenvectors define the orientation of the ellipsoid. The mean vector $\underline{\hat{x}}$ represents the center of the ellipsoid.

Multiplication or convolution of two Gaussian densities results again in a Gaussian density. Specifically, multiplying the two Gaussians $f_1(\underline{x}) = \mathcal{N}(\underline{x}; \underline{\hat{x}}_1, \mathbf{C}_1)$ and $f_2(\underline{x}) = \mathcal{N}(\underline{x}; \underline{\hat{x}}_2, \mathbf{C}_2)$ leads to

$$f_1(\underline{x}) \cdot f_2(\underline{x}) = c \cdot \mathcal{N}(\underline{x}; \underline{\hat{x}}, \mathbf{C}) \,, \tag{A.2}$$

with

$$\begin{aligned}\mathbf{C} &= (\mathbf{C}_1^{-1} + \mathbf{C}_2^{-1})^{-1} \ , \\ \underline{\hat{x}} &= \mathbf{C} \cdot (\mathbf{C}_1^{-1}\underline{\hat{x}}_1 + \mathbf{C}_2^{-1}\underline{\hat{x}}_2) \ , \\ c &= \mathcal{N}(\underline{\hat{x}}_1; \underline{\hat{x}}_2, \mathbf{C}_1 + \mathbf{C}_2) \ ,\end{aligned}$$

which is an unnormalized Gaussian density.

A.1.2 Special Case: Axis-aligned Gaussian

If all elements $\boldsymbol{x}_i$ of $\underline{\boldsymbol{x}}$ are uncorrelated, i.e., the covariance matrix is a diagonal matrix according to $\mathbf{C} = \operatorname{diag}([\sigma_1^2, \sigma_2^2, \ldots, \sigma_n^2])$ with $\sigma_i^2 = \operatorname{Cov}\{\boldsymbol{x}_i, \boldsymbol{x}_i\}$, the principal axes of the covariance ellipsoid are axis-aligned. The corresponding density function can be decomposed dimension by dimension according to

$$f(\underline{x}) = \mathcal{N}(\underline{x}; \underline{\hat{x}}, \mathbf{C}) = \mathcal{N}(x_1; \hat{x}_1, \sigma_1^2) \cdot \mathcal{N}(x_2; \hat{x}_2, \sigma_2^2) \cdots \mathcal{N}(x_n; \hat{x}_n, \sigma_n^2) \ .$$

A.1.3 Special Case: Dirac Delta Distribution

A further interesting special case of a Gaussian density is given by the Dirac delta distribution $\delta(\underline{x} - \underline{\hat{x}})$, which is the limit

$$\delta(\underline{x} - \underline{\hat{x}}) = \lim_{|\mathbf{C}| \to 0} \mathcal{N}(\underline{x}; \underline{\hat{x}}, \mathbf{C}) = \begin{cases} 0 & , \ (\underline{x} - \underline{\hat{x}})^{\mathrm{T}}(\underline{x} - \underline{\hat{x}}) \neq 0 \\ \text{undefined} & , \ \text{otherwise} \end{cases} \ .$$

Thus, the value of the Dirac delta distribution is everywhere zero, except at its location $\underline{\hat{x}}$. This fact leads to the so-called *sifting property*

$$\int_{\mathbb{R}^n} f(\underline{x}) \cdot \delta(\underline{x} - \underline{\hat{x}}) \, \mathrm{d}\underline{x} = f(\underline{\hat{x}}) \ .$$

Thanks to the Dirac delta distribution, a unified treatment of discrete and continuous random variables is possible [141].

A.2 Gaussian Mixture

For accurately representing non-Gaussian density functions like multimodal or skewed densities, Gaussian mixtures are a popular density type. As the name implies, Gaussian mixtures are a weighted sum of L Gaussian densities according to

$$f(\underline{x}) = \sum_{i=1}^{L} \omega_i \cdot \mathcal{N}(\underline{x}; \underline{\hat{x}}_i, \mathbf{C}_i) \ ,$$

where ω_i are the non-negative weighting coefficients. With $\sum_i \omega_i = 1$ it is ensured that the probability mass of the Gaussian mixture is equal one.

A.2.1 Properties

Gaussian mixtures are a universal function approximator in that, given a sufficient number of Gaussian components, they can approximate any smooth function to arbitrary accuracy [124]. If merely axis-aligned Gaussian components are used, the resulting mixture is called analogously *axis-aligned Gaussian mixture*. This special mixture type still is a universal function

approximator, but the approximation capabilities are reduced compared to a general Gaussian mixture, which means that typically more components are necessary for obtaining are comparable approximation quality.

Given the parameters of a Gaussian mixture, i.e., weights, mean vectors, and covariance matrices of the Gaussian components, the mean vector and covariance matrix of the mixture can be calculated according to

$$\hat{\underline{x}} = \sum_{i=1}^{L} \omega_i \cdot \hat{\underline{x}}_i \tag{A.3}$$

and

$$\mathbf{C} = \sum_{i=1}^{L} \omega_i \cdot \left(\mathbf{C}_i + \hat{\underline{x}}_i \cdot \hat{\underline{x}}_i^{\mathrm{T}} \right) - \hat{\underline{x}} \cdot \hat{\underline{x}}^{\mathrm{T}} \ , \tag{A.4}$$

respectively.

A.2.2 Special Case: Dirac Mixture

If all mixture components of a Gaussian mixture are given by a Dirac delta distribution, the resulting density function is called a Dirac mixture

$$f(\underline{x}) = \sum_{i=1}^{L} \omega_i \cdot \delta(\underline{x} - \hat{\underline{x}}_i) \ .$$

This special case of a Gaussian mixture is also a universal function approximator. However, due to the point location of the probability masses of the Dirac delta distributions, representing continuous functions requires an infinite number of mixture components. Mean vector and covariance matrix of a Dirac mixture are given by

$$\hat{\underline{x}} = \sum_{i=1}^{L} \omega_i \cdot \hat{\underline{x}}_i$$

and

$$\mathbf{C} = \sum_{i=1}^{L} \omega_i \cdot \hat{\underline{x}}_i \cdot \hat{\underline{x}}_i^{\mathrm{T}} - \hat{\underline{x}} \cdot \hat{\underline{x}}^{\mathrm{T}} \ ,$$

respectively.

APPENDIX B

Analytic Expressions for PGMR

In this section, the analytic expressions for all relevant terms of the progressive Gaussian mixture reduction approach are provided. These are the coefficients $\mathbf{P}(\underline{\eta}, \gamma)$ and $\underline{b}(\underline{\eta}, \gamma)$ of the system of ODEs (7.6) as well as the gradient $\underline{g}(\underline{\eta}, \gamma)$, which is part of the predictor-corrector scheme.

B.1 Analytical Expression for $\mathbf{P}(\underline{\eta}, \gamma)$

At first, the solution of the first summand in (7.7) is derived, which is

$$\mathbf{P}'(\underline{\eta}) = \int_{\mathbb{R}^{n_x}} \underline{F}(\underline{x}; \underline{\eta}) \cdot \underline{F}(\underline{x}; \underline{\eta})^{\mathrm{T}} \, \mathrm{d}\underline{x} = \begin{bmatrix} \mathbf{P}^{(1,1)} & \mathbf{P}^{(1,2)} & \dots & \mathbf{P}^{(1,L)} \\ \mathbf{P}^{(2,1)} & \mathbf{P}^{(2,2)} & \dots & \mathbf{P}^{(2,L)} \\ \vdots & \vdots & & \vdots \\ \mathbf{P}^{(L,1)} & \mathbf{P}^{(L,2)} & \dots & \mathbf{P}^{(L,L)} \end{bmatrix} .$$

The individual $(2n_x + 1) \times (2n_x + 1)$ block matrices $\mathbf{P}^{(i,j)}$ for $i = 1, \dots, L$ and $j = 1, \dots, L$ are

$$\begin{aligned} \mathbf{P}^{(i,j)} &= \int_{\mathbb{R}^{n_x}} \frac{\partial f_i(\underline{x}; \underline{\eta}_i)}{\partial \underline{\eta}_i} \cdot \left(\frac{\partial f_j(\underline{x}; \underline{\eta}_j)}{\partial \underline{\eta}_j} \right)^{\mathrm{T}} \mathrm{d}\underline{x} \\ &= \omega_i^2 \cdot \omega_j^2 \cdot \mathcal{N}\left(\underline{\hat{x}}_i; \underline{\hat{x}}_j, \mathbf{C}_i + \mathbf{C}_j\right) \cdot \begin{bmatrix} \frac{4}{\omega_i \cdot \omega_j} & \left(\underline{P}_1^{(i,j)}\right)^{\mathrm{T}} & \left(\underline{P}_2^{(i,j)}\right)^{\mathrm{T}} & \dots & \left(\underline{P}_{n_x}^{(i,j)}\right)^{\mathrm{T}} \\ \underline{P}_1^{(j,i)} & \mathbf{P}_1^{(i,j)} & \mathbf{P}_{1,2}^{(i,j)} & \dots & \mathbf{P}_{1,n_x}^{(i,j)} \\ \underline{P}_2^{(j,i)} & \mathbf{P}_{2,1}^{(j,i)} & \mathbf{P}_2^{(i,j)} & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \mathbf{P}_{n_x-1,n_x}^{(i,j)} \\ \underline{P}_{n_x}^{(j,i)} & \mathbf{P}_{n_x,1}^{(j,i)} & \dots & \mathbf{P}_{n_x,n_x-1}^{(j,i)} & \mathbf{P}_{n_x}^{(i,j)} \end{bmatrix} , \end{aligned}$$

