Multi-Extended Object Tracking with Generalized Physical Models

Dissertation
zur Erlangung des Doktorgrades (Dr. rer. nat)
der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

M. Eng. Patrick Hoher

aus Überlingen

Bonn, 2026

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät
der Rheinischen Friedrich-Wilhelms-Universität Bonn

Gutachter/Betreuer: Prof. Dr. Wolfgang Koch
Gutachter: Prof. Dr. Michael Meier

Tag der Promotion: 10.6.2026
Erscheinungsjahr: 2026

Acknowledgments

I would like to express my sincere gratitude to Prof. Dr. Johannes Reuter from the Institute of System Dynamics. His excellent supervision during my Master's studies inspired my enthusiasm for sensor data fusion and laid the foundation for this dissertation. This work would not have been possible without his continuous commitment to the Institute, the research projects of recent years, and the joint scientific publications. I am also grateful to him for initiating the collaboration with Bonn.

I am deeply indebted to Prof. Dr. Wolfgang Koch and Dr. Felix Govaers from Fraunhofer FKIE and the University of Bonn for their valuable guidance and insightful discussions, particularly in the context of our joint publications, which greatly advanced this dissertation. My gratitude also goes to Prof. Dr. Michael Meier, who kindly agreed to serve as an additional supervisor and examiner, thereby enabling the doctoral procedure.

I would further like to thank my current and former colleagues at HTWG Konstanz: Tim Baur, Stefan Wirtensohn, Dennis Griesser, Daniel Dold, Hannes Homburger, Ruven Weiss, Matthias Albrecht, Oliver Hamburger, Dr. Michael Schuster, Florian Wirtensohn, Dr. Tristan Braun and Dr. Christian Benkler. The many technical discussions, as well as the pleasant working atmosphere and our shared lunch and coffee breaks, hiking trips, bowling evenings, and barbecues, made my time at HTWG especially memorable.

Finally, I owe my deepest gratitude to the most important people in my life, who have always supported me and on whom I could always rely. My heartfelt thanks go to my parents, Petra and Gebhard, and to my sister Jenny, who have continuously encouraged me and believed in me from the very beginning of my education to the completion of this dissertation. My final and most personal thanks go to my partner Jane, whose companionship and support throughout my doctoral journey have been invaluable. I feel truly fortunate to have you at my side as I embark on the next chapter of my life.

Contents

1. Introduction **1**
- 1.1. Physics-based modeling in multi-extended object tracking . . . 1
- 1.2. Methodologies and contributions . . . 2
- 1.3. Core theses of this work . . . 3
- 1.4. Structure of this thesis . . . 3
- 1.5. Publications . . . 3

2. Background **5**
- 2.1. Object dynamic model . . . 5
- 2.2. Measurement model and measurement likelihood function . . . 6
- 2.3. Bayesian filtering . . . 7
- 2.4. Linear Kalman filter . . . 7
- 2.5. Random matrix approach . . . 8
- 2.6. Random finite sets . . . 10
- 2.7. Bernoulli filter . . . 11
- 2.8. The labeled multi-Bernoulli filter . . . 12
- 2.9. Probability hypothesis density filter . . . 15
- 2.10. Accumulated state densities and sets of trajectories . . . 16
- 2.11. Trajectory probability hypothesis density filter . . . 17
- 2.12. Gaussian process regression . . . 19

I. Extended Object Tracking with Virtual Measurement Models **21**

3. Measurement and shape models **23**
- 3.1. Ellipse . . . 25
- 3.2. Triangle . . . 28
- 3.3. Rectangle . . . 29
- 3.4. Ellipsoid . . . 30
- 3.5. Elliptical cone . . . 32
- 3.6. Rectangular cuboid . . . 33
- 3.7. Partially visible objects and occlusion . . . 34

4. The virtual measurement model framework **35**
- 4.1. Reference model concept . . . 36
- 4.2. Adaptation algorithm . . . 40
- 4.3. Expansion to multi-extended object tracking . . . 44
- 4.4. Gaussian processes regression model with VMMs . . . 47

5. Shape classification with virtual measurement models **51**
- 5.1. Shape classification based on histograms . . . 52
- 5.2. Shape classification based on Chamfer distances . . . 52

6. Results **55**
- 6.1. Extension estimation with an elliptical object . . . 55
- 6.2. Extension estimation with the Gaussian processes regression model . . . 61
- 6.3. Multi-extended object tracking and occlusions . . . 66
- 6.4. Comparison with other EOT methods . . . 68
- 6.5. Non-elliptical shapes and shape classification in 2D and 3D . . . 70
- 6.6. Discussion and conclusion for virtual measurement models . . . 80
- 6.7. Future work . . . 81

II. Adaptive Birth Densities **83**

7. Detection-driven adaptive birth density **85**
7.1. Related work . . . 86
7.2. Detection-driven adaptation . . . 87
7.3. Rauch-Tung-Striebel (RTS) recursion . . . 87
7.4. Birth density adaptation . . . 88

8. Circular birth density **91**
8.1. Gaussian mixture approximation . . . 92
8.2. Cartesian transformation of a polar Gaussian distribution . . . 96
8.3. Birth velocities . . . 98
8.4. Detection-driven adaptation for a circular birth density . . . 99
8.5. Effects of ego motion . . . 100

9. Results **103**
9.1. Single Gaussian birth density . . . 103
9.2. Circular birth density adaptation . . . 108
9.3. Comparison with other birth models . . . 111
9.4. Simulation studies with ego motion consideration . . . 112
9.5. Full-scale experiments . . . 115
9.6. Critical interpretation of the results . . . 118
9.7. Conclusion for adaptive birth densities . . . 119

A. Derivation of the constant acceleration model **121**

Acronyms

ASD accumulated state density

CA constant acceleration

CV constant velocity

CT coordinated turn

CAD computer-aided design

CBMeMBer cardinality balanced multi-target multi-bernoulli

CDF cumulative distribution function

CoG center of gravity

CPHD cardinalized probability hypothesis density

DBSCAN density-based spatial clustering of applications with noise

EOT extended object tracking

ENU x-East, y-North, z-Up

FISST finite set statistics

GLMB generalized labeled multi-Bernoulli

GM Gaussian mixture

GP Gaussian process

GPDA generalized probabilistic data association

GPRM Gaussian processes regression model

GPS global positioning system

IID independent and identically distributed

IMU inertial measurement unit

IoU intersection over union

KLD Kullback-Leibler divergence

LMB labeled multi-Bernoulli

MC Monte Carlo

MEM-EKF* multiplicative error model extended Kalman filter

MeMBer multi-target multi-Bernoulli

MEOT multi-extended object tracking

MOT multi object tracking

ODM object dynamic model

OOS out-of-sequence

OSPA optimal subpattern assignment

pdf probability density function

PHD probability hypothesis density

RFS random finite set

RHM random hypersurface model

RM random matrix

RMSE root mean square error

RTS Rauch-Tung-Striebel

SNN suboptimal nearest-neighbor

SPRB set of points on a rigid body

TPHD trajectory probability hypothesis density

VMM virtual measurement model

1. Introduction

The primary objective of multi-extended object tracking (MEOT) is to estimate both the kinematic state (e.g. position, velocity, and acceleration) and the geometric parameters, including shape and size, of an unknown and potentially varying number of objects. MEOT methods are essential in numerous applications, including autonomous driving [1–5], robotics [6–8], aviation and air traffic control, maritime surveillance, autonomous water taxis [9–11, C1], ferries [12–14], defense and security [15], and agriculture [16].
This PhD thesis proposes methodological contributions to physics-based modeling within MEOT, aiming for broad applicability across different domains. Special attention is given to maritime scenarios, as they present unique and significant challenges. One notable challenge arises from the heterogeneous nature of maritime traffic, which includes participants varying dramatically in size, shape and dynamics, from swimmers and stand-up paddlers to motorboats, sailing vessels, ferries, cruise ships, and container ships. Additionally, the absence of predefined lanes or structured traffic patterns allows new objects to appear unpredictably from any direction, further complicating the tracking task.

1.1. Physics-based modeling in multi-extended object tracking

Physics-based modeling plays a key role in tracking applications. For example, to predict the movement of other objects, it is crucial to model their motion dynamics. Movement predictions are not only essential for collision avoidance but are also required during the short sampling periods of sensor systems in standard tracking methods, such as the Kalman filter [17]. In tracking and control, various object dynamic model (ODM) range from precise physical models tailored to specific scenarios to more generalized models that can be applied across a broader range of conditions. For instance, the dynamic behavior of a small unmanned surface vehicle can be described by a complex model that considers inertia, Coriolis effects, and hydrodynamic damping [18, 19]. Such detailed modeling is highly relevant in control engineering applications, as it describes the relationship between the voltage at the motor and the resulting acceleration of the boat.
In contrast, tracking applications adopt an external perspective, focusing on observable states such as position, velocity, and acceleration. Internal states of the boat, such as the current and voltage in the motor, do not play a role at all in these models. Instead, all factors influencing the kinematic states, including the effects of actuators and external disturbances like current and wind, are collectively modeled as process noise. The ODMs typically used in tracking, such as the constant acceleration (CA), constant velocity (CV), or coordinated turn (CT) models, are based on underlying physical principles that have been generalized to allow their application in a wide variety of scenarios, ranging from maritime to automotive environments. Therefore, these ODMs serve as examples of generalized physical models that are widely used in tracking applications.
In multi object tracking (MOT), interactions among multiple objects can also be modeled using physical principles. For example, in [20] basketball players are tracked using a Kalman filter incorporating a CA-ODM. However, by modeling human behavior to actively avoid collisions, the tracking performance can be further enhanced [20]. In addition, human motion has been examined in [21, 22], where the dynamics of leg movements during walking are modeled to improve the recognition capabilities of vision-based tracking algorithms. However, these models are highly specialized for capturing the nuances of an anthropomorphic walk and are therefore not sufficiently generalized for transfer to other applications.
Generalized physical models also play an important role in extended object tracking (EOT), particularly in modeling the shape of objects and the distribution of measurement sources. For instance, one common assumption is that the measurements are either normally or uniformly distributed over the object's surface or contour, while an alternative physics-based approach involves tracing each individual lidar beam to determine the exact measurement source via its

intersection with the contour. For the shape itself, there are generalizing approaches, assuming e.g. an elliptical shape or approaches that estimate arbitrary shapes with many details.

1.2. Methodologies and contributions

This PhD thesis makes significant contributions to two key aspects of (MEOT) through the use of generalized physical models. First, complex measurement models for various two-dimensional and three-dimensional shapes are developed and integrated into a tracking framework, referred to as the virtual measurement model (VMM) approach. Second, the thesis introduces advanced physics-based modeling techniques for target birth processes, enabling more accurate initialization of objects in scenarios with 360° sensor coverage.

Lidar measurements, which are typically obtained along object contours, can be accurately modeled using ray tracing. However, lidar beams may pass through openings, such as windows, resulting in measurements that originate either from the object's surface or, in 3D scenarios, from within its volume. Those measurements are also denoted interior measurements. Moreover, the exact shape of the object is not always known in advance. The proposed EOT approach addresses these challenges by accounting for both interior and contour measurements while automatically selecting the most suitable base shape from a predefined set.

The central question of the first chapter is how complex measurement models can be integrated into a unified EOT framework for various shapes. To address this, the VMM approach is introduced. The VMM generates artificial measurements based on input parameters and embedded physical models. It relies on a Kalman filter with random matrix-based extension estimation, which first filters the centroid and the spread of measurements over time. The objective is to estimate the object's state and extent parameters so that the artificial measurements generated by the VMM match the statistical moments of those obtained from the random matrix tracking algorithm. This is best achieved through an iterative adaptation process, whose convergence is formally proven in this work.

The VMM framework incorporates both the object's shape and the measurement category. In cases where the optimal shape representation is unknown or the measurement type (interior, contour, or mixed) is ambiguous, multiple VMM instances can be applied in parallel. The resulting artificial measurements can then be used for classification. To evaluate the proposed VMM approach, extensive simulations are performed and real lidar data, primarily from maritime scenarios, are analyzed.

The second part of this thesis focuses on the search areas of tracking algorithms and the initialization required upon the first detection of new objects. This search area, or more precisely, the probability density used to model it, is referred to as birth density. Determining when, where, and with which initial state parameters new objects appear poses a key challenge in tracking an unknown number of objects.

A widely used approach is the measurement-driven adaptive birth density, where measurements that cannot be assigned to any existing object generate a hypothesis for a new object. If subsequent measurements in close proximity to the initial detection confirm this hypothesis, its weight increases; otherwise, the hypothesis is eventually discarded. While this method proves efficient in simple scenarios, it often results in a high false alarm rate and greater computational effort in cluttered environments.

A central question of the second part is how to model an adaptive birth density that reliably detects all new objects while being less sensitive to clutter than measurement-driven approaches. The adaptive birth density proposed in this thesis is derived from physical principles: a sensor's range is inherently limited, and in an open environment, new objects must appear at the boundary of the surveillance area; otherwise, they would already have been detected and would not be considered new. Consequently, the search area of filtering algorithms does not need to cover the entire state space. However, determining the exact boundary of the surveillance area and its width must be done adaptively, using information about the initial positions of previously

detected objects.
For lidar sensors with a 360° field of view, assuming point symmetry is reasonable, leading to the introduction of a circular birth density in polar coordinates. Furthermore, prior knowledge about the initial states of new objects can be leveraged based on geometric and physical constraints. For example, if the own object is moving in a certain direction, new objects can only appear behind it if they have a higher velocity. Slower objects would either already be within the surveillance area or would never enter it at all. Accurately modeling these relationships improves the initialization process for new objects. The adaptive birth densities are extensively evaluated through simulations, demonstrating that adaptation enhances tracking performance. Their relevance in real-world environments is demonstrated using a maritime scenario.

1.3. Core theses of this work

The findings of this thesis can be summarized in the following three core theses:

1. Virtual measurement models enable reliable and precise estimation of object extent and position across various shapes and sensor modalities.

2. The virtual measurements generated as a byproduct can be used to classify an object's shape or measurement model.

3. The initial states of objects detected by a filter can be leveraged to adapt the birth density for tracking future objects. A formulation in polar coordinates enables full 360° coverage.

The core theses and results of this work are documented in two journal articles and seven conference papers.