with $\underline{\eta}_i = [\omega_i, \underline{\eta}_{i,1}^{\mathrm{T}}, \underline{\eta}_{i,2}^{\mathrm{T}}, \dots, \underline{\eta}_{i,n_x}^{\mathrm{T}}]^{\mathrm{T}}$, $\underline{\eta}_{i,n} = [\hat{x}_{i,n}, \sigma_{i,n}]^{\mathrm{T}}$, and $f_i(\underline{x}; \underline{\eta}_i) := \omega_i^2 \cdot \mathcal{N}(\underline{x}; \underline{\hat{x}}_i, \mathbf{C}_i)$. Furthermore, the matrices $\mathbf{P}^{(i,j)}$ consist of the vectors

$$\underline{P}_n^{(i,j)} = \begin{bmatrix} \frac{2}{\omega_i} \cdot \frac{\hat{x}_{i,n} - \hat{x}_{j,n}}{\sigma_{i,j,n}^2} & \frac{2\sigma_{j,n}}{\omega_i} \cdot \frac{(\hat{x}_{i,n} - \hat{x}_{j,n})^2 - \sigma_{i,j,n}^2}{\sigma_{i,j,n}^4} \end{bmatrix}^{\mathrm{T}} ,$$

which comprise the products of the derivative with respect to the weighting coefficient ω_i and the derivatives with respect to the parameter vector $\underline{\eta}_{j,n}$ of the j-th Gaussian component. Here,

$\sigma_{i,j,n}^2 = \sigma_{i,n}^2 + \sigma_{j,n}^2$. The 2×2 matrix

$$\mathbf{P}_n^{(i,j)} = \begin{bmatrix} \frac{\sigma_{i,j,n}^2 - (\hat{x}_{i,n} - \hat{x}_{j,n})^2}{\sigma_{i,j,n}^4} & \frac{\sigma_{j,n} \cdot (\hat{x}_{j,n} - \hat{x}_{i,n}) \cdot \left((\hat{x}_{i,n} - \hat{x}_{j,n})^2 - 3\sigma_{i,j,n}^2 \right)}{\sigma_{i,j,n}^6} \\ \frac{\sigma_{i,n} \cdot (\hat{x}_{i,n} - \hat{x}_{j,n}) \cdot \left((\hat{x}_{j,n} - \hat{x}_{i,n})^2 - 3\sigma_{i,j,n}^2 \right)}{\sigma_{i,j,n}^6} & \frac{\sigma_{i,n} \cdot \sigma_{j,n} \cdot \left((\hat{x}_{i,n} - \hat{x}_{j,n})^4 + 3\sigma_{i,j,n}^2 \left(\sigma_{i,j,n}^2 - 2(\hat{x}_{i,n} - \hat{x}_{j,n})^2 \right) \right)}{\sigma_{i,j,n}^8} \end{bmatrix}$$

comprises the products of the derivatives with respect to the parameter vector $\underline{\eta}_{i,n}$ and the derivatives with respect to the parameter vector $\underline{\eta}_{j,n}$, while the 2×2 matrix

$$\mathbf{P}_{n,m}^{(i,j)} = \begin{bmatrix} \frac{(\hat{x}_{i,n} - \hat{x}_{j,n}) \cdot (\hat{x}_{i,m} - \hat{x}_{j,m})}{\sigma_{i,j,n}^2 \cdot \sigma_{i,j,m}^2} & \frac{\sigma_{j,m} \cdot (\hat{x}_{i,n} - \hat{x}_{j,n}) \cdot \left(\sigma_{i,j,m}^2 - (\hat{x}_{i,m} - \hat{x}_{j,m})^2 \right)}{\sigma_{i,j,n}^2 \cdot \sigma_{i,j,m}^4} \\ \frac{\sigma_{j,n} \cdot (\hat{x}_{i,m} - \hat{x}_{j,m}) \cdot \left(\sigma_{i,j,n}^2 - (\hat{x}_{i,n} - \hat{x}_{j,n})^2 \right)}{\sigma_{i,j,m}^2 \cdot \sigma_{i,j,n}^4} & \frac{\sigma_{i,n} \cdot \sigma_{j,m} \cdot \left(\sigma_{i,j,n}^2 - (\hat{x}_{i,n} - \hat{x}_{j,n})^2 \right) \cdot \left(\sigma_{i,j,m}^2 - (\hat{x}_{i,m} - \hat{x}_{j,m})^2 \right)}{\sigma_{i,j,n}^4 \cdot \sigma_{i,j,m}^4} \end{bmatrix}$$

comprises the derivatives with respect to the parameter vector $\underline{\eta}_{i,n}$ and the derivatives with respect to the parameter vector $\underline{\eta}_{j,m}$.

The expression for $\Delta\mathbf{P}(\underline{\eta}, \gamma)$ is given by

$$\Delta\mathbf{P}(\underline{\eta}, \gamma) = \int_{\mathbb{R}^{n_x}} \left(f(\underline{x}; \underline{\eta}) - \tilde{f}(\underline{x}; \gamma) \right) \cdot \mathbf{M}(\underline{x}, \underline{\eta}) \, \mathrm{d}\underline{x} \,, \tag{B.1}$$

where

$$\mathbf{M}(\underline{x}, \underline{\eta}) = \frac{\partial^2 f(\underline{x}; \underline{\eta})}{\partial \underline{\eta} \, \partial \underline{\eta}^{\mathrm{T}}} = \begin{bmatrix} \mathbf{M}^{(1)} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{M}^{(2)} & & \mathbf{0} \\ \vdots & & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{M}^{(L)} \end{bmatrix} ,$$

with $(2n_x + 1) \times (2n_x + 1)$ block matrices

$$\mathbf{M}^{(i)} = 2 \cdot f_i(\underline{x}; \underline{\eta}_i) \cdot \begin{bmatrix} \frac{1}{\omega_i^2} & \left(\underline{M}_1^{(i)} \right)^{\mathrm{T}} & \left(\underline{M}_2^{(i)} \right)^{\mathrm{T}} & \cdots & \left(\underline{M}_{n_x}^{(i)} \right)^{\mathrm{T}} \\ \underline{M}_1^{(i)} & \mathbf{M}_1^{(i)} & \mathbf{M}_{1,2}^{(i)} & \cdots & \mathbf{M}_{1,n_x}^{(i)} \\ \underline{M}_2^{(i)} & \mathbf{M}_{2,1}^{(i)} & \mathbf{M}_2^{(i)} & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \mathbf{M}_{n_x-1,n_x}^{(i)} \\ \underline{M}_{n_x}^{(i)} & \mathbf{M}_{n_x,1}^{(i)} & \cdots & \mathbf{M}_{n_x,n_x-1}^{(i)} & \mathbf{M}_{n_x}^{(i)} \end{bmatrix} , \tag{B.2}$$

where

$$\underline{M}_n^{(i)} = \begin{bmatrix} \frac{x_n - \hat{x}_{i,n}}{\omega_i \sigma_{i,n}^2} & \frac{(x_n - \hat{x}_{i,n})^2 - \sigma_{i,n}^2}{\omega_i \sigma_{i,n}^3} \end{bmatrix}^{\mathrm{T}} ,$$

$$\mathbf{M}_n^{(i)} = \begin{bmatrix} \frac{(x_n - \hat{x}_{i,n})^2 - \sigma_{i,n}^2}{2\sigma_{i,n}^4} & \frac{(x_n - \hat{x}_{i,n})^3 - 3\sigma_{i,n}^2 (x_n - \hat{x}_{i,n})}{2\sigma_{i,n}^5} \\ \frac{(x_n - \hat{x}_{i,n})^3 - 3\sigma_{i,n}^2 (x_n - \hat{x}_{i,n})}{2\sigma_{i,n}^5} & \frac{(x_n - \hat{x}_{i,n})^4 - 5\sigma_{i,n}^2 (x_n - \hat{x}_{i,n})^2 + 2\sigma_{i,n}^4}{2\sigma_{i,n}^6} \end{bmatrix} ,$$

$$\mathbf{M}_{n,m}^{(i)} = \begin{bmatrix} \frac{x_n - \hat{x}_{i,n}}{\sigma_{i,n}^2} \frac{x_m - \hat{x}_{i,m}}{\sigma_{i,m}^2} & \frac{x_n - \hat{x}_{i,n}}{\sigma_{i,n}^2} \frac{(x_m - \hat{x}_{i,m})^2 - \sigma_{i,m}^2}{\sigma_{i,m}^3} \\ \frac{(x_n - \hat{x}_{i,n})^2 - \sigma_{i,n}^2}{\sigma_{i,n}^3} \frac{x_m - \hat{x}_{i,m}}{\sigma_{i,m}^2} & \frac{(x_n - \hat{x}_{i,n})^2 - \sigma_{i,n}^2}{\sigma_{i,n}^3} \frac{(x_m - \hat{x}_{i,m})^2 - \sigma_{i,m}^2}{\sigma_{i,m}^3} \end{bmatrix} .$$

It is important to note that the matrices (B.2) are symmetric. Furthermore, solving (B.1) corresponds to the calculation of the zeroth up to the forth moment of the Gaussian mixtures $f(\underline{x}; \underline{\eta}) \cdot f_i(\underline{x}; \underline{\eta}_i)$ and $\tilde{f}(\underline{x}; \gamma) \cdot f_i(\underline{x}; \underline{\eta}_i)$. This can be done similarly as it shown in the following for $\underline{b}(\underline{\eta}, \gamma)$.

B.2 Analytical Expression for $\underline{b}(\underline{\eta}, \gamma)$

The expression for the vector

$$\underline{b}(\underline{\eta}, \gamma) = \int_{\mathbb{R}^{n_x}} \underline{F}(\underline{x}; \underline{\eta}) \frac{\partial \tilde{f}(\underline{x}; \gamma)}{\partial \gamma} \, \mathrm{d}\underline{x}$$

consists of the vector of partial derivatives $\underline{F}(\underline{x}; \underline{\eta})$, which comprises the elements

$$\frac{\partial f(\underline{x}; \underline{\eta})}{\partial \underline{\eta}_i} = f_i(\underline{x}; \underline{\eta}_i) \cdot \begin{bmatrix} \frac{2}{\omega_i} \\ \underline{b}_{i,1} \\ \vdots \\ \underline{b}_{i,n_x} \end{bmatrix} ,$$

where

$$\underline{b}_{i,n} = \begin{bmatrix} \frac{x_n - \hat{x}_{i,n}}{\sigma_{i,n}^2} & \frac{(x_n - \hat{x}_{i,n})^2 - \sigma_{i,n}^2}{\sigma_{i,n}^3} \end{bmatrix}^{\mathrm{T}} .$$

Together with the scalar function

$$\frac{\partial \tilde{f}(\underline{x}; \gamma)}{\partial \gamma} = \tilde{f}(\underline{x}) - \hat{f}(\underline{x}) ,$$

the elements of $\underline{b}(\underline{\eta}, \gamma)$ comprising the derivatives with respect to $\underline{\eta}_i$ are given by

$$\begin{aligned} \underline{b}_i(\underline{\eta}_i, \gamma) &= \int_{\mathbb{R}^{n_x}} \left(\tilde{f}(\underline{x}) - \hat{f}(\underline{x}) \right) \cdot f_i(\underline{x}; \underline{\eta}_i) \cdot \begin{bmatrix} \frac{2}{\omega_i} \\ \underline{b}_{i,1} \\ \vdots \\ \underline{b}_{i,n_x} \end{bmatrix} \mathrm{d}\underline{x} \\ &= \underbrace{\begin{bmatrix} \underline{B}^{(i)} & \underline{0}^{\mathrm{T}} & \cdots & \cdots & \underline{0}^{\mathrm{T}} \\ \underline{0} & \mathbf{B}_1^{(i)} & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \mathbf{0} & \mathbf{B}_2^{(i)} & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \mathbf{0} \\ \underline{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{B}_{n_x}^{(i)} \end{bmatrix}}_{=: \mathbf{B}^{(i)}} \cdot \begin{bmatrix} \mathrm{E}_{\tilde{\mathcal{F}}^{(i)}}\{1\} - \mathrm{E}_{\hat{\mathcal{F}}^{(i)}}\{1\} \\ \mathbf{E}_1^{(i)} \\ \mathbf{E}_2^{(i)} \\ \vdots \\ \mathbf{E}_{n_x}^{(i)} \end{bmatrix} , \end{aligned} \tag{B.3}$$

where

$$\begin{aligned} \underline{B}^{(i)} &= \begin{bmatrix} \frac{2}{\omega_i} & 0 & 0 \end{bmatrix} , \\ \mathbf{B}_n^{(i)} &= \begin{bmatrix} -\frac{\hat{x}_{i,n}}{\sigma_{i,n}^2} & \frac{1}{\sigma_{i,n}^2} & 0 \\ \frac{\hat{x}_{i,n}^2 - \sigma_{i,n}^2}{\sigma_{i,n}^3} & -\frac{2\hat{x}_{i,n}}{\sigma_{i,n}^3} & \frac{1}{\sigma_{i,n}^3} \end{bmatrix} , \\ \mathbf{E}_n^{(i)} &= \begin{bmatrix} \mathrm{E}_{\tilde{\mathcal{F}}^{(i)}}\{x_n\} - \mathrm{E}_{\hat{\mathcal{F}}^{(i)}}\{x_n\} \\ \mathrm{E}_{\tilde{\mathcal{F}}^{(i)}}\{x_n^2\} - \mathrm{E}_{\hat{\mathcal{F}}^{(i)}}\{x_n^2\} \end{bmatrix} . \end{aligned}$$

Thus, $\underline{b}(\underline{\eta}, \gamma)$ can be efficiently calculated using matrix-vector calculus, where the vector comprises the zeroth up to the second moment of the densities $\tilde{\mathcal{F}}^{(i)}(\underline{x}) = \tilde{f}(\underline{x}) \cdot f_i(\underline{x}; \underline{\eta}_i)$ and $\hat{\mathcal{F}}^{(i)}(\underline{x}) = \hat{f}(\underline{x}) \cdot f_i(\underline{x}; \underline{\eta}_i)$. All moments can be determined in closed form and $\mathrm{E}_{\tilde{\mathcal{F}}^i}$ as well as $\mathrm{E}_{\hat{\mathcal{F}}^i}$ are the corresponding expected value operators.

B.3 Analytical Expression for $\underline{g}(\underline{\eta}, \gamma)$

The gradient $\underline{g}(\underline{\eta}, \gamma)$ of the squared integral distance measure comprises the elements

$$\underline{g}_i(\underline{\eta}_i, \gamma) = -\int_{\mathbb{R}^{n_x}} \left(\tilde{f}(\underline{x}; \gamma) - f(\underline{x}; \underline{\eta}) \right) \cdot \frac{\partial f(\underline{x}; \underline{\eta})}{\partial \underline{\eta}_i} \, \mathrm{d}\underline{x} \, , \tag{B.4}$$

for $i = 1, 2, \ldots, L$, which are quite similar to (B.3). Hence, (B.4) can also be written in matrix-vector notation

$$\underline{g}_i(\underline{\eta}_i, \gamma) = \mathbf{B}^{(i)} \cdot \begin{bmatrix} \mathrm{E}_{\mathcal{F}^{(i)}}\{1\} - \mathrm{E}_{\tilde{\mathcal{F}}^{(i)}}\{1\} \\ \mathbf{E}_1^{(i)} \\ \mathbf{E}_2^{(i)} \\ \vdots \\ \mathbf{E}_{n_x}^{(i)} \end{bmatrix} ,$$

with

$$\mathbf{E}_n^{(i)} = \begin{bmatrix} \mathrm{E}_{\mathcal{F}^{(i)}}\{x_n\} - \mathrm{E}_{\tilde{\mathcal{F}}^{(i)}}\{x_n\} \\ \mathrm{E}_{\mathcal{F}^{(i)}}\{x_n^2\} - \mathrm{E}_{\tilde{\mathcal{F}}^{(i)}}\{x_n^2\} \end{bmatrix}$$

and $\mathcal{F}^{(i)}(\underline{x}) = f(\underline{x}; \underline{\eta}) \cdot f_i(\underline{x}; \underline{\eta}_i)$, $\tilde{\mathcal{F}}^{(i)}(\underline{x}) = \tilde{f}(\underline{x}; \gamma) \cdot f_i(\underline{x}; \underline{\eta}_i)$.

Lists of Figures, Tables, Algorithms, and Examples

List of Figures

List of Tables

List of Algorithms

List of Examples

Bibliography

[1] K. M. Abadir and J. R. Magnus. *Matrix Algebra*. Cambridge University, 2005.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A Survey on Sensor Networks. *IEEE Communications Magazine*, 40(8):102–114, Aug. 2002.

[3] E. L. Allgower and K. Georg. *Numerical Continuation Methods: An Introduction*. Springer Series in Computational Mathematics. Springer-Verlag, 1990.

[4] P. Alriksson and A. Rantzer. Sub-Optimal Sensor Scheduling with Error Bounds. In *Proceedings of the 16th IFAC World Congress*, Prague, Czech Republic, July 2005.

[5] D. L. Alspach and H. W. Sorenson. Nonlinear Bayesian Estimation using Gaussian Sum Approximation. *IEEE Transactions on Automatic Control*, 17(4):439–448, Aug. 1972.

[6] M. S. Andersland. Why Open-Loop LQG Measurement Scheduling is Optimal. In *Proceedings of the American Control Conference*, volume 3, pages 2110–2112, June 1995.

[7] I. Arasaratnam, S. Haykin, and R. J. Elliott. Discrete-Time Nonlinear Filtering Algorithms Using Gauss–Hermite Quadrature. *Proceedings of the IEEE*, 95(5):953–977, 2007.

[8] G. Arfken and H. J. Weber. *Mathematical Methods for Physicists*. Elsevier Academic Press, 6th edition, 2005. 173–174 pp.

[9] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, Feb. 2002.

[10] K. J. Astrom. Optimal control of Markov decision processes with incomplete state estimation. *Journal of Mathematical Analysis and Applications*, 10:174–205, 1965.

[11] A. C. Atkinson, A. N. Donev, and R. D. Tobias. *Optimum Experimental Designs, with SAS*. Oxford University Press, 2007.