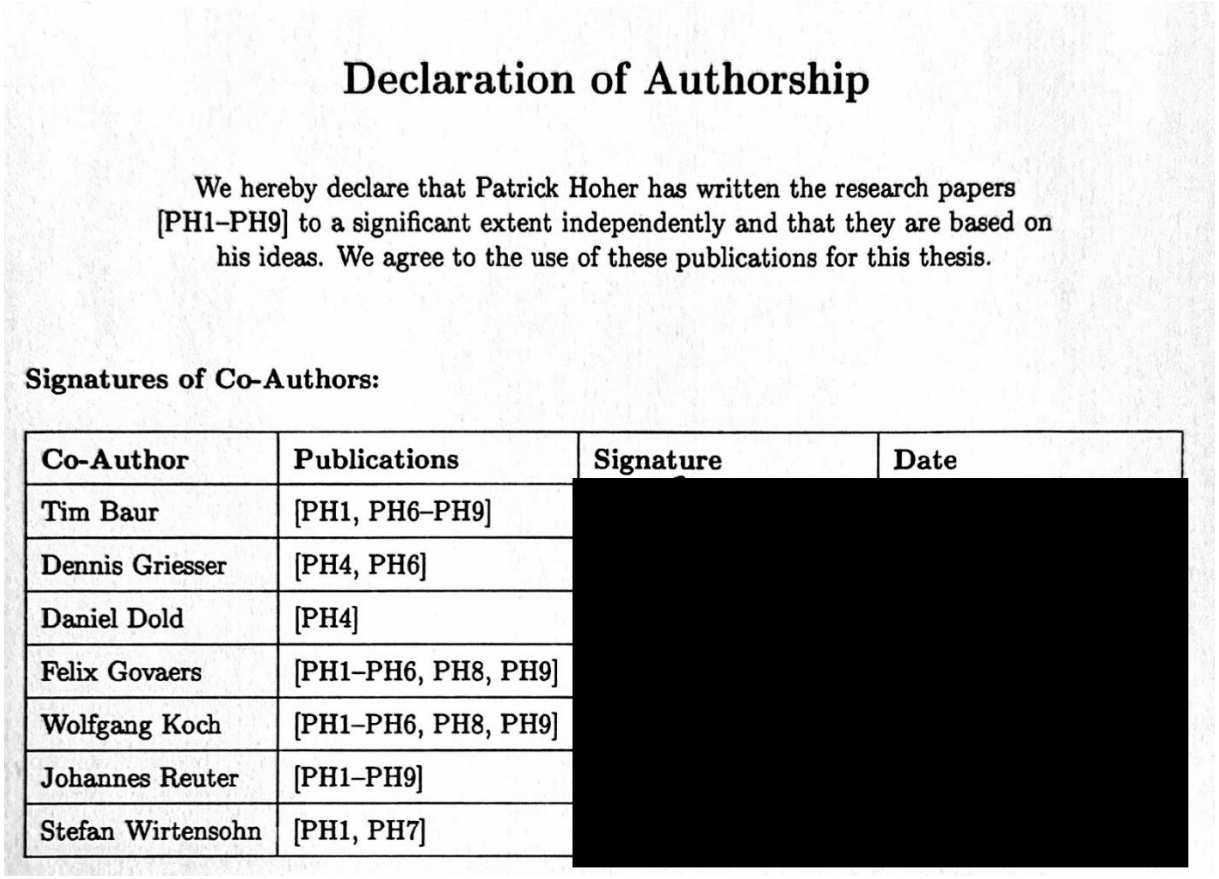
1.4. Structure of this thesis

Section 2 presents the fundamental concepts of MEOT that are necessary for understanding the remainder of this thesis. The body of the thesis is divided into two main parts.
Part 1, comprising Sections 3–6, introduces the novel EOT approach based on VMMs. Section 3 introduces measurement models for various 2D and 3D shapes. For each shape, methods to generate contour and interior measurements are presented. Section 4 describes the VMM framework, including the adaptation algorithm, an extension to the MEOT setting, and the use of a Gaussian process regression model (Gaussian processes regression model (GPRM)) as an alternative to the adaptation algorithm. The artificial measurements generated by the VMM during the adaptation process are used for shape classification in section 5, where two classification methods are presented: one based on histograms and another using Chamfer distances. Finally, section 6 evaluates the VMM framework using both simulation studies and real-world experiments.
Part 2, covering sections 7–9, presents the detection-driven adaptive birth densities. Section 7 introduces the framework for adapting a single Gaussian birth density based on the detections made by the filter. In section 8, the circular birth density is introduced to enable full 360° coverage of the surveillance area. Finally, section 9 evaluates the adaptive birth densities from sections 7 and 8 through simulation studies and real-world experiments.

1.5. Publications

The content of this thesis has been published in two peer-reviewed journal articles and seven conference papers. These include work on EOT [PH1–PH6] and adaptive birth densities [PH7–PH9]. In addition, I contributed as a co-author to several other publications [C1–C6]. These works are not part of this thesis, as the main contributions have been made by colleagues, they fall under the scope of other dissertations, or they are not directly relevant to the core topics addressed here.

The publications [PH1–PH9] forming the basis of this thesis were written independently by me to a significant extent and are based on my own ideas. The following statement affirms that all co-authors agree to the inclusion of these publications in this thesis and support the declaration of my own contributions.



Declaration of Authorship

We hereby declare that Patrick Hoher has written the research papers [PH1–PH9] to a significant extent independently and that they are based on his ideas. We agree to the use of these publications for this thesis.

Signatures of Co-Authors:

Co-Author	Publications	Signature	Date
Tim Baur	[PH1, PH6–PH9]		
Dennis Griesser	[PH4, PH6]		
Daniel Dold	[PH4]		
Felix Govaers	[PH1–PH6, PH8, PH9]		
Wolfgang Koch	[PH1–PH6, PH8, PH9]		
Johannes Reuter	[PH1–PH9]		
Stefan Wirtensohn	[PH1, PH7]		

Although this thesis is based on publications, it is written in the form of a monograph. This reflects the continuous development and refinement of ideas that originated in earlier work, such as [PH1]. Within the thesis, the content from these papers has been restructured in a unified context and updated accordingly. Redundancies have been eliminated, and additional comparative discussions of the various approaches have been included to improve coherence and readability. In contrast, the content of more recent papers, particularly [PH9], which was published almost simultaneously with the completion of this thesis, has been largely incorporated in its original form.

2. Background

2.1. Object dynamic model

A crucial aspect of tracking is the physics-based modeling of an object's dynamics, describing the evolution of the kinematic state $\underline{x}$ as a Markov process over time steps k. The general state transition equation is given by [23, p. 22][24, p. 125]:

$$\underline{x}_k = f\left(\underline{x}_{k-1}, \underline{w}_k\right), \tag{2.1}$$

where $\underline{w}_k$ represents the process noise. For linear models with additive noise, this simplifies to:

$$\underline{x}_k = F_k \underline{x}_{k-1} + \underline{w}_k, \tag{2.2}$$

where F_k is the transition matrix. The state transition density for linear object dynamic models (ODMs) can be expressed by [23, p. 27]

$$p\left(\underline{x}_k \mid \underline{x}_{k-1}\right) = \mathcal{N}\left(\underline{x}_k; F_k \underline{x}_{k-1}, Q_k\right) \tag{2.3}$$

whereas Q_k is the process noise covariance matrix. The simplest transition model is the constant velocity (CV) model with state vector $\underline{x} = (x, v)^\top$ consisting of position and velocity in one direction and sampling period T [23, p. 27] [24, p. 344][25, p. 142][26, p. 203]:

Constant Velocity (CV) Model

$$F_k = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}, \qquad \underline{w}_k = \begin{bmatrix} \frac{T^2}{2} \\ T \end{bmatrix} a_k, \qquad Q_k = \begin{bmatrix} \frac{T^4}{4} & \frac{T^3}{2} \\ \frac{T^3}{2} & T^2 \end{bmatrix} \sigma_{a_k}^2.$$

In the CV model , the noise term $a_k \sim \mathcal{N}\left(a_k; 0, \sigma_{a_k}^2\right)$ represents an unknown acceleration that affects the transition. For d dimensions, the CV model can be expanded using the Kronecker product $\otimes$:

$$F_{k,d} = F_k \otimes I_d, \tag{2.4}$$

where I_d is the identity matrix $I_d \in \mathbb{R}^{d \times d}$. The process noise vector $\underline{w}_k$ and the covariance matrix Q_k are extended similarly. Applying the Kronecker product results in decorrelated dimensions, an assumption commonly made in CV models.
The constant acceleration (CA) model, where the kinematic state $\underline{x} = (x, v, a)^\top$ consists of position, velocity and acceleration, is given by [23, p. 65][24, p. 344][26, p. 204]:

Constant Acceleration (CA) Model

$$F_k = \begin{bmatrix} 1 & T & \frac{T^2}{2} \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix}, \qquad \underline{w}_k = B_k w_k = \begin{bmatrix} \frac{T^3}{6} \\ \frac{T^2}{2} \\ T \end{bmatrix} w_k, \qquad Q_k = \begin{bmatrix} \frac{T^4}{20} & \frac{T^3}{8} & \frac{T^2}{6} \\ \frac{T^3}{8} & \frac{T^2}{3} & \frac{T}{2} \\ \frac{T^2}{6} & \frac{T}{2} & 1 \end{bmatrix} \sigma_w^2 T^2.$$

In the CA model, the noise term w_k represents an unknown change in acceleration. Consequently, its unit is $\mathrm{m/s^4}$ and the covariance matrix Q_k models the uncertainty associated with this noise. The derivations of F_k and $\underline{w}_k$ are done using an exact discretization of a linear time continuous system, which is shown as an example in Appendix A. Approximations, e.g. the Euler method or the trapezoidal approach, that have to be used for nonlinear systems, lead to similar but different results. A simple approach to calculate the covariance matrix is given by $Q_k = \sigma_w^2 B_k B_k^\top$, which

is used for most simulations in this work, but this does not lead to the result for Q_k given in this section. Further details are discussed in Appendix A.
Another linear model frequently used within this work is Singer's ODM [27, p. 475][24, p. 346], presented here in its simplified form

Singer's Object Dynamic Model (ODM)

$$F_k = \begin{bmatrix} 1 & T & \frac{T^2}{2} \\ 0 & 1 & T \\ 0 & 0 & e^{-\frac{T}{\tau_s}} \end{bmatrix}, \qquad \underline{w}_k = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} a_k, \qquad Q_k = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \sigma_{a_k}^2 \end{bmatrix}.$$

Singer's ODM introduces a time constant τ_s, which models a natural decay of acceleration. In this simplified version, the process noise term a_k directly affects only the acceleration state, consistent with the result obtained from an explicit Euler discretization.
Other commonly used ODMs include the horizontal turn model [28, p. 389] and the coordinated turn model [29].

2.2. Measurement model and measurement likelihood function

The measurement at the current step $\underline{z}_k$, depends on the current state $\underline{x}_k$ and a measurement noise term $\underline{v}_k$ [23, p. 23][24, p. 125]

$$\underline{z}_k = h\left(\underline{x}_k, \underline{v}_k\right), \tag{2.5}$$

which simplifies to [23, p. 25][24, p. 21]

$$\underline{z}_k = H_k \underline{x}_k + \underline{v}_k \tag{2.6}$$

for linear measurement models with additive white Gaussian measurement noise, where $\underline{v}_k \sim \mathcal{N}\left(\underline{v}_k; 0; R_k\right)$. For positional measurements in combination with the CV model, the measurement matrix is $H_k = (1, 0)$. Similarly, for the CA and Singer's model, $H_k = (1, 0, 0)$. For d dimensions, the Kronecker product can be used to expand H_k. The linear measurement likelihood function for objects without extension is given by:

$$p\left(\underline{z}_k \mid \underline{x}_k\right) = \mathcal{N}\left(\underline{z}_k; H_k \underline{x}_k, R_k\right), \tag{2.7}$$

where R_k is the covariance matrix of the measurement noise.
In extended object tracking (EOT), however, the tracked object is characterized by an additional extension state X_k, in addition to its kinematic state $\underline{x}_k$. The random matrix (RM) algorithm [30], which is introduced later in this section, assumes that measurements are distributed normally across the object's extent. Taking into account multiple measurements and the additional spread caused by the object's extension, the measurement likelihood is expressed as follows [31, p. 6][32, p. 1031]:

$$p(\boldsymbol{Z}_k | n_k, \underline{x}_k, X_k) = \prod_{j=1}^{n_k} \mathcal{N}(\underline{z}_k^j; (H_k \otimes I_d)\underline{x}_k, X_k + R_k), \tag{2.8}$$

whereas n_k is the number of measurements.
In many of the measurement and shape models that are introduced in this work, the likelihood function can not be expressed in a closed form. For instance, using a lidar sensor results in contour measurements, where the center of gravity (CoG) does not necessarily coincide with the object's centroid. Additionally, measurements are generated with a nonlinear function (2.5), that is also not necessarily given in closed form. Within the lidar measurement model, all measurements in $\boldsymbol{Z}_k$ are correlated to each other, although measurement noise independently affects

each measurement.
For arbitrary measurement distributions, it is still possible to use the linear measurement likelihood function (2.8) in a recursive filter. However, this approach requires careful interpretation of the results. Specifically, the filter will estimate the CoG of the measurements rather than the actual position of the object's centroid. Moreover, the filter will track the covariance matrix of the measurement spread, which is influenced by the object's extension, rather than estimating the true extension itself.

2.3. Bayesian filtering

The goal of tracking is to determine the conditional probability density function (pdf) $p\left(\underline{x}_k|\boldsymbol{Z}^k\right)$ which represents the belief about the current state $\underline{x}_k$ of an object given the accumulated measurements $\boldsymbol{Z}^k = \{\underline{z}_1, \underline{z}_2, ..., \underline{z}_k\}$ up to the current time step k. The Bayes filter provides a recursive framework for estimating $p\left(\underline{x}_k|\boldsymbol{Z}^k\right)$ and consists of a prediction and an update step [23, p. 23].

Recursive Bayes Filter

Prediction:

$$p\left(\underline{x}_k \mid \boldsymbol{Z}^{k-1}\right) = \int p\left(\underline{x}_k \mid \underline{x}_{k-1}\right) p\left(\underline{x}_{k-1} \mid \boldsymbol{Z}^{k-1}\right) d\underline{x}_{k-1}.$$

Update:

$$p\left(\underline{x}_k \mid \boldsymbol{Z}^{k}\right) = \frac{p\left(\underline{z}_k \mid \underline{x}_k\right) p\left(\underline{x}_k \mid \boldsymbol{Z}^{k-1}\right)}{p\left(\underline{z}_k \mid \boldsymbol{Z}^{k-1}\right)},$$

where

$$p\left(\underline{z}_k \mid \boldsymbol{Z}^{k-1}\right) = \int p\left(\underline{z}_k \mid \underline{x}_k\right) p\left(\underline{x}_k \mid \boldsymbol{Z}^{k-1}\right) d\underline{x}_k.$$

In the prediction step, the prior $p\left(\underline{x}_{k-1} \mid \boldsymbol{Z}^{k-1}\right)$ is obtained from the posterior result of the previous iteration $p\left(\underline{z}_k \mid \boldsymbol{Z}^{k-1}\right)$ considering the state transition probability $p\left(\underline{x}_k \mid \underline{x}_{k-1}\right)$. In the update step, the prior is refined using the current measurement using the measurement likelihood function $p\left(\underline{z}_k \mid \underline{x}_k\right)$, yielding the posterior of the current time step $p\left(\underline{x}_k \mid \boldsymbol{Z}^{k}\right)$.

2.4. Linear Kalman filter

Under the assumptions that the posterior always follows a normal distribution $p\left(\underline{x}_k \mid \boldsymbol{Z}^{k}\right) \sim \mathcal{N}\left(\underline{x}_k; \underline{x}_{k|k}, P_{k|k}\right)$ and that the state transition probability and the measurement likelihood function are both linear with additive white Gaussian noise (see Eq. (2.3) and (2.7)), the linear Kalman filter can be derived from the Bayes filter [23, p. 23]. A detailed derivation of the Linear Kalman filter can e.g. be found in [23, p. 25-30] Under those assumptions, the Kalman filter is the optimal Bayes filter for a single object without extension.