[12] J. M. Aughenbaugh and B. R. LaCour. Measurement prioritization for optimal Bayesian fusion. In *Proceedings of the 10th International Conference on Information Fusion (Fusion)*, 2007.

[13] A. J. Baker. *Finite Element Computational Fluid Mechanics*. Taylor and Francis, 1983.

[14] Y. Bar-Shalom and X.-R. Li. *Multitarget-multisensor Tracking: Principles and Techniques*. YBS Publishing, Storrs, CT, 1995.

[15] M. J. Beal. *Variational Algorithms for Approximate Bayesian Inference*. PhD thesis, Gatsby Computational Neuroscience Unit, University College London, 2003.

[16] R. E. Bellman. On the Theory of Dynamic Programming. *Proceedings of the National Academy of Sciences*, 38:716–719, 1952.

[17] R. E. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[18] A. Bensoussan. *Stochastic Control of Partially Observable Systems*. Cambridge University Press, 1992.

[19] R. Beran. Minimum Hellinger Distance Estimates for Parametric Models. *The Annals of Statistics*, 5(3):445–463, 1977.

[20] N. Bergman, A. Doucet, and N. Gordon. Optimal estimation and Cramér-Rao bounds for partial non-Gaussian state space models. *Annals of the Institute of Statistical Mathematics*, 53(1):97–112, 2001.

[21] D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 1. Athena Scientific, Belmont, Massachusetts, U.S.A., 2nd edition, 2000.

[22] F. Beutler and U. D. Hanebeck. The Probabilistic Instantaneous Matching Algorithm. In *Proceedings of the 2006 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 311–316, Heidelberg, Germany, Sept. 2006.

[23] O. Bochardt, R. Calhoun, J. K. Uhlmann, and S. J. Julier. Generalized Information Representation and Compression Using Covariance Union. In *Proceedings of the 9th International Conference on Information Fusion (Fusion)*, Florence, Italy, July 2006.

[24] D. D. Boos. Minimum Distance Estimators for Location and Goodness of Fit. *Journal of the American Statistical association*, 76(375):663–670, 1981.

[25] G. E. P. Box, J. S. Hunter, and W. G. Hunter. *Statistics for Experiments: Design, Innovation, and Discovery*. Wiley & Sons, 2nd edition, 2005.

[26] D. Brigo and F. LeGland. A Finite Dimensional Filter with Exponential Density. In *Proceedings of the 1997 IEEE Conference on Decision and Control (CDC), San Diego, CA*, volume 2, pages 1643–1644, 1997.

[27] A. Brooks, A. Makarenko, S. Williams, and H. Durrant-Whyte. Parametric POMDPs for planning in continuous state spaces. *Robotics and Autonomous Systems*, 54(11):887–897, Nov. 2006.

[28] R. S. Bucy. Bayes Theorem and Digital Realizations for Non-Linear Filters. *Journal of Astronautical Sciences*, 17:80–94, 1969.

[29] Y. U. Cao, A. S. Fukunaga, and A. B. Kahng. Cooperative Mobile Robotics: Antecedents and Directions. *Autonomous Robots*, 4(1):7–27, Oct. 1997.

[30] J. Carpenter, P. Clifford, and P. Fearnhead. Improved particle filter for nonlinear problems. In *IEE Proceedings Radar, Sonar and Navigation*, volume 146, pages 2–7, Feb. 1999.

[31] M. Á. Carreira-Perpiñán. Mode-finding for mixtures of Gaussian distributions. Technical Report CS–99–03, Department of Computer Science, University of Sheffield, UK, 1999.

[32] M. A. Carreira-Perpinán. Mode-Finding for Mixtures of Gaussian Distribution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1318–1323, Nov. 2000.

[33] D. A. Castañón. Approximate Dynamic Programming For Sensor Management. In *Proceedings of the 36th Conference on Decision & Control (CDC)*, pages 1202–1207, San Diego, California, Dec. 1997.

[34] D. A. Castañón and L. Carin. *Foundations and Applications of Sensor Management*, chapter 2. Springer, Berlin, 2007.

[35] D. E. Catlin. *Estimation, Control, and the Discrete Kalman Filter*, volume 71 of *Applied Mathematical Sciences*. New York: Springer-Verlag, 1st edition, 1989.

[36] S. Challa, Y. Bar-Shalom, and V. Krishnamurthy. Nonlinear Filtering via Generalized Edgeworth Series and Gauss–Hermite Quadrature. *IEEE Transactions on Signal Processing*, 48(6):1816–1820, 2000.

[37] L. Chen and R. K. Mehra. Reformulating Kalman Filter Based Optimal Dynamic Coverage Control. In *Proceedings of the 17th IFAC World Congress*, pages 4174–4179, Seoul, Korea, July 2008.

[38] Y. Chen, K. L. Moore, and Z. Song. Diffusion boundary determination and zone control via mobile actuator-sensor networks. In *Proceedings of the SPIE Defense and Security Symposium on Intelligent Computing: Theory and Applications II (OR53)*, Orlando, FL, USA, Apr. 2004.

[39] R. C. H. Cheng and N. A. K. Amin. Estimating Parameters in Continuous Univariate Distributions with a Shifted Origin. *Journal of the Royal Statistical Society: Series B*, 45 (3):394–403, 1983.

[40] A. S. Chhetri, D. Morrell, and A. Papandreou-Suppappola. Scheduling multiple sensors using paricle filters in target tracking. In *Proceedings of IEEE Workshop on Statistical Signal Processing*, pages 549–552, St. Louis, Missouri, Sept. 2003.

[41] A. S. Chhetri, D. Morrell, and A. Papandreou-Suppappola. Energy Efficient Target Tracking in a Sensor Network Using Non-myopic Sensor Scheduling. In *Proceedings of the 7th International Conference on Information Fusion (Fusion)*, volume 1, pages 558–565, 2005.

[42] A. S. Chhetri, D. Morrell, and A. Papandreou-Suppappola. Nonmyopic Sensor Scheduling and its Efficient Implementation for Target Tracking Applications. *EURASIP Journal on Applied Signal Processing*, 2006:1–18, 2006.

[43] M. Chu, H. Haussecker, and F. Zhao. Scalable Information-driven Sensor Querying and Routing for ad hoc Heterogeneous Sensor Networks. *International Journal of High Performance Computing Applications*, 16(3):293–313, 2002.

[44] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active Learning with Statistical Models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

[45] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., 1991.

[46] D. Culler, D. Estrin, and M. Srivastava. Overview of Sensor Networks. *IEEE Computer*, 37(8):41–49, Aug. 2004.

[47] T. B. Curtin, J. G. Bellingham, J. Catipovic, and D. Webb. Autonomous oceanographic sampling network. *Oceanography*, 6(3):86–94, 1993.

[48] F. Daum. Nonlinear Filters: Beyond the Kalman Filter. *IEEE Aerospace and Electronic Systems Magazine*, 20(8):57–69, Aug. 2005.

[49] F. Daum and J. Huang. Curse of Dimensionality and Particle Filters. In *Proceedings of the 2003 IEEE Aerospace Conference*, volume 4, pages 1979–1993, Mar. 2003.

[50] N. de Freitas. Unscented Particle Filter. [Online]. http://www.cs.ubc.ca/∼nando.

[51] J. Denzler and C. Brown. Optimal Selection of Camera Parameters for State Estimation of Static Systems: An Information Theoretic Approach. Technical Report TR-732, Computer Science Department, University of Rochester, 2000.

[52] J. Denzler and C. M. Brown. Information Theoretic Sensor Data Selection for Active Object Recognition and State Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):145–157, Feb. 2002.

[53] A. Dhariwal, B. Zhang, B. Stauffer, C. Oberg, G. S. Sukhatme, D. A. Caron, and A. A. G. Requicha. Networked aquatic microbial observing system. In *Proceeding of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–19, Orlando, FL, May 2006.

[54] A. Doucet, N. de Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice*. Statistics for Engineering and Information Science. New York: Springer-Verlag, 2001.

[55] A. Doucet, N. de Freitas, and V. Krishnamurthy. Particle filters for state estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing*, 49(3):613–624, 2001.

[56] A. Drake. *Observation of a Markov process through a noisy channel*. PhD thesis, Massachusetts Institute of Technology, 1962.

[57] E. Ertin, J. W. Fisher, and L. C. Potter. Maximum Mutual Information Principle for Dynamic Sensor Query Problems. In *Information Processing in Sensor Networks (IPSN)*, pages 405–416, 2003.

[58] R. Evans, V. Krishnamurthy, G. Nair, and L. Sciacca. Networked Sensor Management and Date Rate Control for Tracking Maneuvering Targets. *IEEE Transactions on Signal Processing*, 53(6):1979–1991, June 2005.

[59] J. W. Fisher and T. Darrell. Speaker Association With Signal-Level Audiovisual Fusion. *IEEE Transactions on Multimedia*, 6(3):406–413, June 2004.

[60] J. W. Fisher III and J. C. Principe. A methodology for information theoretic feature extraction. In A. Stuberud, editor, *Proceedings of the IEEE International Joint Conference on Neural Networks*, volume 3, pages 1712–1716, 1998.

[61] R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, 2nd edition, May 2000.

[62] A. Gilpin and T. Sandholm. A competitive Texas Hold'em poker via automated abstraction and real-time equilibrium computation. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Boston, MA, 2006.