Linear Kalman Filter

Prediction Step:

$$\underline{x}_{k|k-1} = F_k \underline{x}_{k-1|k-1},$$
$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^\top + Q_k.$$

Update Step:

$$\underline{x}_{k|k} = \underline{x}_{k|k-1} + K_k \left(\underline{z}_k - H_k \underline{x}_{k|k-1} \right),$$
$$P_{k|k} = \left(I - K_k H_k \right) P_{k|k-1}.$$

with Kalman gain and innovation:

$$K_k = P_{k|k-1} H_k^\top S_k^{-1},$$
$$S_k = H_k P_{k|k-1} H_k^\top + R_k.$$

2.5. Random matrix approach

The random matrix (RM) approach is a Bayesian filtering recursion to estimate the joint density $p(\underline{x}_k, X_k | \boldsymbol{Z}^k)$ and was initially introduced in [30]. Several assumptions were made to estimate kinematics and extension separately [30, p. 1046]. The density of the kinematical state $\underline{x}_k$ is Gaussian (2.3) and the density of the extension X_k is an inverse Wishart density

$$p(X_k | \boldsymbol{Z}^k) = \mathcal{IW}(X_k; \nu_{k|k}, X_{k|k}) \tag{2.9}$$

with $X_{k|k} > 0$ and $\nu_{k|k} > (2d + 2)$. The expected value of the inverse Wishart density is given by:

$$\mathbb{E}(X_{k|k}) = \frac{X_{k|k}}{\nu_{k|k} - (2d + 2)}. \tag{2.10}$$

Since $\mathbb{E}(X_{k|k})$ is the expected value for the spread of the measurements caused by the object's extent, it is further assumed that the square roots of the eigenvalues of $\mathbb{E}(X_{k|k})$ correspond to the extension parameters length, width and in 3D additionally height. The object's orientation $\psi_{k|k}$ can either be given from the extension matrix or the kinematics.

Prediction For the kinematic state, the prediction equations of the linear Kalman filter are used within the RM framework. However, for the extent, the prediction equations must ensure that the expected value (2.10) stays constant and are given as follows [30, p. 1046]:

Prediction equations for the extension [30]

$$\nu_{k|k-1} = e^{-\frac{T}{\tau}} \nu_{k-1|k-1}, \tag{2.11}$$
$$X_{k|k-1} = \frac{e^{-\frac{T}{\tau}} \nu_{k-1|k-1} - d - 1}{\nu_{k-1|k-1} - d - 1} X_{k-1|k-1}. \tag{2.12}$$

This prediction, however, does not ensure that $X_{k|k-1}$ is always positive definite, therefore an advanced version [33, p. 2409] can be used as an alternative:

Alternative prediction equations for the extension [33]

$$\nu_{k|k-1} = 6 + \frac{\alpha_k}{1+\alpha_k}(\nu_{k-1|k-1} - 6), \tag{2.13}$$

$$X_{k|k-1} = \frac{\alpha_k}{1+\alpha_k} X_{k-1|k-1}. \tag{2.14}$$

Both tuning parameters, τ and α_k enable a trade-off between a small stationary error and a dynamic behavior in a turn.

Update The update for the kinematic state $\underline{x}_k$ is similar to the update within the linear Kalman filter, but slightly adjusted to accommodate for multiple measurements being received at each time step. For the following, $\bar{\underline{z}}_k$ is the measurement's mean and Z_k is the covariance matrix of the current set of measurements $\boldsymbol{Z}_k = \{\underline{z}_k^1, \underline{z}_k^2, ..., \underline{z}_k^{n_k}\}$.

$$\bar{\underline{z}}_k = \frac{1}{n_k} \sum_{j=1}^{n_k} \underline{z}_k^j, \tag{2.15}$$

$$Z_k = \sum_{j=1}^{n_k} (\underline{z}_k^j - \bar{\underline{z}}_k)(\underline{z}_k^j - \bar{\underline{z}}_k)^T. \tag{2.16}$$

The kinematic state update is given for d dimensions by:

Update equations for the kinematic state [30]

$$\underline{x}_{k|k} = \underline{x}_{k|k-1} + (W_{k|k-1} \otimes I_d)(\bar{\underline{z}}_k - (H_k \otimes I_d)\underline{x}_{k|k-1}), \tag{2.17}$$

$$P_{k|k} = P_{k|k-1} - W_{k|k-1} S_{k|k-1} W_{k|k-1}^T, \tag{2.18}$$

with innovation factor $S_{k|k-1}$ and Kalman gain $W_{k|k-1}$:

$$S_{k|k-1} = H_k P_{k|k-1} H_k^T + \frac{\sigma_x^2}{n_k}, \tag{2.19}$$

$$W_{k|k-1} = P_{k|k-1} H_K^T S_{k|k-1}^{-1}, \tag{2.20}$$

where n_k is the number of measurements and σ_x^2 is the covariance of the measurement noise. Note that measurement noise was not considered in the original approach [30]. Equation (2.19) has been modified to include measurement noise. To update the parameters of the inverse Wishart density, the following equations are given [30, p. 1048]:

Update equations for the extent state [30]

$$X_{k|k} = X_{k|k-1} + N_{k|k-1} + Z_k, \tag{2.21}$$

$$\nu_{k|k} = \nu_{k|k-1} + n_k. \tag{2.22}$$

with innovation matrix [30, p. 1048]:

$$N_{k|k-1} = S_{k|k-1}^{-1}(\bar{\underline{z}}_k - (H_k \otimes I_d)\underline{x}_{k|k-1})(\bar{\underline{z}}_k - (H_k \otimes I_d)\underline{x}_{k|k-1})^T. \tag{2.23}$$

There are extensions of the random matrix approach which exclude the measurement noise from the filtered covariance matrix [31, 32, 34, 35]. $\mathbb{E}(X_{k|k})$ (2.10) then represents only the covariance

of the measurement data caused by the object's extension.

Applications for the random matrix approach The random matrix approach was extended in several publications [31–37] and is nowadays a state-of-the-art filter for EOT. It was also integrated into the probability hypothesis density (PHD) filter [38] to allow multi-extended object tracking (MEOT). In previous works, the random matrix approach has already been applied to ship tracking using radar sensors [39][40]. It has been shown that an ellipse is a suitable shape approximation for ships [39, p. 8][40, p. 6626]. In [41], random matrices are combined with the generalized probabilistic data association (GPDA) filter. The method is also evaluated using vessel tracking with radar data [41, p. 967]. A combination of a lidar sensor with the random matrix approach can be found e.g. in [38, p. 5659]. There, the extensions of a bicyclist and a pedestrian are estimated. However, the surfaces of a bicyclist and a pedestrian are not that homogeneous as the surface of vessels observed from the side. Therefore the measurement distribution in [38, p. 5659, Figure 1] is closer to the measurement model assumed in [30] than the measurement model we assume for lidar in the next section. Non-uniformly distributed measurements were considered in [37] with a conditional Gaussian mixture model. The idea is to split the object into several areas with different measurement densities [37, p. 3819].

2.6. Random finite sets

Random finite sets (RFSs) are used in multi object tracking (MOT) to mathematically model the multi-object state $\boldsymbol{X}$ and the set of measurement $\boldsymbol{Z}$:

$$\boldsymbol{X} = \{\underline{x}^{(1)}, ..., \underline{x}^{(n)}\}, \tag{2.24}$$

$$\boldsymbol{Z} = \{\underline{z}^{(1)}, ..., \underline{z}^{(m)}\}, \tag{2.25}$$

whereas $n \in \mathcal{N}_0$ is the number of objects and $m \in \mathcal{N}_0$ the number of measurements. In an RFS, the cardinality is random but finite, just like the number of objects n and measurements m is random but finite. The RFS $\boldsymbol{X}$ can be empty ($\boldsymbol{X} = \emptyset$), contain a single state vector ($\boldsymbol{X} = \{\underline{x}\}$) or contain a finite number of n state vectors. The same is true for the RFS $\boldsymbol{Z}$, representing m measurements.
Finite set statistics (FISST) is introduced by Mahler in [42–44]. A RFS can be described by its cardinality distribution $\rho(n)$ and a family of symmetric joined distributions $p_n\left(\underline{x}^{(1)}, ..., \underline{x}^{(n)}\right)$ [45, p. 3408]. For a RFS $\boldsymbol{X}$, the FISST pdf $f(\boldsymbol{X})$ can be written as [45, p. 3408]:

$$f(\boldsymbol{X}) = f(\{\underline{x}^{(1)}, ..., \underline{x}^{(n)}\}) = n!\rho(n)p_n(\underline{x}^{(1)}, ..., \underline{x}^{(n)}). \tag{2.26}$$

The expression in (2.26) can also be represented by a case notation [46, p. 25]:

$$f(\boldsymbol{X}) = \begin{cases} f(\emptyset) & \text{if } \boldsymbol{X} = \emptyset \\ f(\{\underline{x}^{(1)}\}) & \text{if } \boldsymbol{X} = \{\underline{x}^{(1)}\} \\ f(\{\underline{x}^{(1)}, \underline{x}^{(2)}\}) & \text{if } \boldsymbol{X} = \{\underline{x}^{(1)}, \underline{x}^{(2)}\} \\ \vdots & \vdots \end{cases} \tag{2.27}$$

Bernoulli RFS The pdf of a Bernoulli RFS is given by [45, p. 3408]:

$$f(\boldsymbol{X}) = \begin{cases} 1 - q & \text{if } \boldsymbol{X} = \emptyset \\ q \cdot p(\underline{x}) & \text{if } \boldsymbol{X} = \underline{x} \end{cases}, \tag{2.28}$$

whereas q is the probability of existence for the object and $p(\underline{x})$ the spatial density. The Bernoulli RFS can contain either the empty set $\emptyset$ or a single element $\{\underline{x}\}$. By assuming multiple independent objects, the Bernoulli RFS can be extended to a Multi-Bernoulli RFS [46, p. 28].

Labeled multi-Bernoulli RFS Labels can be used to uniquely identify each state vector within the multi-object state. The labeled state is consequently given as $\underline{x} = (\underline{x}, l)$ with the unlabeled state $\underline{x}$ and the label $l \in \mathbb{L}$ [46, p. 29], where $\mathbb{L}$ is the label space. The labeled multi-Bernoulli (LMB) RFS is characterized by the following parameter set [46, p. 30]

$$f(\boldsymbol{X}) = \{(q^{(l)}, p^{(l)}(\boldsymbol{x}))\}_{l \in \mathbb{L}}. \tag{2.29}$$

Since each labeled component has its own probability of existence $q^{(l)}$ and its own spatial density $p^{(l)}$, all components are assumed to be statistically independent [46, p. 31] but defined within the same state space.

Generalized labeled multi-Bernoulli RFS If the condition that components are independent cannot be fulfilled, then a generalized labeled multi-Bernoulli (GLMB) RFS must be used. The GLMB RFS is described by weights and spatial densities of hypotheses $c \in \mathbb{C}$ instead of components. The FISST pdf of a GLMB RFS is given by: [46, p. 31] [47, p. 3248]

$$f(\boldsymbol{X}) = \Delta(\boldsymbol{X}) \sum_{c \in \mathbb{C}} w^{(c)}(\mathcal{L}(\boldsymbol{X}))[p^{(c)}]^{\boldsymbol{X}} \tag{2.30}$$

where $w^{(c)}(\mathcal{L}(\boldsymbol{X}))$ are the weights of the hypotheses and $p^{(c)}$ are the spatial densities. The notation $[p^{(c)}]^{\boldsymbol{X}}$ represents the multi-object exponential [48, p. 3464], which is defined for a function $h(\underline{x})$ as $h^{\boldsymbol{X}} = \prod_{\underline{x} \in \boldsymbol{X}} h(\underline{x})$ [49, p. 918]. $\Delta(\boldsymbol{X})$ is the *distinct label indicator*, ensuring that labels are unique and that the number of labels corresponds to the cardinality of the multi-object state [46, p. 31] [47, p. 3248]. This uniqueness is already ensured in (2.30) by $l \in \mathbb{L}$.
In an implementation, the GLMB RFS has the disadvantage that the spatial densities of objects appear in multiple hypotheses, leading to additional computational effort [46, p. 32]. To optimize and improve computational efficiency, the δ-GLMB RFS can be used [48, p. 3464-3466].
Most MOT methods based on RFSs can either be categorized as Bernoulli filters or PHD filters. Both types of filters are used in this work and are therefore presented in the following subsections.

2.7. Bernoulli filter

Bernoulli filters can be used for single object tracking in clutter [45] or for MOT [42, 43, 46, 47, 50]. The single object Bernoulli propagates a Bernoulli RFS and therefore assumes that an object either exists or not exists. If it exists at the current time step, then it will still exist at the next time step with probability of survival p_s. If the object is not existing, a newborn object is modeled by the birth probability p_b and a spatial birth density $b(\underline{x})$, representing the distribution of initial states.
The Bernoulli measurement model considers clutter and true measurements $\boldsymbol{Z} = \boldsymbol{C} \cup \boldsymbol{W}$ [45, p. 3412], whereas the set of true measurements $\boldsymbol{W}$ is either empty or contains a single measurement with probability of detection p_d. The number of clutter measurements follows a Poisson distribution [45, p. 3412] and the measurements are typically uniformly distributed over the entire observation area. A tutorial on Bernoulli filters is given by Ristic et al. in [45], covering the derivation of the single object Bernoulli filter and its Gaussian mixture implementation for linear transition and measurement models.
The multi-target multi-Bernoulli (MeMBer) filter [42, p. 655-682] predicts and updates a multi-Bernoulli RFS, containing several independent tracks. Each track has a unique identification label, a probability of existence and a spatial density representing the kinematic state [42, p. 657]. During the update step for the MeMBer filter, every track is updated with every measurement and the approximation within the calculation of their probability of existence leads to a cardinality bias. To overcome this issue, the cardinality balanced multi-target multi-bernoulli (CBMeMBer) was proposed by Vo et al. [50]. The CBMeMBer update compensates most of the bias that is caused by the approximation within the track update. However, since it also assumes a low

clutter rate, a cardinality bias is still observable in high-clutter scenarios [M1].
A further improvement for the update step was proposed by Reuter et al. by introducing the LMB filter. Since an adaptive birth density for the LMB filter was presented in the first conference paper [PH7] published from this dissertation, the LMB filter is briefly explained in the following section. The adaptive birth density is presented in section 7 in the methodological part of this thesis.