[63] J. Goldberger, S. Gordon, and H. Greenspan. An Efficient Image Similarity Measure based on Approximations of KL-Divergence Between Two Gaussian Mixtures. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, volume 1, pages 487–493, Oct. 2003.

[64] B. Grocholsky, H. Durrant-Whyte, and P. Gibbens. An Information-Theoretic Approach to Decentralized Control of Multiple Autonomous Flight Vehicles. In *Proceedings of SPIE*, volume 4196 of *Sensor Fusion and Decentralized Control*, Nov. 2000.

[65] B. Grocholsky, A. Makarenko, and H. Durrant-Whyte. Information-Theoretic Coordinated Control of Multiple Sensor Platforms. In *Proceedings of the 2003 IEEE International Conference on Robotics & Automation (ICRA)*, pages 1521–1526, Taipei, Taiwan, Sept. 2003.

[66] V. Gupta, T. H. Chung, B. Hassibi, and R. M. Murray. Sensor Scheduling Algorithms Requiring Limited Computations. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2004.

[67] V. Gupta, T. H. Chung, B. Hassibi, and R. M. Murray. On a Stochastic Sensor Selection Algorithm with Applications in Sensor Scheduling and Sensor Coverage. *Automatica*, 42 (2):251–260, Feb. 2006.

[68] V. Gupta, D. E. Jeffcoat, and R. Murray. On Sensor Coverage by Mobile Sensors. In *Proceedings of the 45th IEEE Conference on Decision & Control (CDC)*, pages 5912–5917, San Diego, California, Dec. 2006.

[69] P. Hall. On Kullback-Leibler Loss and Density Estimation. *The Annals of Statistics*, 15 (4):1491–1519, 1987.

[70] U. D. Hanebeck. *Nonlinear Methods for State Estimation in Stochastic Dynamical Systems – A Concise Introduction*. 2002.

[71] U. D. Hanebeck, K. Briechle, and A. Rauh. Progressive Bayes: A New Framework for Nonlinear State Estimation. In *Proceedings of SPIE, AeroSense Symposium*, volume 5099, pages 256–267, Orlando, Florida, May 2003.

[72] U. D. Hanebeck and V. Klumpp. Localized Cumulative Distributions and a Multivariate Generalization of the Cramér-von Mises Distance. In *Proceedings of the 2008 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, Seoul, Republic of Korea, Aug. 2008.

[73] U. D. Hanebeck and O. C. Schrempf. Greedy Algorithms for Dirac Mixture Approximation of Arbitrary Probability Density Functions. In *Proceedings of the 2007 IEEE Conference on Decision and Control (CDC)*, pages 3065–3071, New Orleans, Louisiana, Dec. 2007.

[74] A. Hanselmann, O. C. Schrempf, and U. D. Hanebeck. Optimal Parametric Density Estimation by Minimizing an Analytic Distance Measure. In *Proceedings of the 10th International Conference on Information Fusion (Fusion)*, Quebec, Canada, July 2007.

[75] T. Hanselmann, M. Moreland, B. Moran, and P. Sarunic. Sensor scheduling for multiple target tracking and detection using passive measurements. In *Proceedings of the 11th International Conference on Information Fusion (Fusion)*, pages 1528–1535, Cologne, Germany, 2008.

[76] V. Hasselblad. Estimation of Parameters for a Mixture of Normal Distributions. *Technometrics*, 8(8):431–444, Aug. 1966.

[77] M. Hauskrecht. Value-function Approximations for Partially Observable Markov Decision Processes. *Journal of Artificial Intelligence Research*, 13:33–94, Aug. 2000.

[78] S. Haykin. *Communication Systems*. John Wiley & Sons, 4th edition, 2001.

[79] M. L. Hernandez. Optimal Sensor Trajectories in Bearings-Only Tracking. In *Proceedings of the 7th International Conference on Information Fusion (Fusion)*, pages 893–900, 2004.

[80] M. L. Hernandez, T. Kirubarajan, and Y. Bar-Shalom. Multisensor Resource Deployment using Posterior Cramer-Rao Bounds. *IEEE Transactions on Aerospace and Electronic Systems*, 40(2):399–416, 2004.

[81] J. R. Hershey and P. A. Olsen. Approximating the Kullback-Leibler Divergence between Gaussian Mixture Models. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, pages IV–317–IV–320, Apr. 2007.

[82] S. Howard, S. Suvorova, and B. Moran. Optimal Policy for Scheduling of Gauss-Markov Systems. In *Proceedings of the 7th International Conference on Information Fusion (Fusion)*, pages 888–892, Stockholm, Sweden, 2004.

[83] T. L. Huntsberger, A. Trebi-Ollennu, H. Aghazarian, P. S. Schenker, P. Pirjanian, and H. D. Nayar. Distributed control of multi-robot systems engaged in tightly coupled tasks. *Autonomous Robots*, 17(1):79–92, July 2004.

[84] I. I. Hussein. A Kalman filter-based control strategy for dynamic coverage control. In *Proceedings of the 2007 American Control Conference (ACC)*, pages 3271–3276, New York, New York, July 2007.

[85] K. Ito and K. Xiong. Gaussian Filters for Nonlinear Filtering Problems. *IEEE Transactions on Automatic Control*, 45(5):910–927, May 2000.

[86] A. J. Izenman. Recent developments in nonparametric density estimation. *Journal of the American Statistical Association*, 86(413):205–224, Mar. 1991.

[87] A. Jeremić and A. Nehorai. Landmine Detection and Localization Using Chemical Sensor Array Processing. *IEEE Transactions on Signal Processing*, 48(5):1295–1305, May 2000.

[88] S. Joshi and S. Boyd. Sensor Selection via Convex Optimization. *IEEE Transactions on Signal Processing*, 57(2):451–462, Feb. 2009.

[89] S. J. Julier. A skewed approach to filtering. In *Signal and Data Processing of Small Targets*, volume 3373, pages 271–282. SPIE, 1998.

[90] S. J. Julier. The Scaled Unscented Transform. In *Proceedings of the American Control Conference (ACC)*, pages 4555–4559, Anchorage, Alaska, May 2002.

[91] S. J. Julier and J. K. Uhlmann. A New Extension of the Kalman Filter to Nonlinear Systems. In *International Symposium on Aerospace/Defence Sensing, Simulation and Control*, 1997.

[92] S. J. Julier and J. K. Uhlmann. Unscented Filtering and Nonlinear Estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004.

[93] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[94] S. Kagami and M. Ishikawa. A Sensor Selection Method Considering Communication Delays. In *Proceedings of the 2004 IEEE International Conference on Robotics & Automation (ICRA)*, New Orleans, LA, Apr. 2004.

[95] T. Kailath, A. H. Sayed, and B. Hassibi. *Linear Estimation*. Prentice Hall, 2nd edition, 2000.

[96] R. E. Kalman. A new Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME, Journal of Basic Engineering*, 82 (Series D)(1):35–45, 1960.

[97] V. Klumpp and U. D. Hanebeck. Dirac Mixture Trees for Fast Suboptimal Multi-Dimensional Density Approximation. In *Proceedings of the 2008 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, Seoul, Republic of Korea, Aug. 2008.

[98] D. A. Knoll and D. E. Keyes. Jacobian-free Newton-Krylov methods: a survey of approaches and applications. *Journal of Computational Physics*, 193(2):357–397, Jan. 2004.

[99] W. Koch. Advanced Sensor and Dynamics Models with an Application to Sensor Management. In V. Kordic, editor, *Advances in Sensor Data Fusion*, chapter 12. IN-TECH Publishers, Apr. 2009.

[100] J. H. Kotecha and P. M. Djurić. Gaussian Particle Filtering. *IEEE Transactions on Signal Processing*, 51(10):2592–2601, 2003.

[101] J. H. Kotecha and P. M. Djurić. Gaussian Sum Particle Filtering. *IEEE Transactions on Signal Processing*, 51(10):2602–2612, 2003.

[102] S. C. Kramer and H. W. Sorenson. Recursive Bayesian estimation using piece-wise constant approximations. *Automatica*, 24:789–801, 1988.

[103] A. Krause and C. Guestrin. Near-optimal Nonmyopic Value of Information in Graphical Models. In *Proceedings of Uncertainty in Artificial Intelligence (UAI)*, July 2005.

[104] A. Krause and C. Guestrin. Near-optimal Observation Selection using Submodular Functions. In *Proceedings of 22nd Conference on Artificial Intelligence (AAAI)*, 2007.

[105] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg. Near-optimal Sensor Placements: Maximizing Information while Minimizing Communication Cost. In *Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN)*, Nashville, Tennessee, Apr. 2006.

[106] T. P. Krekeler. Multivariate Dichteapproximation mit Dirac Mixtures (Multivariate Density Approximation using Dirac Mixtures). Diploma thesis, Universität Karlsruhe (TH), 2007.

[107] C. Kreucher, A. O. Hero, III, K. Kastella, and D. Chang. Efficient Methods of Non-myopic Sensor Management for Multitarget Tracking. In *Proceedings of the 43rd IEEE Conference on Decision & Control (CDC)*, Atlantis, Paradise Island, Bahamas, Dec. 2004.

[108] C. Kreucher, K. Kastella, and A. O. Hero, III. Information Based Sensor Management for Multitarget Tracking. In *The 2003 Workshop on Multiple Hypothesis Tracking: A Tribute to Samuel S. Blackman*, pages 77–81, May 2003.

[109] C. Kreucher, K. Kastella, and A. O. Hero, III. Sensor management using an active sensing approach. *Signal Processing*, 85(3):607–624, Mar. 2005.

[110] C. M. Kreucher and A. O. Hero, III. Non-myopic approaches to scheduling agile sensors for multitarget detection, tracking, and identification. In *Proceedings of the 2005 IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages V–885–V–888, Philadelphia, PA, Mar. 2005.

[111] C. M. Kreucher, A. O. Hero, III, K. D. Kastella, and M. R. Moreland. An Information-Based Approach to Sensor Management in Large Dynamic Networks. *Proceedings of the IEEE*, 95(5):978–999, May 2007.

[112] V. Krishnamurthy. Algorithms for Optimal Scheduling and Management of Hidden Markov Model Sensors. *IEEE Transactions on Signal Processing*, 50(6):1382–1397, June 2002.

[113] S. Kullback and R. A. Leibler. On Information and Sufficiency. *Annals of Mathematical Statistics*, 22(2):79–86, 1951.

[114] T. Lefebvre, H. Bruyninckx, and J. D. Schutter. Comments on "A New Method for the Nonlinear Transformation of Means and Covariances in Filters and Estimators". *IEEE Transactions on Automatic Control*, 45(8):1406–1408, 2002.

[115] T. Lefebvre, H. Bruyninckx, and J. D. Schutter. *Nonlinear Kalman Filtering for Force-Controlled Robot Tasks*. Springer Berlin, 2005.

[116] J. M. Leiva-Murillo and A. Artés-Rodríguez. *Independent Component Analysis and Blind Signal Separation*, volume 3195, chapter A Gaussian Mixture Based Maximization of Mutual Information for Supervised Feature Extraction, pages 271–278. Springer Berlin / Heidelberg, 2004.

[117] D. Li, K. D. Wong, Y. H. Hu, and A. M. Sayeed. Detection, Classification, and Tracking of Targets. *IEEE Signal Processing Magazine*, 19(2):17–29, Mar. 2002.

[118] M. L. Littman, A. R. Cassandra, and L. P. Kaelbling. Efficient dynamic-programming updates in partially observable Markov decision processes. Technical Report CS-95-19, Brown University, 1995.

[119] A. Logothetis and A. Isaksson. On sensor scheduling via information theoretic criteria. *Proceedings of the 1999 American Control Conference (ACC)*, 4:2402–2406, 1999.

[120] P. C. Mahalanobis. On the generalised distance in statistics. *Proceedings of the National Institute of Science of India*, 12(1):49–55, 1936.

[121] J. Manyika and H. Durrant-Whyte. Information as a basis for management and control in decentralised fusion architectures. In *IEEE Conference on Decision and Control (CDC)*, 1992.

[122] G. M. Mathews. *Asynchronous Decision Making for Decentralised Autonomous Systems*. PhD thesis, Australian Centre for Field Robotics, The University of Sydney, Mar. 2008.

[123] P. Maybeck. *Stochastic Models, Estimation, and Control*. Academic Press, 1979.

[124] V. Maz'ya and G. Schmidt. On approximate approximations using gaussian kernels. *IMA J. Numer. Anal.*, 16:13–29, 1996.

[125] R. McLenaghan and C. Roberts. Continuous Mathematics. In D. Zwillinger, editor, *CRC Standard Mathematical Tables and Formulae*, pages 389–390. CRC Press, 2003.

[126] L. Meier, J. Peschon, and R. M. Dressler. Optimal Control of Measurement Subsystems. *IEEE Transactions on Automatic Control*, AC-12(5):528–536, Oct. 1967.

[127] J. V. Michalowicz, J. M. Nichols, and F. Bucholtz. Calculation of Differential Entropy for a Mixed Gaussian Distribution. *Entropy*, 10(3):200–206, Aug. 2008.

[128] F. Miyazaki and S. Arimoto. Sensory Feedback based on the Artifical Potential for Robots. In *Proceedings of the 9th IFAC World Congress*, pages 2381–2386, 1984.

[129] M. H. Moghari and P. Abolmaesumi. Comparing Unscented and Extended Kalman Filter Algorithms in the Rigid-Body Point-Based Registration. In *28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 497–500, 2006.

[130] K. L. Moore, Y. Chen, and Z. Song. Diffusion-based path planning in mobile actuator-sensor networks (MAS-net): Some preliminary results. In *Proceedings of the SPIE Defense and Security Symposium on Intelligent Computing: Theory and Applications II (OR53)*, Orlando, FL, USA, Apr. 2004.

[131] A. I. Mourikis and S. I. Roumeliotis. Optimal Sensing Strategies for Mobile Robot Formations: Resource-Constrained Localization. In *Proceedings of Robotics: Science and Systems*, pages 281–288, Boston, MA, June 2005.

[132] A. I. Mourikis and S. I. Roumeliotis. Optimal Sensor Scheduling for Resource-Constrained Localization of Mobile Robot Formations. *IEEE Transactions on Robotics*, 22(5):917–931, Oct. 2006.

[133] C. Musso, N. Oudjane, and F. LeGland. Improving Regularised Particle Filters. In A. Doucet, N. de Freitas, and N. Gordon, editors, *Sequential Monte Carlo Methods in Practice*, chapter 12. New York: Springer Verlag, 2001.

[134] A. G. Mutambara. *Decentralized Estimation and Control for Multisensor Systems*. CRC Press Inc., 1998.

[135] Y. Nakamura. *Data Fusion in Robotics and Machine Intelligence*, chapter Geometric Fusion: Minimizing Uncertainty Volumes, pages 457–480. Academic Press, 1992.

[136] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Number 63 in CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, 1992.

[137] M. Nørgaard, O. Ravn, and N. K. Poulsen. New developments in state estimation for nonlinear systems. *Automatica*, 36(11):1627–1638, Nov. 2000.

[138] Y. Oshman. Optimal Sensor Selection Strategy for Discrete-Time State Estimators. *IEEE Transactions on Aerospace and Electronic Systems*, 30(2):307–314, Apr. 1994.

[139] A. B. Owen. Monte Carlo Extensions of Quasi-Monte Carlo. In *Winter Simulation Conference Proceedings*, pages 571–577, 1998.

[140] C. H. Papadimitriou and J. N. Tsitsiklis. The Complexity of Markov Decision Processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.

[141] A. Papoulis and S. U. Pillai. *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, 4th edition, 2002.

[142] E. Parzen. On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, Sept. 1962.

[143] M. Patan and D. Uciński. Optimal scheduling of mobile sensor networks for detection and localization of stationary contamination sources. In *Proceedings of the 11th International Conference on Information Fusion (Fusion)*, pages 366–372, Cologne, Germany, 2008.

[144] A. Peñalver, J. M. Sáez, and F. Escolano. *Progress in Pattern Recognition, Speech and Image Analysis*, volume 2905, chapter An Entropy Maximization Approach to Optimal Model Selection in Gaussian Mixtures, pages 432–439. Springer Berlin / Heidelberg, 2003.

[145] V. Philomin, R. Duraiswami, and L. Davis. Quasi-Random Sampling for Condensation. In *Proceedings of the European Conference of Computer Vision*, volume 1843/2000, pages 134–149. Springer Berlin / Heidelberg, 2000.

[146] M. J. D. Powell. A Fortran Subroutine for Solving Systems of Nonlinear Algebraic Equations. In P. Rabinowitz, editor, *Numerical Methods for Nonlinear Algebraic Equations*, chapter 7. 1970.

[147] F. Pukelsheim. *Optimal Design of Experiments*. Cambridge University Press, 2006.

[148] Z. Quan, W. J. Kaiser, and A. H. Sayed. A Spatial Sampling Scheme Based on Innovations Diffusion in Sensor Networks. In *Proceedings of the Sixth International Conference on Information Processing in Sensor Networks (IPSN)*, pages 323–330, Cambridge, MA, Apr. 2007.

[149] E. Rafajlowicz. Optimum choice of moving sensor trajectories for distributed parameter system identification. *International Journal of Control*, 43(5):1441–1451, 1986.

[150] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[151] I. M. Rekleitis, G. Dudek, and E. E. Milios. Multi-robot collaboration for robust exploration. *Annals of Mathematics and Artificial Intelligence*, 31(1–4):7–40, 2001.

[152] S. I. Roumeliotis and G. A. Bekey. Distributed Multirobot Localization. *IEEE Transactions on Robotics and Automation*, 18(5):781–795, 2002.

[153] A. R. Runnalls. Kullback-Leibler Approach to Gaussian Mixture Reduction. *IEEE Transactions on Aerospace and Electronic Systems*, 43(3):989–999, July 2007.

[154] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2nd edition, international edition, 2003.

[155] D. J. Salmond. Mixture reduction algorithms for target tracking. In *IEE Colloquium on State Estimation in Aerospace and Tracking Applications*, pages 7/1–7/4, London, UK, Dec. 1989.

[156] D. J. Salmond. *Tracking in Uncertain Environments*. PhD thesis, Royal Aerospace Establishment, 1989.

[157] D. J. Salmond. Mixture reduction algorithms for target tracking in clutter. In *Proceedings of SPIE Signal and Data Processing of Small Targets*, volume 1305, pages 434–445, Oct. 1990.