2.8. The labeled multi-Bernoulli filter

The LMB filter propagates an LMB RFS, that is transformed to a GLMB RFS for the update step. Fig 1 shows the structure of the LMB filter:

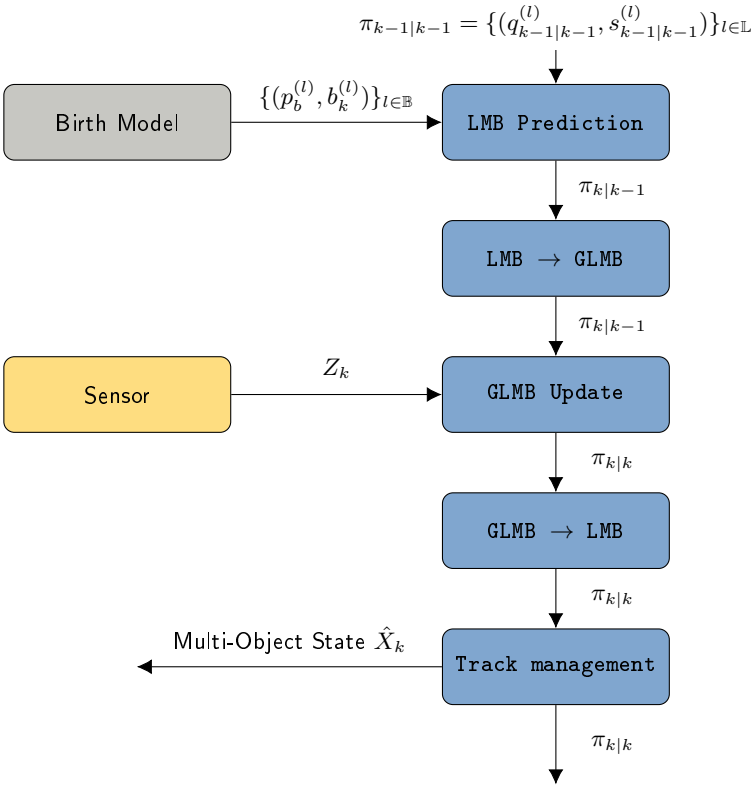


Figure 1: LMB Filter (following [47, p. 3253]) [PH7].

The LMB filter recursion starts with the posterior density in LMB form with a variable number of LMB components. Each component consists of a label l, a probability of existence $q^{(l)}$ and a spatial density $s^{(l)}$ and represents a single track. In the prediction step, each track is propagated independently, and birth tracks for the current time step are included. The update step is performed in GLMB form. Instead of updating individual tracks, each possible hypothesis is updated. This approach is necessary because tracks are not independent, as a measurement can be assigned to only one track at a time. After the update, the GLMB RFS is approximated by an LMB RFS to improve computational efficiency. In the final recursion step, tracks that meet the detection criteria are extracted, and methods are applied to manage the number of tracks effectively.

Prediction The predicted LMB RFS is the union of the previous posterior LMB RFS and the birth LMB RFS [47, p. 3253]):

$$\pi_{k|k-1} = \{(q^{(l)}_{k|k-1}, s^{(l)}_{k|k-1})\}_{l\in\mathbb{L}} \cup \{(p^{(l)}_b, b^{(l)}_k)\}_{l\in\mathbb{B}}. \tag{2.31}$$

In a Gaussian mixture (GM) implementation, the probabilities of existence $q^{(l)}$ and spatial densities $s^{(l)}$ can be predicted using [46, p. 105]:

$$q_{k|k-1}^{(l)} = p_s q_{k-1|k-1}^{(l)}, \tag{2.32}$$

$$s_{k|k-1}^{(l)} = \sum_{e=1}^{N} w^{(l,e)} \mathcal{N}(\underline{x}; \underline{m}_{k|k-1}^{(l,e)}, P_{k|k-1}^{(l,e)}). \tag{2.33}$$

where p_s is the probability that an object survives between the time steps $k-1$ and k. Each GM component is predicted individually using the standard Kalman filter equations, as described in Section 2.4.

GLMB transformation To prepare for the update step, the LMB RFS is transformed into a GLMB RFS, which requires calculating the weights $w^{(I)}$ for each hypothesis I [46, p. 114]:

$$w_{k|k-1}^{(I)} = \prod_{j \in \mathbb{L}} (1 - q_{k|k-1}^{(j)}) \prod_{l \in I} \frac{q_{k|k-1}^{(l)}}{1 - q_{k|k-1}^{(l)}}. \tag{2.34}$$

Since the number of hypotheses increases exponentially with the number of tracks, measures must be taken to retain only the most important ones [46, p. 115][47, p. 3254], e.g., by using a k-shortest paths algorithm [51, 52] or stochastic sampling [46, p. 115]. The spatial densities remain unchanged during the GLMB transformation; however, the corresponding tracks are assigned to the hypotheses that include them.

GLMB update The GLMB update can be performed independently for the spatial densities and hypothesis probabilities.
To update the spatial densities, all possible combinations of existing tracks and measurements are evaluated. Considering missed detections as well, a total of $mL+L$ tracks are generated for L tracks and m measurements. When approximating the result to the LMB form, tracks with the same label are merged. This reduces the number of tracks back to L, ensuring it remains constant during the measurement update. In a GM implementation, the mL measurement-updated tracks are given as [46, p. 109]:

$$\underline{\eta}_{k|k-1}^{(l,e)} = H \underline{m}_{k|k-1}^{(l,e)}, \tag{2.35}$$

$$S_{k|k-1}^{(l,e)} = H P_{k|k-1}^{(l,e)} H^T + R, \tag{2.36}$$

$$K_k^{(l,e)} = P_{k|k-1}^{(l,e)} H^T [S_{k|k-1}^{(l,e)}]^{-1}, \tag{2.37}$$

$$P_{k|k}^{(l,e)} = P_{k|k-1}^{(l,e)} - K_k^{(l,e)} H^T P_{k|k-1}^{(l,e)}, \tag{2.38}$$

$$\underline{m}_{k|k}^{(l,e,j)} = \underline{m}_{k|k-1}^{(l,e)} + K_k^{(l,e)} (\underline{z}_k^{(j)} - \underline{\eta}_{k|k-1}^{(l,e)}). \tag{2.39}$$

The L miss-detection tracks are simply the predicted tracks:

$$P_{k|k}^{(l,e)} = P_{k|k-1}^{(l,e)}, \tag{2.40}$$

$$\underline{m}_{k|k}^{(l,e)} = \underline{m}_{k|k-1}^{(l,e)}. \tag{2.41}$$

During the hypothesis update, the weights $w_{k|k}^{(I,\theta)}$ of the posterior hypotheses (I, θ) are calculated. I is the hypothesis from the GLMB transformation, indicating which objects exist, and θ represents the measurement association for the update step. In general, tracks can be assigned to a measurement or a missed detection. Because the number of possible assignments grows rapidly, only the k most important assignments are computed [46, p. 117]. The calculation is performed

using Murty's algorithm [53]. In the first step, a cost matrix is constructed, which is given as [46, p. 117]:

$$C_Z^I = \begin{pmatrix} \tilde{\eta}^{(1)}(l_1) & \dots & \tilde{\eta}^{(m)}(l_1) & \tilde{\eta}^{(0)}(l_1) & \dots & \tilde{\eta}^{(0)}(l_1) \\ \tilde{\eta}^{(1)}(l_2) & \dots & \tilde{\eta}^{(m)}(l_2) & \tilde{\eta}^{(0)}(l_2) & \dots & \tilde{\eta}^{(0)}(l_2) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \tilde{\eta}^{(1)}(l_n) & \dots & \tilde{\eta}^{(m)}(l_n) & \tilde{\eta}^{(0)}(l_n) & \dots & \tilde{\eta}^{(0)}(l_n) \end{pmatrix}. \tag{2.42}$$

The cost matrix has n rows, one for each track, and $m + n$ columns [46, p. 117]. The first m columns contain the costs of assigning tracks to measurements [46, p. 117]. The remaining n columns represent the costs of missed detections, allowing each track to be assigned to a missed detection hypothesis [46, p. 117]. The elements of the cost matrix are computed using the weight of the predicted hypothesis $w_{k|k-1}$ and the measurement assignment likelihood η_z [46, p. 117]:

$$\tilde{\eta}^{(i)}(l_n) = -\log\big(w_{k|k-1}(I) \cdot \eta_z^{(\theta)}(l_n)\big), \tag{2.43}$$

$$\eta_z^{(\theta)}(l) = \frac{p_d}{\kappa(z_{\theta(l)})} \sum_{e=1}^{N} w_{k|k-1}^{(l,e)} \mathcal{N}(\underline{z}_{\theta(l)}; \underline{z}_{k|k-1}^{(l,e)}, S^{(l,e)}). \tag{2.44}$$

For an assignment to a missed detection hypothesis, η_z is given by $1 - p_d$ [46, p. 110]. The total cost of a global assignment hypothesis is obtained by summing the costs of all individual assignments [46, p. 117]:

$$c_Z^{(I)}(\theta^l) = \sum_{i=1}^{|I|} C_Z^{(I)}(i, \theta^l(l_i)). \tag{2.45}$$

Since equation (2.43) defines costs in logarithmic form, the weight for the hypothesis (I, θ) is determined using the exponential function [46, p. 118]:

$$\tilde{w}^{(I,\theta)}(\boldsymbol{Z}) = \exp\big(-c_Z^{(I)}(\theta^l)\big). \tag{2.46}$$

Finally, the weights must be normalized [46, p. 118]:

$$w_{k|k}^{(I,\theta^l)} = \frac{\tilde{w}^{(I,\theta)}(\boldsymbol{Z})}{\sum_I \sum_\theta \tilde{w}^{(I,\theta)}(\boldsymbol{Z})}. \tag{2.47}$$

LMB approximation To convert the update result back to the initial form of the recursion, it must be approximated as an LMB RFS. This requires to calculate the existence probabilites of the posterior tracks by summing up the weights of all hypotheses that contain the corresponding track, identified by its label [46, p. 97]:

$$q_{k|k}^{(l)} = \sum_{L(I,\theta)\ni l} w_{k|k}^{(I,\theta)}(\boldsymbol{Z}). \tag{2.48}$$

The spatial densities of the individual tracks result from the weighted sum of the spatial densities of the tracks that are assigned to a hypothesis containing the track label l [46, p. 97]:

$$s_{k|k}^{(l)} = \frac{1}{q_{k|k}^{(l)}} \sum_{L(I,\theta)\ni l} w_{k|k}^{(I,\theta)}(\boldsymbol{Z}) s_{k|k}^{(\theta,l)}. \tag{2.49}$$

After the LMB approximation, the posterior $\pi_{k|k} = \{(q_{k|k}^{(l)}, s_{k|k}^{(l)})\}_{l\in\mathbb{L}}$ may have too many components to be computational feasible. Therefore, components with low weights have to be pruned and components that are close to each other have to be merged.

2.9. Probability hypothesis density filter

The probability hypothesis density (PHD) filter, introduced by Mahler [54], is an alternative approach to multi-Bernoulli filters for MOT, and operates by approximating the multi-object state through a function of the single-object state, known as the PHD $D(\underline{x})$. Unlike a pdf, integrating the PHD over the entire state space does not yield unity. Instead, it yields the expected cardinality i.e. the expected number of objects [54, p. 1152]. It is further assumed that the cardinality is Poisson distributed and that the elements within the PHD are independent and identically distributed (IID) [55, p. 5702][M2, p. 19].
One common practical implementation of the PHD filter is the GM-PHD filter [56], where the density is represented by a weighted sum of Gaussian distributions:

$$D(\underline{x}) = \sum_{i=1}^{I} w^{(i)} \mathcal{N}\left(\underline{x}; \underline{m}^{(i)}, P^{(i)}\right), \tag{2.50}$$

with weights $w^{(i)}$, means $\underline{m}^{(i)}$, and covariance matrices $P^{(i)}$. The key difference from GM-multi-Bernoulli filters lies in the absence of explicit track assignments for each Gaussian component. The Bayesian recursion of the PHD filter is illustrated in Figure 2.

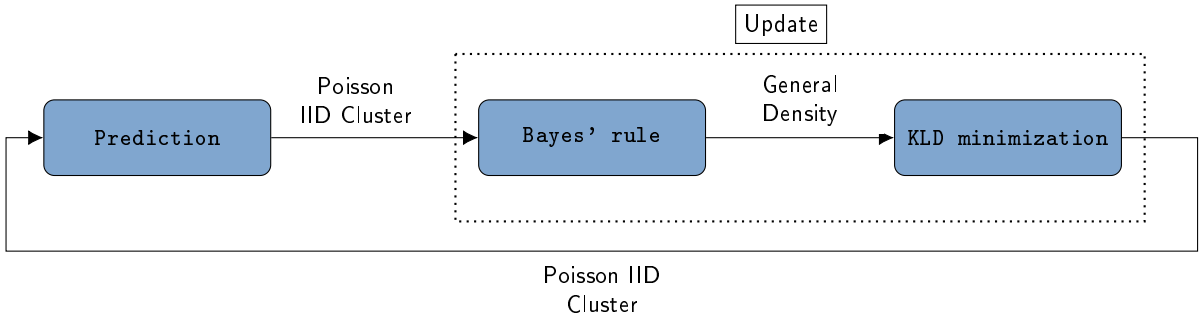


Figure 2: Bayesian recursion of the PHD filter. Figure adapted from [55, p. 5703] and [M2, p. 5703].

The posterior PHD resulting from the measurement update step generally loses the Poisson IID property. Hence, it must be approximated back to a Poisson IID cluster using Kullback-Leibler divergence (KLD) minimization.

Prediction The prediction for the PHD filter is given by [54, p. 1167][56, p. 4094]:

$$D_{k|k-1} = \int p_s p(\underline{x}_k|\underline{x}_{k-1}) D_{k-1|k-1} + b_k(\underline{x}), \tag{2.51}$$

where p_s is the probability of survival, $p(\underline{x}_k|\underline{x}_{k-1})$ denotes the transition density and $b_k(\underline{x})$ represents the birth intensity.

Update The update step of the PHD filter incorporates new measurements into the predicted intensity, resulting in the posterior intensity given by [56, p. 4094]:

$$D_{k|k}(\underline{x}_k) = (1 - p_d) D_{k|k-1}(\underline{x}_k) + \sum_{\underline{z}_k \in \boldsymbol{Z}_k} \frac{p_d \, p(\underline{z}_k|\underline{x}_k) \, D_{k|k-1}(\underline{x}_k)}{\kappa_k(\underline{z}_k) + \int p_d \, p(\underline{z}_k|\underline{x}_k) \, D_{k|k-1}(\underline{x}_k) \, d\underline{x}_k}, \tag{2.52}$$

where p_d is the detection probability, $p(\underline{z}_k|\underline{x}_k)$ the measurement likelihood, and $\kappa_k(\underline{z}_k)$ the clutter intensity. The main advantage of the PHD filter update is its computational efficiency compared to multi-Bernoulli-based methods. However, as it approximates Bayes' rule directly on intensity

level, it generally offers lower accuracy compared to more complex methods, such as the GLMB update within the LMB filter. A widely used practical implementation is the (GM)-PHD, the details of which are presented later with the trajectory probability hypothesis density (TPHD) filter in section 2.11. Before discussing the TPHD filter, a brief theoretical background on sets of trajectories is provided.