[158] A. V. Savkin, R. J. Evans, and E. Skafidas. The Problem of Optimal Robust Sensor Scheduling. In *Proceedings of the 39th IEEE Conference on Decision & Control (CDC)*, volume 4, pages 3791–3796, Sydney, Australia, Dec. 2000.

[159] F. Sawo, K. Roberts, and U. D. Hanebeck. Bayesian Estimation of Distributed Phenomena using Discretized Representations of Partial Differential Equations. In *Proceedings of the 3rd International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, pages 16–23, Setúbal, Portugal, Aug. 2006.

[160] T. S. Schei. A finite difference method for linearizing in nonlinear estimation algorithms. *Automatica*, 33(11):2051–2058, Nov. 1997.

[161] T. Schön, F. Gustafsson, and P.-J. Nordlund. Marginalized Particle Filters for Mixed Linear/Nonlinear State-Space Models. *IEEE Transactions on Signal Processing*, 53(7): 2279–2287, July 2005.

[162] O. C. Schrempf, D. Brunn, and U. D. Hanebeck. Density Approximation Based on Dirac Mixtures with Regard to Nonlinear Estimation and Filtering. In *Proceedings of the 2006 IEEE Conference on Decision and Control (CDC)*, San Diego, California, Dec. 2006.

[163] O. C. Schrempf, D. Brunn, and U. D. Hanebeck. Dirac Mixture Density Approximation Based on Minimization of the Weighted Cramér-von Mises Distance. In *Proceedings of the 2006 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 512–517, Heidelberg, Germany, Sept. 2006.

[164] O. C. Schrempf and U. D. Hanebeck. Recursive Prediction of Stochastic Nonlinear Systems Based on Dirac Mixture Approximations. In *Proceedings of the 2007 American Control Conference (ACC)*, pages 1768–1774, New York, New York, July 2007.

[165] F. C. Schweppe. *Uncertain Dynamic Systems*. Prentice-Hall, 1973.

[166] D. W. Scott and W. F. Szewczyk. From kernels to mixtures. *Technometrics*, 43(3): 323–335, 2001.

[167] C. E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27(Part I):379–423, 1948.

[168] M. Simandl and J. Duník. Sigma point gaussian sum filter design using square root unscented filters. In *Proceedings of the 16th IFAC World Congress*, Prague, Czech Republic, July 2005.

[169] D. Simon. *Optimal State Estimation: Kalman, H-Infinity, and Nonlinear Approaches*. Wiley & Sons, 1 edition, 2006.

[170] S. S. Singh, B.-N. V. Nikolaos Kantas, A. Doucet, and R. J. Evans. Simulation-based optimal sensor scheduling with application to observer trajectory planning. *Automatica*, 43(5):817–830, May 2007.

[171] E. Skafidas and A. Nerode. Optimal Measurement Scheduling in Linear Quadratic Gaussian Control Problems. In *Proceedings of the 1998 IEEE International Conference on Control Applications (CCA)*, pages 1225–1229, Trieste, Italy, Sept. 1998.

[172] H. W. Sorenson. *Kalman Filtering: Theory and Application*. Piscataway, NJ: IEEE, 1985.

[173] W. M. Spears, D. F. Spears, J. C. Hamann, and R. Heil. Distributed, Physics-Based Control of Swarms of Vehicles. *Autonomous Robots*, 17(2–3), Nov. 2004.

[174] H. G. Sung. *Gaussian Mixture Regression and Classification*. PhD thesis, Rice University, Houston, Texas, May 2004.

[175] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[176] D. Tenne and T. Singh. The Higher Order Unscented Filter. In *Proceedings of the American Control Conference*, pages 2441–2446, June 2003.

[177] G. Terejanu, P. Singla, T. Singh, and P. D. Scott. A Novel Gaussian Sum Filter Method for Accurate Solution to the Nonlinear Filtering Problem. In *Proceedings of the 11th International Conference on Information Fusion (Fusion)*, Cologne, Germany, July 2008.

[178] D. Uciński. *Optimal Measurement Methods for Distributed Parameter System Identification*. CRC Press, 2000.

[179] D. Uciński. Optimal sensor location for parameter estimation of distributed processes. *International Journal of Control*, 73(13):1235–1248, 2000.

[180] R. van der Merwe and E. A. Wan. Efficient Derivative-Free Kalman Filters for Online Learning. In *Proceedings of the European Symposium on Artificial Neural Networks (ESANN)*, pages 205–210, Bruges, Belgium, Apr. 2001.

[181] T. Vercauteren and X. Wang. Decentralized Sigma-Point Information Filters for Target Tracking in Collaborative Sensor Networks. *IEEE Transactions on Signal Processing*, 53 (8):2997–3009, Aug. 2005.

[182] P. Viola and W. M. Wells III. Alignment by Maximization of Mutual Information. *International Journal of Computer Vision*, 24(2):137–154, 1997.

[183] E. A. Wan and R. van der Merwe. The Unscented Kalman Filter for Nonlinear Estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000*, pages 153–158, 2000.

[184] E. A. Wan and R. van der Merwe. The Unscented Kalman Filter. In S. Haykin, editor, *Kalman Filtering and Neural Networks*, Adaptive and Learning Systems for Signal Processing, Communication, and Control, chapter 7. Wiley & Sons, 2001.

[185] H. Wang, K. Yao, and D. Estrin. Information-theoretic approaches for sensor selection and placement in sensor networks for target localization and tracking. *Journal of Communications and Networks*, 7(4):481–491, 2005.

[186] H. Wang, K. Yao, G. Pottie, and D. Estrin. Entropy based Sensor Selection Heuristic for Target Localization. In *Proceedings of the Third Symposium on Information Processing in Sensor Netoworks (IPSN)*, pages 35–45, Berkeley, California, Apr. 2004.

[187] F. Weissel. *Stochastische modell-prädiktive Regelung nichtlinearer System (Stochastic Model Predictive Control of nonlinear Systems)*. PhD thesis, Universität Karlsruhe (TH), 2009.

[188] M. West. Approximating Posterior Distributions by Mixtures. *Journal of the Royal Statistical Society: Series B*, 55(2):409–422, 1993.

[189] J. L. Williams. *Information Theoretic Sensor Management*. PhD thesis, Massachusetts Institute of Technology, Feb. 2007.

[190] J. L. Williams, J. W. Fisher, III, and A. S. Willsky. Approximate Dynamic Programming for Communication-Constrained Sensor Network Management. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 55(8):4300–4311, Aug. 2007.

[191] J. L. Williams, J. W. Fisher, III, and A. S. Willsky. Performance Guarantees for Information Theoretic Active Inference. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, Mar. 2007.

[192] J. L. Williams, J. W. Fisher, III, and A. S. Willsky. Performance guarantees for information theoretic sensor resource management. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages III–933 – III–936, Apr. 2007.

[193] J. L. Williams and P. S. Maybeck. Cost-Function-Based Gaussian Mixture Reduction for Target Tracking. In *Proceedings of the 6th International Conference on Information Fusion (Fusion)*, volume 2, pages 1047–1054, 2003.

[194] W. Wu and A. Arapostathis. Optimal Control of Stochastic Systems with Costly Observations—The General Markovian Model and the LQG Problem. In *Proceedings of the 2005 American Control Conference (ACC)*, pages 294–299, Portland, Oregon, June 2005.

[195] W.-R. Wu. Target Tracking with Glint Noise. *IEEE Transactions on Aerospace and Electronic Systems*, 29(1):174–185, Jan. 1993.

[196] Y. Wu, D. Hu, M. Wu, and X. Hu. Unscented Kalman Filtering for Additive Noise Case: Augmented versus Nonaugmented. *IEEE Signal Processing Letters*, 12(5):357–360, May 2005.

[197] Y. Wu, M. Wu, D. Hu, and X. Hu. An Improvement to Unscented Transformation. In *AI 2004: Advances in Artifical Intelligence*, volume 3339/2004. Springer Berlin/Heidelberg, 2004.

[198] E. A. Yildirim. On the Minimum Volume Covering Ellipsoid of Ellipsoids. *SIAM Journal on Optimization*, 17(3):621–641, Sept. 2006.

[199] F. Zhao, J. Shin, and J. Reich. Information-Driven Dynamic Sensor Collaboration. *IEEE Signal Processing Magazine*, 19(2):61–72, Mar. 2002.

Supervised Student Theses

[200] C. Chlebek. Mehrschrittige nichtlineare Sensoreinsatzplanung zur Lokalisierung mobiler Roboter (Non-myopic Nonlinear Sensor Management for Mobile Robot Localization). Student research project, Intelligent Sensor-Actuator-Systems Laboratory, Universität Karlsruhe (TH), 2009.

[201] P. Ding. Erweiterung eines progressiven Verfahrens zur Reduktion multivariater Gaußmischdichten (Extension of a Progressive Gaussian Mixture Reduction Method to the Multivariate Case). Student research project, Intelligent Sensor-Actuator-Systems Laboratory, Universität Karlsruhe (TH), 2008.

[202] D. Itte. Mehrstufiges Clustering-Verfahren zur Komponentenreduktion von Gaußmischdichten (Multi-stage Clustering-based Approach to Gaussian Mixture Reduction). Student research project, Intelligent Sensor-Actuator-Systems Laboratory, Universität Karlsruhe (TH), 2009.