2.10. Accumulated state densities and sets of trajectories

The joint densities of all state vectors up to the current time step, denoted as accumulated state densitys (ASDs), are addressed in numerous publications [57–59]. The accumulated state from time t_k back to t_n is defined in [58, p. 2202] as

$$\underline{x}_{k:n} = \left(\underline{x}_k, \underline{x}_{k-1}, \ldots, \underline{x}_n\right), \tag{2.53}$$

which describes a trajectory of length $n+1$. The accumulated state density corresponds to the joint probability density function $p(\underline{x}_{k:n}|\underline{z}_{1:k})$. This formulation has the advantage of preserving all correlations between the object's states at different time steps [58, p. 2202]. A typical application of ASDs is the handling of out-of-sequence (OOS) measurements [58].
Sets of trajectories can be interpreted as a representation of ASDs within the framework of RFSs [PH2]. A trajectory is defined as a sequence of single-target states. In the context of multi-object tracking, where objects can appear and disappear over time, trajectories have explicit start and end times. Formally, a trajectory is defined as [60, p. 1687]:

$$\mathcal{X} = (t, \underline{x}_{1:\iota}) = (t, \underline{x}_1, ..., \underline{x}_\iota) \tag{2.54}$$

with initial time step t and length ι. Therefore, the trajectory exists from time step t to time step $t+\iota-1$ [60, p. 1687]. If the final time step $t+\iota-1$ is equal to the current time step k, the trajectory is referred to as an *alive trajectory*.
The set of all trajectories is denoted by $\boldsymbol{\mathcal{X}}$ and is an RFS that can be interpreted as the trajectory version of (2.24). From a given set of trajectories, the multi-object state at a specific time step k can be extracted using the extraction function [60, p. 1687]:

$$\tau_k(\boldsymbol{\mathcal{X}}) = \bigcup_{\mathcal{X}\in\boldsymbol{\mathcal{X}}} \tau_k(\mathcal{X}), \tag{2.55}$$

$$\tau_k(\mathcal{X}) = \begin{cases} \{\underline{x}_{k+1-t}\} & t \leq k \leq t+\iota-1 \\ \emptyset & \text{elsewhere} \end{cases}. \tag{2.56}$$

Multi-trajectory densities The single trajectory density is denoted by $\pi(\mathcal{X})$, or equivalently by $\pi(t, x_{1:\iota})$ [55, p. 5703][M2, p. 37]. Its integral over the space of all trajectories must equal one and is given by [55, p. 5703]:

$$\int \pi(\mathcal{X})d\mathcal{X} = \sum_{(t,\iota)\in I_{(k)}} \int \pi(t, x_{1:\iota})dx_{1:\iota} = 1 \tag{2.57}$$

with $I_{(k)} = \{(t,\iota) : 0 \leq t \leq k \text{ and } 1 \leq \iota \leq k-t+1\}$ [55, p. 5703]. This set integral corresponds to the sum over all possible start times and lengths of trajectories, with integration over the corresponding state sequences. The multi-trajectory density is denoted as $\pi(\boldsymbol{\mathcal{X}})$. Within the PHD filter recursion, the posterior is modeled as a Poisson multi-trajectory density. Such a density is defined as [55, p. 5704]:

$$\pi(\{\mathcal{X}^{(1)}, ..., \mathcal{X}^{(n)}\}) = e^{-\lambda_\pi}\lambda_\pi^n \prod_{j=1}^{n} \pi(\mathcal{X}^{(j)}), \tag{2.58}$$

where $\lambda_\pi \geq 0$ denotes the expected number of trajectories.

Bayesian recursion for sets of trajectories The Bayesian recursion for sets of trajectories follows the classical two-step structure consisting of prediction and update [55, p. 5705][M2, p. 38]:

$$\pi(\boldsymbol{\mathcal{X}}_k|\boldsymbol{Z}_{1:k-1}) = \int p(\boldsymbol{\mathcal{X}}_k|\boldsymbol{\mathcal{X}}_{k-1})\pi(\boldsymbol{\mathcal{X}}_{k-1}|\boldsymbol{Z}_{1:k-1})d\boldsymbol{\mathcal{X}}_{k-1}, \tag{2.59}$$

$$\pi(\boldsymbol{\mathcal{X}}_k|\boldsymbol{Z}_{1:k}) = \frac{p(\boldsymbol{Z}_k|\tau_k(\boldsymbol{\mathcal{X}}_k))\pi(\boldsymbol{\mathcal{X}}_k|\boldsymbol{Z}_{1:k-1})}{\int p(\boldsymbol{Z}_k|\tau_k(\boldsymbol{\mathcal{X}}_k))\pi(\boldsymbol{\mathcal{X}}_k|\boldsymbol{Z}_{1:k-1})d\boldsymbol{\mathcal{X}}_k}. \tag{2.60}$$

In the prediction (2.59), $\pi(\boldsymbol{\mathcal{X}}_{k-1}|\boldsymbol{Z}_{1:k-1})$ denotes the posterior multi-trajectory density from the previous time step, $p(\boldsymbol{\mathcal{X}}_k|\boldsymbol{\mathcal{X}}_{k-1})$ is the multi-trajectory transition density, and $\pi(\boldsymbol{\mathcal{X}}_k|\boldsymbol{Z}_{1:k-1})$ is the predicted density [55, p. 5705][M2, p. 38].
The update step (2.60) incorporates the measurement likelihood $p(\boldsymbol{Z}_k|\tau_k(\boldsymbol{\mathcal{X}}_k))$, which depends only on the current multi-object state. Since the measurement set $\boldsymbol{Z}_k$ at time step k is assumed to be conditionally independent of past states given the current one, the extraction function $\tau_k(\boldsymbol{\mathcal{X}})$ (Equation 2.55) is used to obtain the current multi-object state from the set of trajectories [55, p. 5705][M2, p. 38]. One concrete implementation of the Bayesian recursion over sets of trajectories is the trajectory probability hypothesis density (TPHD) filter, which approximates the multi-trajectory posterior by a Poisson point process.

2.11. Trajectory probability hypothesis density filter

The trajectory probability hypothesis density (TPHD) filter is one of the most computationally efficient filters based on sets of trajectories. It was first introduced in [61] and further detailed in [55]. The TPHD filter builds upon the standard PHD filter described in Section 2.9 and inherits its underlying assumptions. However, several modifications are required to accommodate the representation of trajectories. Similar to the PHD prediction, the prediction result of the TPHD filter consists of two components: surviving trajectories $D^{(s)}_{k-1|k-1}(\boldsymbol{\mathcal{X}})$ and newborn trajectories $D^{(b)}_{k-1|k-1}(\boldsymbol{\mathcal{X}})$. The adjusted prediction equation for surviving trajectories is given by [55, p. 5706]:

$$D^{(s)}_{k|k-1}(t, \underline{x}_{1:\iota}) = p_s g(\underline{x}_\iota|\underline{x}_{\iota-1}) D_{k-1|k-1}(t, \underline{x}_{1:\iota-1}) 1_{\mathbb{N}_{k-1}(t)} \tag{2.61}$$

with posterior PHD $D_{k-1|k-1}$, and $t+\iota-1 = k$ since trajectories are considered to be alive. The indicator $\mathbb{N}_k = \{1, ..., k\}$ ensures that the transition accesses only the last state of the trajectory [M2, p. 39]. Newborn components are added similarly than in the PHD filter and are basically just trajectories with length $\iota = 1$.
The update step of the TPHD filter is performed by multiplying the predicted PHD with a pseudolikelihood function $L_{\underline{z}_k}$ [55, p. 5706][M2, p. 39]

$$D_{k|k}(t, \underline{x}_{1:\iota}) = D_{k|k-1}(t, \underline{x}_{1:\iota}) L_{\underline{z}^k}(\underline{x}_{1:\iota}). \tag{2.62}$$

The pseudolikelihood function is defined as [55, p. 5706]:

$$L_{\underline{z}_k}(\underline{x}) = 1 - p_d + p_d \sum_{\underline{z} \in \boldsymbol{Z}} \frac{l(\underline{z}|\underline{x})}{\kappa_k(\underline{z}) + \int p_d l(\underline{z}|\underline{\zeta}) D_{\tau,k|k-1}(\zeta) d\underline{\zeta}} \tag{2.63}$$

where ζ is the previous state and κ_k is the clutter density.

Gaussian mixture implementation In the Gaussian mixture (GM) implementation, the posterior PHD is assumed to be a GM, as given in [55, p. 5708][M2, p. 39]:

$$D_{k-1|k-1}(\boldsymbol{\mathcal{X}}) = \sum_{i=1}^{I} w_{k-1|k-1}^{(i)} \mathcal{N}\left(\boldsymbol{\mathcal{X}}; t_{k-1}^{(i)}, \underline{m}_{k-1|k-1}^{(i)}, P_{k-1|k-1}^{(i)}\right). \tag{2.64}$$

where each component is a Gaussian distribution over a trajectory hypothesis, defined by the birth time $t_{k-1}^{(i)}$, the mean vector $\underline{m}_{k-1|k-1}^{(i)} \in \mathbb{R}^{n_x \iota_{k-1|k-1}^{(i)}}$, and the covariance matrix $P_{k-1|k-1}^{(i)} \in \mathbb{R}^{n_x \iota_{k-1|k-1}^{(i)} \times n_x \iota_{k-1|k-1}^{(i)}}$, with n_x state dimensions and trajectory length $\iota_{k-1|k-1}^{(i)}$. The vector $\underline{m}_{k-1|k-1}^{(i)}$ contains a concatenation of all single-target state vectors of the trajectory. Thus, the trajectory length is given by [55, p. 5708][M2, p. 39]:

$$\iota_{k-1|k-1}^{(i)} = \dim(\underline{m}_{k-1|k-1}^{(i)})/n_x. \tag{2.65}$$

The GM prediction follows the general structure of (2.61) and is given by [55, p. 5708][p. 39][M2]:

$$D_{k|k-1}(\boldsymbol{\mathcal{X}}) = D_{k|k-1}^{(b)}(\boldsymbol{\mathcal{X}}) + p_s \sum_{i=1}^{I} w_{k-1|k-1}^{(i)} \mathcal{N}\left(\boldsymbol{\mathcal{X}}; t_{k-1}^{(i)}, \underline{m}_{k|k-1}^{(i)}, P_{k|k-1}^{(i)}\right) \tag{2.66}$$

with

$$\underline{m}_{k|k-1}^{(i)} = \left[{\underline{m}_{k-1|k-1}^{(i)}}^{\top}, (F_k^{(i)} \underline{m}_{k-1|k-1}^{(i)})^{\top} \right]^{\top}, \tag{2.67}$$

$$F_k^{(i)} = \left[0_{1,\iota_{k-1|k-1}^{(i)}-1}, 1 \right] \otimes F, \tag{2.68}$$

$$P_{k|k-1}^{(i)} = \begin{bmatrix} P_{k-1|k-1}^{(i)} & P_{k-1|k-1}^{(i)} {F_k^{(i)}}^{\top} \\ F_k^{(i)} P_{k-1|k-1}^{(i)} & F_k^{(i)} P_{k-1|k-1}^{(i)} {F_k^{(i)}}^{\top} + Q_k \end{bmatrix}. \tag{2.69}$$

In Equation (2.68), the matrix $F_k^{(i)}$ is constructed via a Kronecker product to expand the single-target transition matrix F according to the trajectory length $\iota_{k-1|k-1}^{(i)}$. This is achieved using an access vector composed of $\iota_{k-1|k-1}^{(i)} - 1$ zeros followed by a one, which selects the last state in the trajectory for propagation. For instance, if the trajectory length is $\iota = 3$, the access vector is $[0, 0, 1]$, resulting in a multiplication of F with the final state of the trajectory [M2, p. 40]. The result of the prediction in Equation (2.66) can be written in compact form as a new Gaussian mixture [M2, p. 40]:

$$D_{k|k-1}(\boldsymbol{\mathcal{X}}) = \sum_{i=1}^{I} w_{k|k-1}^{(i)} \mathcal{N}\left(\boldsymbol{\mathcal{X}}; t_k^{(i)}, \underline{m}_{k|k-1}^{(i)}, P_{k|k-1}^{(i)}\right). \tag{2.70}$$

The posterior PHD after the update step is given by [55, p. 5708][M2, p. 40]:

$$D_{k|k}(\boldsymbol{\mathcal{X}}) = (1 - p_d) D_{k|k-1}(\boldsymbol{\mathcal{X}}) + \sum_{\underline{z} \in \boldsymbol{Z}_k} \sum_{i=1}^{I} w_{k|k}^{(i)}(\underline{z}) \mathcal{N}\left(\boldsymbol{\mathcal{X}}, t_k^{(w,i)}, \underline{m}_{k|k}^{(i)}(\underline{z}), P_{k|k}^{(i)}\right). \tag{2.71}$$

The first term accounts for missed detections, where the predicted component weights are scaled by the factor $(1 - p_d)$. For measurement-updated components, an expanded measurement matrix $H_k^{(i)}$ is constructed using the Kronecker product to apply the measurement update to the last

state in the trajectory: [55, p. 5708][M2, p. 40]:

$$H_k^{(i)} = \left[0_{1,\iota_{k|k-1}^{(i)}-1}, 1\right] \otimes H_k, \tag{2.72}$$

$$\underline{z}_{k|k-1}^{(i)} = H_k^{(i)} \underline{m}_{k|k-1}^{(i)}, \tag{2.73}$$

$$S_{k|k-1}^{(i)} = H_k^{(i)} P_{k|k-1}^{(i)} {H_k^{(i)}}^\top + R_k, \tag{2.74}$$

$$K_k^{(i)} = P_{k|k-1}^{(i)} {H_k^{(i)}}^\top \left(S_{k|k-1}^{(i)}\right)^{-1}, \tag{2.75}$$

$$\underline{m}_{k|k}^{(i)}(\underline{z}) = \underline{m}_{k|k-1}^{(i)} + K_k^{(i)} \left(\underline{z} - \underline{z}_{k|k-1}^{(i)}\right), \tag{2.76}$$

$$P_{k|k}^{(i)} = P_{k|k-1}^{(i)} - K_k^{(i)} H_k^{(i)} P_{k|k-1}^{(i)}. \tag{2.77}$$

Finally, the updated weights for the measurement components are computed as: [55, p. 5708][M2, p. 41]:

$$w_{k|k}^{(i)}(\underline{z}) = \frac{p_d w_{k|k-1}^{(i)} \mathcal{N}\left(\underline{z}; \underline{z}_{k|k-1}^{(i)}, S_{k|k-1}^{(i)}\right)}{\kappa_k(\underline{z}) + p_d \sum_{i=1}^{I} w_{k|k-1}^{(i)} \mathcal{N}\left(\underline{z}; \underline{z}_{k|k-1}^{(i)}, S_{k|k-1}^{(i)}\right)}. \tag{2.78}$$

To prevent an exponential growth in the number of Gaussian components, pruning and merging techniques are applied [61, p. 5][M2, p. 41]. Furthermore, to limit the memory and computational complexity associated with long trajectory histories, an L-scan implementation as proposed in [55, p. 5709] can be used, which restricts the trajectory length to the most recent L time steps [M2, p. 42].
In summary, the TPHD filter provides a computationally efficient framework for trajectory estimation by extending the PHD recursion to sets of trajectories, while retaining the tractability of GM implementations.

2.12. Gaussian process regression

A Gaussian process (GP) is a collection of random variables with a joint multivariate Gaussian distribution, defined by:

$$f(\underline{x}) \sim \mathcal{GP}\left(m(\underline{x}), k(\underline{x}, \underline{x}')\right) \tag{2.79}$$

where $m(\underline{x}) = \mathbb{E}\left[f(\underline{x})\right]$ is the mean function and the covariance function is given by [62, p. 13]:

$$k(\underline{x}, \underline{x}') = \mathbb{E}\left[(f(\underline{x}) - m(\underline{x}))(f(\underline{x}') - m(\underline{x}'))^\top\right]. \tag{2.80}$$

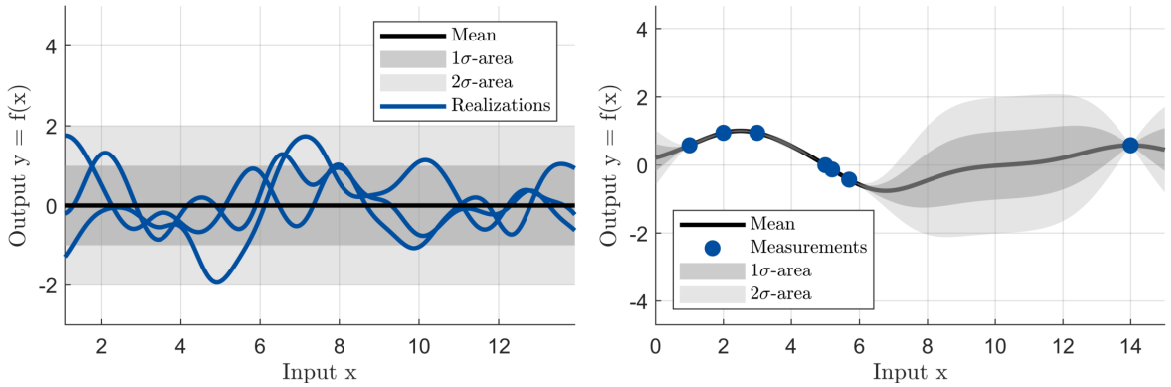
In many cases, the squared exponential function [62, p. 19]

$$k(\underline{x}_p, \underline{x}_q) = \sigma_f^2 \exp\left(-\frac{1}{2} \sum_{m=1}^{d} \frac{\left(\underline{x}_{pm} - \underline{x}_{qm}\right)^2}{2\ell^2}\right) + \sigma_n^2 \delta_{pq} \tag{2.81}$$

is used, since the normal distribution has the same kernel. The hyperparameters are signal variance σ_f^2, length-scale ℓ and the noise covariance σ_n^2 [62, p. 20]. Figure 3a shows a Gaussian Process with zero mean. Using the parameters $\sigma_f^2 = 1$, $\sigma_n^2 = 0$ and $\ell = 0.5$ within the squared exponential kernel(2.81), three realizations are drawn from the multivariate normal distribution $\mathcal{N}(X_*, \underline{0}, K(X_*, X_*))$. Note that the covariance matrix $K(X_*, X_*)$ is evaluated by calculating all correlations between all elements within the test data set X_* using 2.81 (see (2.84) for the structure).

For data with linear trends, however, the dot product kernel is the better choice [62, p. 80]:

$$k(\underline{x}_p, \underline{x}_q) = \underline{x}_p \underline{x}_q^\top + \sigma_n^2 \delta_{pq}. \tag{2.82}$$



(a) Realizations of a Gaussian Process (b) Posterior Gaussian process for a sine function

Figure 3: Gaussian Processes examples. Inspired by Rasmussen [62, p. 15]

The key to regression is the conditional distribution

$$p(Y_*|X_*, X, Y) = \mathcal{N}(X_*; K(X_*, X)K(X, X)^{-1}Y, \\ K(X_*, X_*) - K(X_*, X)K(X, X)^{-1}K(X, X_*)), \quad (2.83)$$

where X is the set of training data with corresponding outputs $Y = f(X)$ and X_* are test data points [62, p. 16]. The covariance matrices $K(X_*, X)$, $K(X, X)$ and $K(X, X_*)$ are computed by:

$$K(X^a, X^b) = \begin{pmatrix} k(x_1^a, x_1^b) & k(x_1^a, x_2^b) & \dots & k(x_1^a, x_n^b) \\ k(x_2^a, x_1^b) & k(x_2^a, x_2^b) & \dots & k(x_2^a, x_n^b) \\ \vdots & \vdots & \ddots & \vdots \\ k(x_n^a, x_1^b) & k(x_n^a, x_2^b) & \dots & k(x_n^a, x_n^b) \end{pmatrix} \quad (2.84)$$

using the squared exponential kernel (2.81). Figure 3b shows an example of a 1-dimensional posterior Gaussian process.
A sine function was observed at 7 points. For a smooth visualization, 1500 training points distributed uniformly in x were used. Figure 3b shows the mean function and the 1σ resp. 2σ area obtained by the conditional distribution (2.83).

Part I.
Extended Object Tracking with Virtual Measurement Models

3. Measurement and shape models

EOT methods differ significantly in how explicitly they represent the properties of generalized phy sical objects. They can be classified according to their assumed geometric shape or the underlying measurement likelihood model. The extent to which these methods incorporate physical realism versus maintaining a high level of generality is crucial for their applicability and accuracy. As an introductory example, a tracking scenario involving a boat observed by a lidar sensor illustrates three representative approaches (Figure 4), each highlighting different degrees of generalization and physical realism in modeling the shape and measurement distribution.

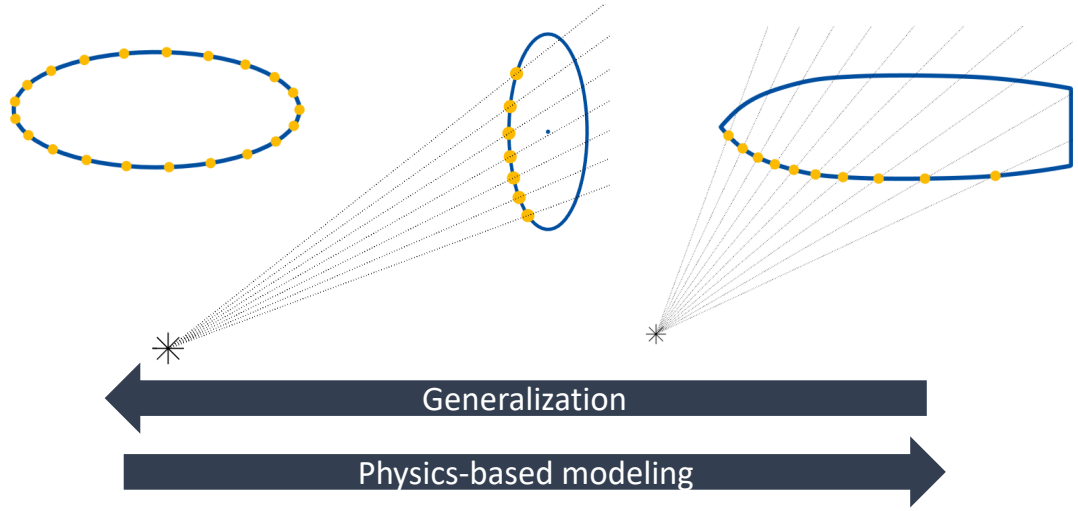
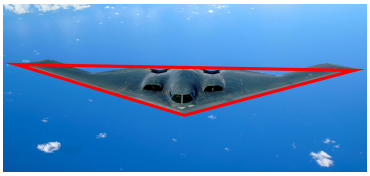


Figure 4: Physics-based modeling and generalization in tracking of boats with a lidar sensor.

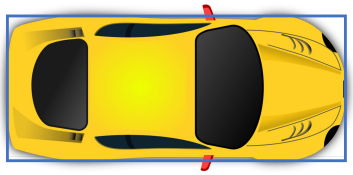
The first approach, shown on the left side of Figure 4, models the object's shape as an ellipse and assumes that lidar measurements are uniformly distributed along its contour. This combined shape and measurement model was, for example, used in [63]. The approach achieves a high level of generalization, as an ellipse provides a suitable approximation for a wide range of boats. Furthermore, due to the uniform distribution assumption, specific sensor details such as its exact position and orientation relative to the object do not need to be considered. However, several physical constraints are neglected, most notably that a lidar sensor only generates measurements from parts of the object's contour within its direct line of sight.
This limitation is addressed by the second approach, in which measurements are generated using ray tracing. Here, explicitly considering the sensor's position and physical constraints such as line-of-sight results in a more realistic measurement model. A ray tracing method is used in [PH1, PH3–PH5, 64] and will also be proposed in this section not just for ellipses but for further 2D and 3D shapes.
The next level of physics-based modeling could involve replacing the elliptical shape with the bird's-eye view contour extracted from a computer-aided design (CAD) model. While this provides an exact representation of the object's shape, it significantly reduces generalization since it is only applicable to one specific object.
As a compromise between physical realism and generalization, the central approach shown in section 4 is further pursued. It serves as an example of a generalized physical model.
To place this modeling choice within the broader context of existing literature, the remainder of this section provides an overview of commonly used shape models in both 2D and 3D EOT. This review highlights the prevalence of standard geometric representations and motivates the specific modeling choices adopted and developed in this work.
The most common shapes in EOT are ellipses [30–41, 64–73], rectangles [64, 74–76] and star-convex shapes [72, 76–78]. In 3D-EOT, commonly used shapes include ellipsoids [30, 33, 34, 38], cones [79], cylinders [80] or rectangular cuboids [PH2]. Figure 5 illustrates real-world examples

of the shapes used in this work.

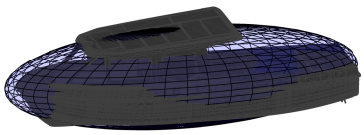
(a) 2D Ellipse [1].



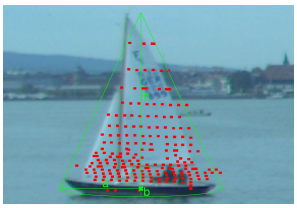
(b) 2D Triangle [2].



(c) 2D Rectangle [3].



(d) 3D Ellipsoid.



(e) 3D Cone [79].

(f) 3D Cuboid [4].

Figure 5: Object-specific shapes.

Two-dimensional shapes such as ellipses, triangles, and rectangles are often derived from the bird's-eye view of three-dimensional objects, particularly in scenarios where object height is irrelevant. In maritime environments, ellipses are the most common representation for various types of boats, including sailing boats and motorboats, while rectangular shapes are sometimes more appropriate for ferries. In 3D maritime EOT, different vessel types are typically modeled using ellipsoids for motorboats, cones for sailing boats, and cuboids for ferries. A similar approach is used in automotive applications, where cars, buses, and trams are commonly represented as 2D rectangles or 3D cuboids. Shape information can thus provide valuable cues for object classification. On the following pages, mathematical formulations of the shape models used in this work, along with a range of associated measurement models are presented.
The key difference between EOT and point-object tracking is that extended targets may generate multiple measurements per time step. This means that every target has a finite or infinite number of measurement sources [81, p. 141]. The detection probability p_d describes how likely a measurement source causes a measurement. Typically, these measurements are affected by white Gaussian measurement noise. To model this probabilistic relationship, the measurement likelihood $p(\boldsymbol{Z}|\underline{x})$ is used, whereas the number of detections (L) and their spatial distribution across the extent of the target are considered [81, pp. 142–143]. In their overview paper about EOT, Granström et al. classify EOT methods by the three most common ways of modeling the measurement likelihood [81, p. 143-145]:

- set of points on a rigid body (SPRB) [69]
- spatial model [30–41, 65–68, 71–78]
- physics-based modeling [64, 70]

An example of an SPRB model is provided in [69, p. 2375]: A discrete number of reflection centers (point reflectors and plane reflectors) are distributed over the object's contour, in this

[1] Research vessel Solgenia, image from Hannes Homburger, HTWG Konstanz
[2] Northrop B-2, image retrieved from:
`https://de.wikipedia.org/wiki/Northrop_B-2#/media/Datei:B-2_Spirit_original.jpg` on July, 26, 2024
[3] Car topview, image retrieved from:
`https://commons.wikimedia.org/wiki/File:Travel-car-topview.svg` on November, 14, 2024
[4] Konstanz City bus, image retrieved from:
`https://www.busnetz.de/sechs-neue-e-busse-fuer-konstanz/` on April, 24, 2025

case a car [69, p. 2375], and each reflector can generate a measurement independently with probability p_d [81, p. 143]. A significant drawback of SPRBs is the required data association problem of detections with measurement sources or reflection areas [81, p. 143], which must be solved, for example, by using Murty's algorithm [53].
A spatial model uses a Poisson distributed number of detections and the detections themselves are spatially distributed across the target [81, p. 144]. An example of a spatial model is the RM approach proposed in [30] and further extended in [31–38, 41, 65–67]. The basic idea of the RM approach is that the measurements are normally distributed across the object's center, with a covariance being proportional to the object's extension [30, p. 1047]. Further spatial models are e.g. random hypersurface models (RHMs) [71, 72], where the measurements are distributed uniformly across the extent or contour of the target. The advantage of spatial models compared to SPRB models is that there is no assignment problem of detections to measurement sources [81, p. 144]. Physics-based modeling is done when the model does not fit into an SPRB or a spatial model [81, p. 145]. Using a lidar sensor makes it possible to determine the measurement sources by ray tracing [81, p. 145][64, p. 5]. The result is also a set of points on a rigid body. However, these points have a more profound physical background than typical SPRB models. It is easy to derive artificial measurements with ray tracing, but finding a measurement likelihood function in closed form is challenging. The lack of a measurement likelihood function in closed form can make the integration into the Bayesian framework difficult. In this work, mainly lidar measurements that are obtained from physics-based ray tracing are considered, but also measurements that are distributed normally or uniformly over the object's extent.
Own publications on this subject: The ray tracing method for ellipses has been published in the journal paper [PH1] and was also the main measurement model in [PH3] and [PH4]. In [PH5], the measurement models for triangles and rectangles were introduced and all models were expanded with uniform distributed measurements. The 3D models for ellipsoid, cone and cuboid were published in [PH6]. Partial visibility due to objects covering each other was addressed in [PH3]. Reproduction of textual material, illustrations, and tables from those papers for this section is in accordance with the IEEE and ISIF copyright policies.

3.1. Ellipse

The equation of an ellipse centered at the origin and aligned with the coordinate axes is given by its Cartesian representation:

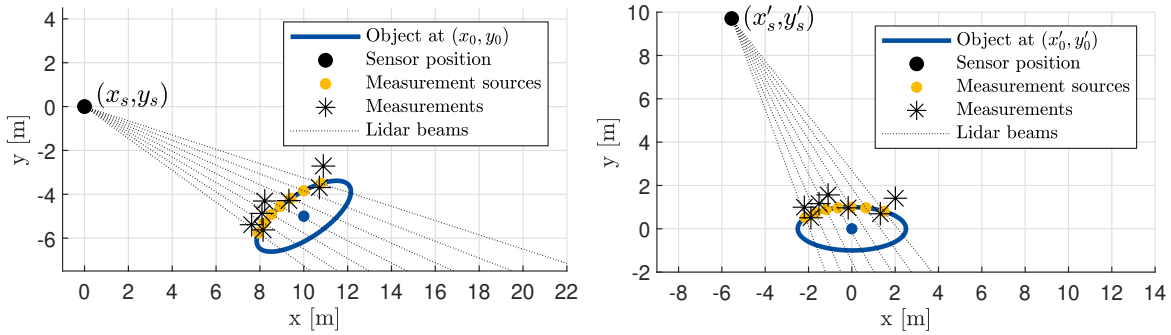
$$\frac{x^2}{l^2} + \frac{y^2}{w^2} = \frac{1}{4}. \tag{3.1}$$

where l and w denote the ellipse's length and width, respectively. In this unrotated case ($\psi = 0°$), the center is located at the origin. In the general case, where the ellipse is centered at (x_0, y_0) and rotated by an orientation angle ψ, its parametric form is given by:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \begin{pmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{pmatrix} \begin{pmatrix} \frac{l}{2}\cos(t) \\ \frac{w}{2}\sin(t) \end{pmatrix}, \tag{3.2}$$

with $t \in [0, 2\pi]$. The following paragraphs present various measurement generation methods for elliptical objects, including contour measurements, as well as measurements uniformly or normally distributed over the object's spatial extent.