[203] A. Kuwertz. Verteilte Sensoreinsatzplanung zur modellbasierten Rekonstruktion räumlich ausgedehnter Phänomene (Distributed Sensor Scheduling for Model-based Reconstruction of Spatially Distributed Phenomena). Student research project, Intelligent Sensor-Actuator-Systems Laboratory, Universität Karlsruhe (TH), 2008.

[204] A. Kuwertz. Nichtlineare Sensoreinsatzplanung zur modellbasierten Quellenverfolgung bei räumlich ausgedehnten Phänomenen (Nonlinear Sensor Management for Model-based Source Detection in Spatially Distributed Phenomena). Diploma thesis, Intelligent Sensor-Actuator-Systems Laboratory, Universität Karlsruhe (TH), 2009.

[205] J. Meyer. Nichtlineare Sensoreinsatzplanung für Sensor-Aktor-Netzwerke (Nonlinear Sensor Scheduling for Sensor-Actuator-Networks). Diploma thesis, Intelligent Sensor-Actuator-Systems Laboratory, Universität Karlsruhe (TH), 2007.

[206] E. Stiegeler. Prädiktions- und Einsatzplanungsverfahren zur effizienten Zustandsschätzung von Prozessketten (Prediction and Scheduling Methods for Efficient State Estimation of Supply Chains). Diploma thesis, Intelligent Sensor-Actuator-Systems Laboratory, Universität Karlsruhe (TH), 2007.

[207] E. Stiegeler. Sensoreinsatzplanung in unzuverlässigen Kommunikationsnetzwerken (Sensor Scheduling in Unreliable Communication Networks). Student research project, Intelligent Sensor-Actuator-Systems Laboratory, Universität Karlsruhe (TH), 2007.

Own Publications

[208] F. Beutler, M. F. Huber, and U. D. Hanebeck. Gaussian Filtering using State Decomposition Methods. In *Proceedings of the 12th International Conference on Information Fusion (Fusion)*, pages 579–586, Seattle, Washington, July 2009.

[209] F. Beutler, M. F. Huber, and U. D. Hanebeck. Instantaneous Pose Estimation using Rotation Vectors. In *Proceedings of the 34th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 3413–3416, Taipei, Taiwan, Apr. 2009.

[210] F. Beutler, M. F. Huber, and U. D. Hanebeck. Probabilistic Instantaneous Model-Based Signal Processing applied to Localization and Tracking. *Journal of Robotics and Autonomous Systems — Special Issue: Selected papers from 2006 IEEE International Conference on Multisensor Fusion and Integration (MFI 2006)*, 57(3):249–258, Mar. 2009.

[211] M. P. Deisenroth, M. F. Huber, and U. D. Hanebeck. Analytic Moment-based Gaussian Process Filtering. In *26th International Conference on Machine Learning (ICML)*, pages 225–232, Montreal, Canada, June 2009.

[212] M. Huber, D. Brunn, and U. D. Hanebeck. Closed-Form Prediction of Nonlinear Dynamic Systems by Means of Gaussian Mixture Approximation of the Transition Density. In *Proceedings of the 2006 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 98–103, Heidelberg, Germany, Sept. 2006.

[213] M. F. Huber, T. Bailey, H. Durrant-Whyte, and U. D. Hanebeck. On Entropy Approximation for Gaussian Mixture Random Vectors. In *Proceedings of the 2008 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 181–188, Seoul, Republic of Korea, Aug. 2008.

[214] M. F. Huber, D. Brunn, and U. D. Hanebeck. Efficient Nonlinear Measurement Updating based on Gaussian Mixture Approximation of Conditional Densities. In *Proceedings of the 2007 American Control Conference (ACC)*, pages 4425–4430, New York, New York, July 2007.

[215] M. F. Huber and U. D. Hanebeck. Hybrid Transition Density Approximation for Efficient Recursive Prediction of Nonlinear Dynamic Systems. In *International Conference on Information Processing in Sensor Networks (IPSN)*, pages 283–292, Cambridge, Massachusetts, Apr. 2007.

[216] M. F. Huber and U. D. Hanebeck. The Hybrid Density Filter for Nonlinear Estimation based on Hybrid Conditional Density Approximation. In *Proceedings of the 10th International Conference on Information Fusion (Fusion)*, Quebec, Canada, July 2007.

[217] M. F. Huber and U. D. Hanebeck. Gaussian Filter based on Deterministic Sampling for High Quality Nonlinear Estimation. In *Proceedings of the 17th IFAC World Congress*, Seoul, Republic of Korea, July 2008.

[218] M. F. Huber and U. D. Hanebeck. Priority List Sensor Scheduling using Optimal Pruning. In *Proceedings of the 11th International Conference on Information Fusion (Fusion)*, Cologne, Germany, July 2008.

[219] M. F. Huber and U. D. Hanebeck. Progressive Gaussian Mixture Reduction. In *Proceedings of the 11th International Conference on Information Fusion (Fusion)*, Cologne, Germany, July 2008.

[220] M. F. Huber, A. Kuwertz, F. Sawo, and U. D. Hanebeck. Distributed Greedy Sensor Scheduling for Model-based Reconstruction of Space-Time Continuous Physical Phenomena. In *Proceedings of the 12th International Conference on Information Fusion (Fusion)*, pages 102–109, Seattle, Washington, July 2009.

[221] M. F. Huber, E. Stiegeler, and U. D. Hanebeck. On Sensor Scheduling in Case of Unreliable Communication. In *INFORMATIK 2007 - the 37th Annual Conference of the Gesellschaft für Informatik e.V. (GI), 3rd German Workshop Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, pages 90–94, Bremen, Germany, Sept. 2007.

[222] F. Sawo, M. F. Huber, and U. D. Hanebeck. Parameter Identification and Reconstruction Based on Hybrid Density Filter for Distributed Phenomena. In *Proceedings of the 10th International Conference on Information Fusion (Fusion)*, Quebec, Canada, July 2007.

[223] D. Schieferdecker and M. F. Huber. Gaussian Mixture Reduction via Clustering. In *Proceedings of the 12th International Conference on Information Fusion (Fusion)*, pages 1536–1543, Seattle, Washington, July 2009.

[224] F. Weissel, M. F. Huber, D. Brunn, and U. D. Hanebeck. Stochastic Optimal Control based on Value-Function Approximation using Sinc Interpolation. In *Proceedings of the 17th IFAC World Congress*, Seoul, Republic of Korea, July 2008.

[225] F. Weissel, M. F. Huber, and U. D. Hanebeck. A Closed–Form Model Predictive Control Framework for Nonlinear Noise–Corrupted Systems. In *Proceedings of the 4th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, volume SPSMC, pages 62–69, Angers, France, May 2007.

[226] F. Weissel, M. F. Huber, and U. D. Hanebeck. A Nonlinear Model Predictive Control Framework Approximating Noise Corrupted Systems with Hybrid Transition Densities. In *Proceedings of the 2007 IEEE Conference on Decision & Control (CDC)*, pages 3661–3666, New Orleans, Louisiana, Dec. 2007.

[227] F. Weissel, M. F. Huber, and U. D. Hanebeck. Efficient Control of Nonlinear Noise–Corrupted Systems Using a Novel Model Predictive Control Framework. In *Proceedings of the 2007 American Control Conference (ACC)*, pages 3751–3756, New York, New York, July 2007.

[228] F. Weissel, M. F. Huber, and U. D. Hanebeck. Test-Environment based on a Team of Miniature Walking Robots for Evaluation of Collaborative Control Methods. In *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2474–2479, San Diego, California, Nov. 2007.

[229] F. Weissel, M. F. Huber, and U. D. Hanebeck. Stochastic Nonlinear Model Predictive Control based on Gaussian Mixture Approximations. In *Selected Papers from the International Conference on Informatics in Control, Automation and Robotics 2007*, volume 24 of *Lecture Notes in Electrical Engineering*, pages 239–252. Springer, Sept. 2008.

[230] F. Weissel, T. Schreiter, M. F. Huber, and U. D. Hanebeck. Stochastic Model Predictive Control of Time-Variant Nonlinear Systems with Imperfect State Information. In *Proceedings of the 2008 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 40–46, Seoul, Republic of Korea, Aug. 2008.

Karlsruhe Series on
Intelligent Sensor-Actuator-Systems
Edited by Prof. Dr.-Ing. Uwe D. Hanebeck

The information content of measurements from a sensor system with many different sensing modalities varies with the chosen sensor configuration. To maximize the utility of such a sensor system, sensor management aims for dynamically configuring the sensor system and its sensing modalities in the most informative way. For high quality sensor management, however, the anticipation of long-term effects of possible sensor configurations and the inference of the internal state of the observed object from noisy measurements is essential. Incorporating these requirements, in this thesis stochastic control and Bayesian estimation techniques are combined to form a versatilely applicable sensor management framework.

For anticipating future effects of sensor management decisions, two predictive methods are introduced in the first part of this thesis. Their application domain depends on the available resources of the sensor system and on the complexity of the underlying nonlinear estimation problem, i.e., whether the estimation problem is close to the well-known linear Gaussian problem or not. Correspondingly, an open-loop or closed-loop control approach for determining sensor configurations is employed. Both sensor management methods are tightly knit with the estimation techniques that form the second part of this thesis. Instead of exactly solving the nonlinear estimation problem, which is computationally highly demanding, different ways of approximating the exact solution are proposed that differ in approximation quality and resource requirements.

ISSN: 1867-3813
ISBN: 978-3-86644-405-8

www.uvka.de