Contour measurements Figure 6a illustrates the generation of contour measurements for an elliptical object centered at (x_0, y_0), observed by a lidar sensor located at (x_s, y_s).



(a) Sensor/local coordinates.

(b) Target coordinates.

Figure 6: Measurement generation with a lidar sensor for elliptical targets [PH1, PH4, PH5].

The sensor is assumed to be located at the origin of the coordinate system, as shown in Figure 6a, which therefore represents the sensor or local coordinate system. This distinction becomes important when considering ego motion: in the local coordinate system, the sensor remains at the origin, while the environment appears to move relative to it. In contrast, in a global or stationary frame, the sensor's position changes over time.
Since the elliptical object is not necessarily centered at $(0, 0)$ and may have a nonzero orientation yaw angle ψ, the simple Cartesian form given in (3.1) cannot be used directly in the sensor frame. However, it remains valid in target coordinates, where the coordinate system is centered on the object and aligned with the ellipse's semi-axes, as illustrated in Figure 6b.
Measurement sources are defined as the intersection points of lidar beams with the object's contour. Actual measurements are generated by adding random noise to these intersection points. Because the detection probability p_d is less than one, not every source produces a measurement. The resulting measurements do not follow a simple analytical distribution.
A lidar measurement model based on uniformly distributed measurement sources along the contour of an ellipse was proposed in [68]. Ray tracing for elliptical objects was derived in [64] and later applied to vessel extension estimation in [70]. This section presents a method to generate artificial lidar measurements based on ray tracing, similar to the approach introduced in [64, p. 5].
Determining the measurement sources requires an analytic solution for the intersection points between the lidar beams and the contour of the ellipse. Since the ellipse is most conveniently described in target coordinates, the entire computation is performed in this coordinate system. This requires a transformation of the sensor position into the target frame:

$$\begin{pmatrix} x'_s \\ y'_s \end{pmatrix} = \begin{pmatrix} \cos(\psi) & \sin(\psi) \\ -\sin(\psi) & \cos(\psi) \end{pmatrix} \begin{pmatrix} x_s - x_0 \\ y_s - y_0 \end{pmatrix}. \tag{3.3}$$

The ellipse itself does not need to be transformed, since its center and orientation in target coordinates are given by $x'_0 = 0$, $y'_0 = 0$, and $\psi' = 0$.
With the sensor position expressed in the target coordinate system, the equation of a lidar beam is given by:

$$y = \tan(\alpha_j)x + \left(y's - \tan(\alpha_j)x's\right), \tag{3.4}$$

where α_j denotes the directional angle of the j-th beam. The angular range of the sensor is defined by the start angle α_0 and the end angle $\alpha_{\max}$, while the angular resolution is determined by the step size $\Delta\alpha = \alpha_{j+1} - \alpha_j$. For each angle α_j, the corresponding beam equation is evaluated. To compute the intersection points between the lidar beams and the contour of the object, equation (3.4) is substituted into the ellipse equation (3.1). After some simplifications

and rearrangements, we get:

$$\left(\frac{1}{l^2}+\frac{\tan(\alpha_j)^2}{w^2}\right)x^2+\frac{2\tan(\alpha_j)}{w^2}\left(y'_s-\tan(\alpha_j)x'_s\right)x + \frac{\left(y'_s-\tan(\alpha_j)x'_s\right)^2}{w^2}-\frac{1}{4}=0. \tag{3.5}$$

Equation (3.5) is of the form $ax^2+bx+c=0$ and can be solved using the quadratic formula. The corresponding y'_m value is then obtained by evaluating equation (3.4).
If real-valued solutions $x'_{m1}, x'_{m2} \in \mathbb{R}$ exist, the next step is to determine which intersection point lies on the side of the object facing the sensor. Therefore, the Euclidean distance to the sensor position (x'_s, y'_s) is computed for both solutions. The solution with the smaller distance is selected as the measurement source. This resulting measurement source (x'_m, y'_m) must then be transformed from target coordinates back to the original coordinate system using:

$$\begin{pmatrix} x_m \\ y_m \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \begin{pmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{pmatrix} \begin{pmatrix} x'_m \\ y'_m \end{pmatrix}, \tag{3.6}$$

which is the inverse of (3.3). A similar derivation of the intersection points between lidar beams and the object's contour can be found in [64, p. 5]. To generate measurements, a random value $u \sim U(0,1)$ is drawn for each measurement source. If $u < p_d$, the measurement source generates a measurement by adding white Gaussian noise to the true position:

$$\underline{z}_j = \begin{pmatrix} x_m \\ y_m \end{pmatrix} + \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}, \tag{3.7}$$

where $v_i \sim \mathcal{N}(v; 0, \sigma_x^2)$ are independent zero-mean Gaussian noise terms with variance σ_x^2.

Gaussian distributed measurements Gaussian distributed measurements can be interpreted as noisy observations of the object's centroid, where the measurement uncertainty is proportional to the object's spatial extent [30, p. 1047]. In the 2D case, this results in a Gaussian distribution whose σ-ellipse scales with the size of the object. Consequently, this measurement model is suitable only for elliptical shapes.

Uniformly distributed interior measurements Measurements can also be uniformly distributed across the object's extent [34, Figure 1b]. The corresponding measurement sources can be efficiently generated using rejection sampling, as illustrated in Figure 7.

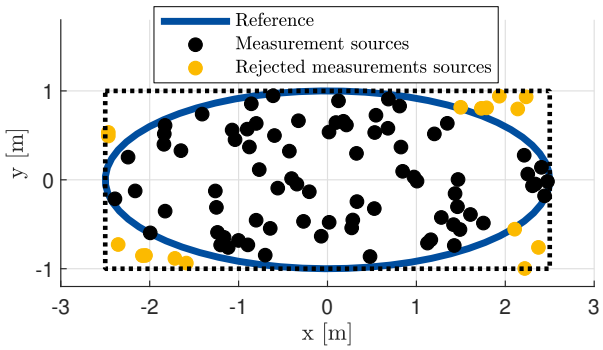


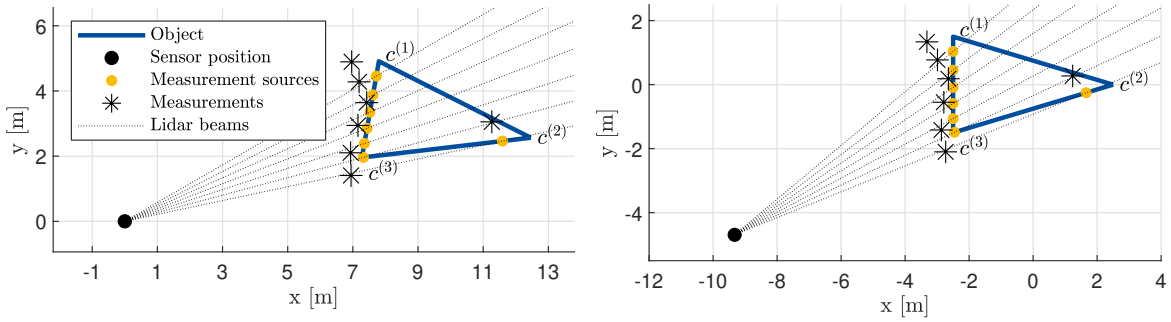
Figure 7: Drawing uniformly distributed samples over an ellipse via rejection sampling [PH5].

First, samples are drawn from s uniform distribution over the bounding rectangle using $x_i \sim \mathbb{U}(-l/2, l/2)$ and $y_i \sim \mathbb{U}(-w/2, w/2)$ with sampling index i. Samples that satisfy the condition $\frac{x_i^2}{l^2}+\frac{y_i^2}{w^2} \leq \frac{1}{4}$ lie within the ellipse and are accepted as measurement sources; all others are re-

jected. Since this method is applicable to arbitrary densities [82], it can also be used for more complex shapes. As measurements are obtained by adding white Gaussian noise to the accepted measurement sources, some resulting measurements may lie outside the object's true extent.

3.2. Triangle

Contour measurements For contour measurements, ray tracing is used to determine the measurement sources by computing the intersections of the lidar beams (see equation 3.4) with the object's contour in target coordinates. Figure 8 illustrates the ray tracing process for a triangular object, shown both in sensor coordinates and in target coordinates.



(a) Sensor coordinates.

(b) Target coordinates.

Figure 8: Ray tracing with a triangular object [PH5].

In target coordinates, the three corners of the triangle are given by:

$$c^{(1)} = \left(\frac{-l}{2}, \frac{w}{2}\right), \tag{3.8}$$

$$c^{(2)} = \left(\frac{l}{2}, 0\right), \tag{3.9}$$

$$c^{(3)} = \left(\frac{-l}{2}, \frac{-w}{2}\right). \tag{3.10}$$

These corner points define the triangle's three edges, each represented by a linear equation of the form:

$$f_y^{(1)}(x) = m^{(1)}x + b^{(1)} \tag{3.11}$$

where, for example, the slope and intercept of the first edge are given by:

$$m^{(1)} = \frac{c_y^{(2)} - c_y^{(1)}}{c_x^{(2)} - c_x^{(1)} + \epsilon}, \tag{3.12}$$

$$b^{(1)} = c_y^{(1)} - m^{(1)}c_x^{(1)}, \tag{3.13}$$

where ϵ is a small value to avoid division by zero. The equations for the other two edges $f_y^{(2)}(x)$, $f_y^{(3)}(x)$ can be calculated simultaneously. The intersection of edge i with the lidar beam equation (in the direction of α) is calculated by equating $f_y^{(i)}(x)$ and (3.4) and solving for x:

$$m^{(i)}x + b^{(i)} = \tan(\alpha)x + y_s - \tan(\alpha)x_s, \tag{3.14}$$

$$x = \frac{y_s \tan(\alpha)y_s - b^{(i)}}{m - \tan(\alpha)} \tag{3.15}$$

The corresponding y-value can be obtained by evaluating either equation (3.11) or equation (3.4) at the computed x-coordinate. This procedure is repeated for all beam directions α and for

each triangle edge to determine the complete set of potential measurement sources in target coordinates.
While the linear equations for the triangle edges describe infinite lines, only the segments between the defined corner points represent the actual edges of the triangle. Therefore, after computing the intersection points between the lidar beams and the edge equations, it is necessary to verify whether these points lie within the bounds of the corresponding edge segments. Intersection points that fall outside the triangular contour must be discarded.

Uniformly distributed interior measurements Uniformly distributed measurements can be generated in target coordinates by first sampling the x-coordinate from a triangular distribution with the following pdf:

$$f_{tri}(x) = \begin{cases} 0 & \text{for } x < a, \\ \frac{2}{b-a} & \text{for } x = a, \\ \frac{2(b-x)}{(b-a)^2} & \text{for } a < x \leq b, \\ 0 & \text{for } b < x \end{cases} \tag{3.16}$$

with $a = \frac{-l}{2}$ and $b = \frac{l}{2}$. For each sampled x-value, the corresponding y-value is drawn from a uniform distribution with bounds that depend on x:

$$y \sim \mathcal{U}\left(\frac{-w}{2} + \frac{w}{l}\left(x + \frac{l}{2}\right), \frac{w}{2} - \frac{w}{l}\left(x + \frac{l}{2}\right)\right). \tag{3.17}$$

3.3. Rectangle

Contour measurements Similar to the triangle, the corner points of the rectangle in target coordinates are defined as:

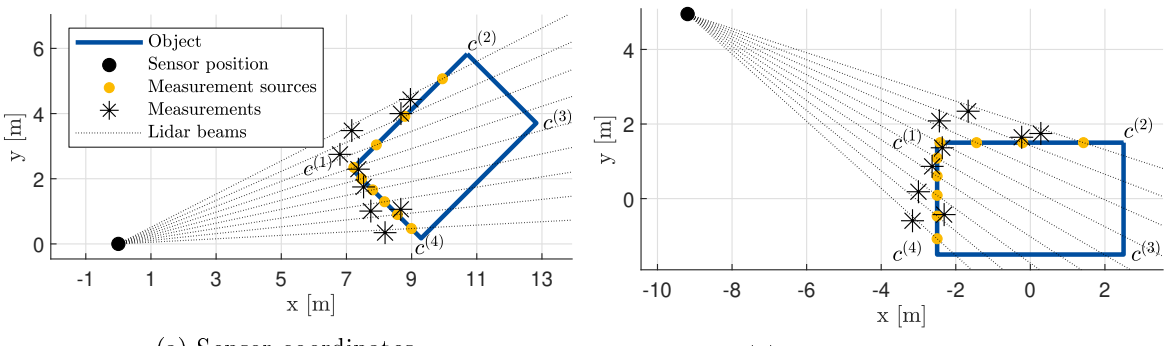
$$c^{(1)} = \left(\frac{-l}{2}, \frac{w}{2}\right), \tag{3.18}$$

$$c^{(2)} = \left(\frac{l}{2}, \frac{w}{2}\right), \tag{3.19}$$

$$c^{(3)} = \left(\frac{l}{2}, \frac{-w}{2}\right), \tag{3.20}$$

$$c^{(4)} = \left(\frac{-l}{2}, \frac{-w}{2}\right). \tag{3.21}$$

The four edges can be described analogously to those of the triangle using linear equations as in (3.11) and (3.13). The intersection points between the lidar beams and the rectangle edges are then computed by solving the corresponding equations as outlined in (3.15). Figure 9 illustrates the ray tracing procedure for the rectangle in both sensor and target coordinate systems.



(a) Sensor coordinates.

(b) Target coordinates.

Figure 9: Ray tracing with a rectangular object [PH5].

Uniformly distributed interior measurements The rectangle is the most straightforward shape for generating uniformly distributed measurements, as the x- and y-coordinates can be sampled independently in target coordinates using

$$x \sim \mathcal{U}(-l/2, l/2), \tag{3.22}$$
$$y \sim \mathcal{U}(-w/2, w/2). \tag{3.23}$$

3.4. Ellipsoid

The ellipsoid is the first 3D shape model considered in this work. As in the previous sections, both a local (sensor) coordinate system and a target coordinate system are used. For all 3D models, both coordinate systems follow the x-East, y-North, z-Up (ENU). Since the target coordinate system is body-fixed, it is sometimes referred to as the BODY frame. A detailed discussion of 3D coordinate systems and their transformations is provided by Fossen in [83].

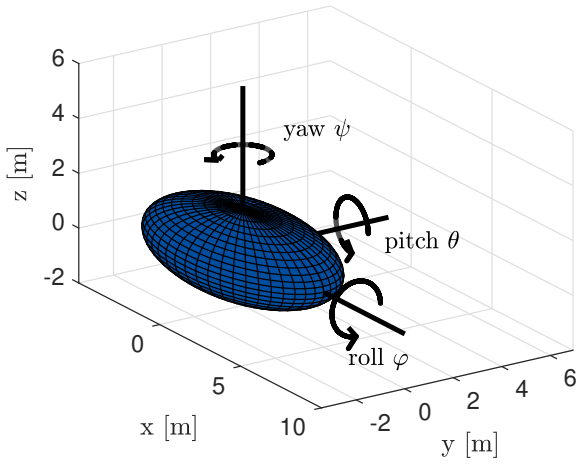
The surface of a triaxial ellipsoid is defined by the implicit equation:

$$\frac{x^2}{l^2} + \frac{y^2}{w^2} + \frac{z^2}{h^2} = \frac{1}{4}, \tag{3.24}$$

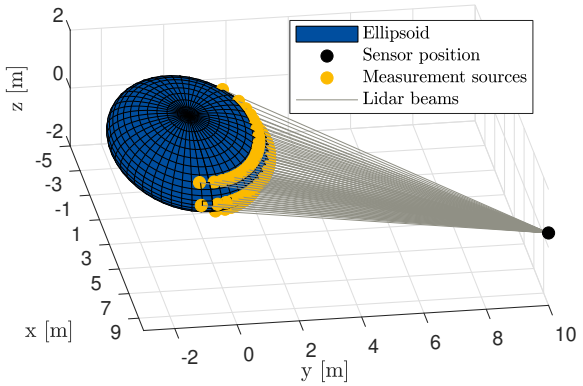
where l, w, and h denote the length, width, and height, respectively. In spherical coordinates, its surface can be described using:

$$x_1 = \frac{l}{2}\sin(t_2)\cos(t_1), \tag{3.25}$$
$$y_1 = \frac{w}{2}\sin(t_2)\sin(t_2), \tag{3.26}$$
$$z_1 = \frac{h}{2}\cos(t_2), \tag{3.27}$$

with azimuth angle t_1 and inclination t_2. In general, however, the ellipsoid is located at (x_0, y_0, z_0) and rotated by pitch θ, yaw ψ and roll φ (see Figure 10a). This transformation from ENU to



(a) Pitch θ, yaw ψ and roll φ.



(b) Ray tracing.

Figure 10: 3D Ellipsoid [PH6].

BODY coordinates is expressed by [83, p. 24, adjusted for zUp]:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} + \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\varphi) & -\sin(\varphi) \\ 0 & \sin(\varphi) & \cos(\varphi) \end{pmatrix}}_{R_\varphi \text{ for roll}} \cdot \underbrace{\begin{pmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{pmatrix}}_{R_\theta \text{ for pitch}} \underbrace{\begin{pmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{R_\psi \text{ for yaw}} \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix}. \tag{3.28}$$

Contour measurements Since the ellipsoid equation in (3.24) is significantly simpler than its representation in ENU coordinates (see equation 3.28), the subsequent intersection calculations for ray tracing are performed in the *BODY* coordinate system. Figure 10b illustrates the generation of measurement sources from an ellipsoid using a lidar sensor and ray tracing.
The first step in computing the intersection points is to transform the sensor position (x_s, y_s, z_s) into BODY coordinates using the inverse rotation matrices:

$$\begin{pmatrix} x'_s \\ y'_s \\ z'_s \end{pmatrix} = R_\varphi^{-1} R_\theta^{-1} R_\psi^{-1} \begin{pmatrix} x_s \\ y_s \\ z_s \end{pmatrix}. \tag{3.29}$$

Then, a single lidar beam in azimuth direction θ_s and inclination ϕ_s can be described using:

$$x = x'_s + r\cos(\phi_s)\cos(\theta_s), \tag{3.30}$$
$$y = y'_s + r\cos(\phi_s)\sin(\theta_s), \tag{3.31}$$
$$z = z'_s + r\sin(\phi_s). \tag{3.32}$$

Inserting equations (3.30)–(3.32) into the ellipsoid equation (3.24) yields:

$$\left(\frac{(x'_s + r\cos(\phi_s)\cos(\theta_s))^2}{l^2}\right) + \left(\frac{(y'_s + r\cos(\phi_s)\sin(\theta_s))^2}{w^2}\right) + \left(\frac{(z'_s + r\sin(\phi_s))^2}{h^2}\right) - \frac{1}{4} = 0, \tag{3.33}$$

which can be rearranged into a quadratic equation in r:

$$\begin{aligned} &\left(\frac{\cos(\phi_s)^2\cos(\theta_s)^2}{l^2} + \frac{\cos(\phi_s)^2\sin(\theta_s)^2}{w^2} + \frac{\sin(\phi_s)^2}{h^2}\right) r^2 \\ &+ \left(\frac{2x'_s\cos(\phi_s)\cos(\theta_s)}{l^2} + \frac{2y'_s\cos(\phi_s)\sin(\theta_s)}{w^2} + \frac{2z'_s\sin(\phi_s)}{h^2}\right) r \\ &+ \frac{x'^2_s}{l^2} + \frac{y'^2_s}{w^2} + \frac{z'^2_s}{h^2} - \frac{1}{4} = 0 \end{aligned} \tag{3.34}$$

This quadratic equation is solved for r. If no real solution $r \in \mathbb{R}$ exists, the lidar beam does not intersect the ellipsoid. If two solutions exist, the smaller value of r corresponds to the first intersection point and is selected as the measurement source. Inserting this value of r into equations (3.30)–(3.32) yields the Cartesian coordinates of the measurement source in BODY coordinates.
Finally, the result is transformed back into ENU coordinates using:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} + R_\varphi R_\theta R_\psi \begin{pmatrix} x \\ y \\ z \end{pmatrix}. \tag{3.35}$$

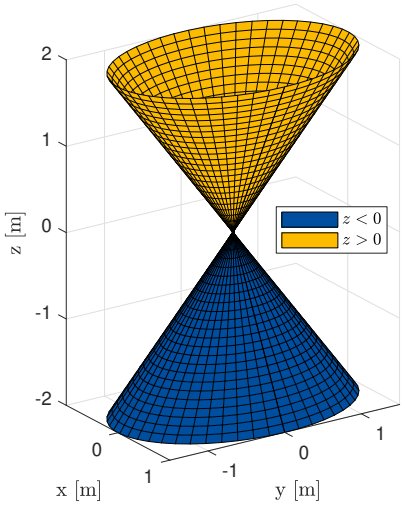
This intersection calculation is performed for each lidar beam.

3.5. Elliptical cone

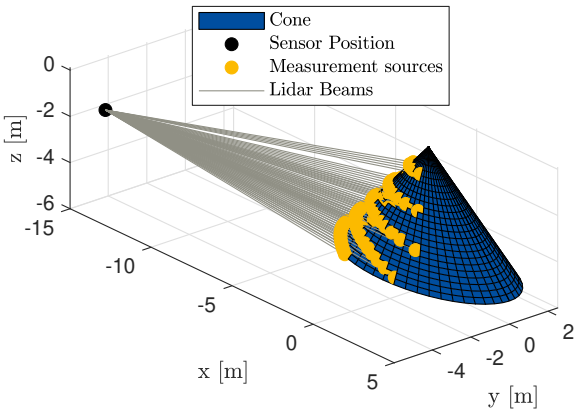
The Cartesian equation of an elliptical cone is given by:

$$\frac{x^2}{\left(\frac{l}{2}\right)^2} + \frac{y^2}{\left(\frac{w}{2}\right)^2} - \frac{z^2}{h^2} = 0. \tag{3.36}$$

where l, w, and h denote the base length, width, and height of the cone, respectively. The apex of the cone is located at the origin, and the lateral surface extends infinitely in the positive and negative z-direction. However, for the purpose of this work, only the finite portion of the cone with $z \in [-h, 0]$ is considered, as illustrated by the blue-highlighted region in Figure 11a.



(a) Symmetric representation.



(b) Ray tracing in target coordinates.

Figure 11: Elliptical cone [PH6].

The cone can also be expressed in parametric form [79, 84]:

$$c(u, s) = \begin{bmatrix} u \cdot l/2 \cdot \cos(s) \\ u \cdot w/2 \cdot \sin(s) \\ -u \cdot h \end{bmatrix}, \tag{3.37}$$

with $u \in [0, 1]$ and $s \in [0, 2\pi]$. This parameterization describes the lateral surface of the cone and ensures that the apex is located at the origin, consistent with its Cartesian representation given in equation (3.36).

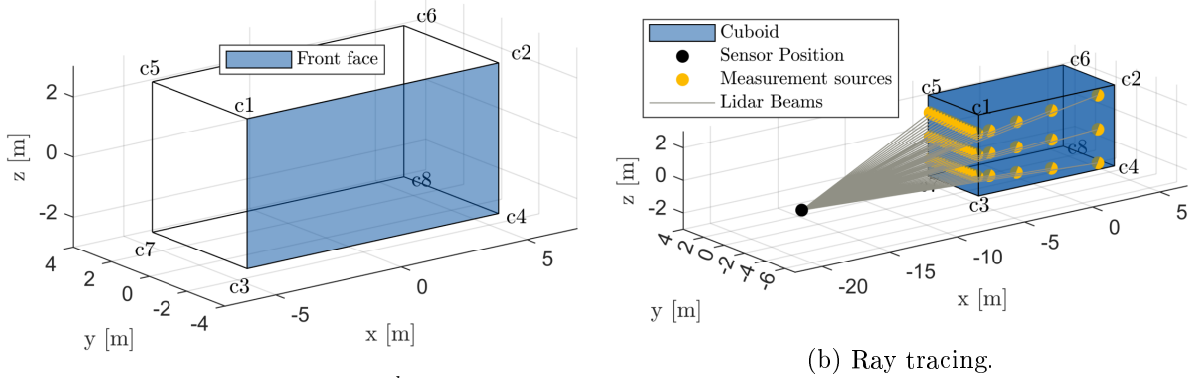
Contour measurements Ray tracing for the elliptical cone follows a methodology similar to that for the ellipsoid. Intersections of lidar beams with the cone's surface are determined by substituting (3.30)-(3.32) into the cone equation (3.36). This yields an equation similar to (3.34),

$$\begin{aligned} &\left(\frac{\cos(\phi_s)^2 \cos(\theta_s)^2}{\left(\frac{l}{2}\right)^2} + \frac{\cos(\phi_s)^2 \sin(\theta_s)^2}{\left(\frac{w}{2}\right)^2} - \frac{\sin(\phi)^2}{h^2} \right) r^2 \\ &+ \left(\frac{2x'_s \cos(\phi_s) \cos(\theta_s)}{\left(\frac{l}{2}\right)^2} + \frac{2y'_s \cos(\phi_s) \sin(\theta_s)}{\left(\frac{w}{2}\right)^2} - \frac{2z'_s \sin(\phi_s)}{h^2} \right) r, \\ &+ \frac{{x'_s}^2}{\left(\frac{l}{2}\right)^2} + \frac{{y'_s}^2}{\left(\frac{w}{2}\right)^2} - \frac{{z'_s}^2}{h^2} = 0, \end{aligned} \tag{3.38}$$

which is then solved for r. Subsequently, the corresponding Cartesian coordinates are calculated using (3.30)-(3.32) and only solutions with $z \in [-h, 0]$ are considered as valid measurement sources, as the cone surface is originally unbounded in z. These valid measurement sources must be transformed into ENU coordinates using (3.35). Figure 11b illustrates an example of ray tracing for a 3D elliptical cone. While contour measurements from the elliptical base area could potentially be generated, they have not been incorporated into this work, as they are not visible in the examined scenarios due to observation angle limitations.

3.6. Rectangular cuboid

Contour measurements The cuboid has a total of six faces, each capable of intersecting with lidar beams to generate measurement sources. One of those faces is highlighted in Figure 12a and Figure 12b illustrates the ray tracing procedure in BODY coordinates and in the corresponding target coordinate system.



(a) Cuboid: The face with $y = \pm\frac{l}{2}$ is highlighted.

(b) Ray tracing.

Figure 12: Rectangular cuboid [PH6].

First, the sensor's position is transformed into this coordinate system using (3.29). In the BODY coordinate system, the cuboid's six plane equations are defined as follows: $x = \pm\frac{l}{2}$, $y = \pm\frac{w}{2}$ and $z = \pm\frac{h}{2}$. To find the intersections with a lidar beam, (3.31) is substituted into the plane equations and solved for the radius r, leading to:

$$r_{1,2} = \frac{\pm\frac{l}{2} - x'_s}{\cos(\phi_s)\cos(\theta_s)}, \tag{3.39}$$

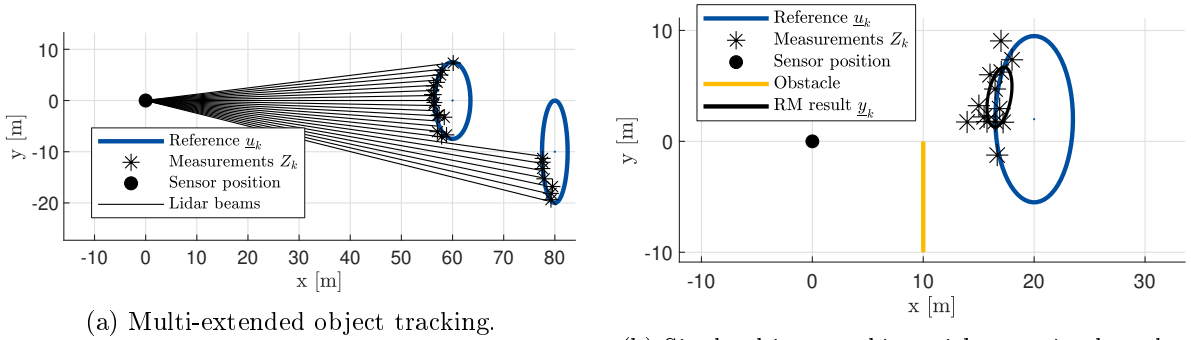
$$r_{3,4} = \frac{\pm\frac{w}{2} - y'_s}{\cos(\phi_s)\sin(\theta_s)}, \tag{3.40}$$

$$r_{5,6} = \frac{\pm\frac{h}{2} - z'_s}{\sin(\theta_s)}. \tag{3.41}$$

The Cartesian coordinates of the intersection points are then computed using equations (3.30)-(3.32). However, since the planes defining the cuboid's faces are mathematically unbounded, only solutions that satisfy the conditions $-\frac{l}{2} \leq x \leq \frac{l}{2}$, $-\frac{w}{2} \leq y \leq \frac{w}{2}$, and $-\frac{h}{2} \leq z \leq \frac{h}{2}$ are considered as valid intersections of the lidar beams with the cuboid's contour.
Among all valid intersection points, only those corresponding to the smallest positive radius r_i for each beam are selected as measurement sources, since lidar measurements occur only on the face of the object that directly faces the sensor. Finally, the resulting coordinates are transformed from BODY to ENU coordinates using equation (3.35).

3.7. Partially visible objects and occlusion

In both, single-object and multi-object tracking, objects may not be fully observable. This can occur due to occlusion by other dynamic objects, as illustrated in Figure 13a, or due to static barriers, as shown in Figure 13b. The topic of occlusion and partially visible objects has been investigated extensively in [PH3].
In physics-based modeling using ray tracing, such occlusions can be naturally accounted for: For each lidar beam, all potential intersection points with the contours of all objects or static obstacles are computed, and only the closest intersection point to the sensor is selected as the measurement source.
In the special case of a static wall, as shown in Figure 13b, a more efficient alternative to explicit intersection calculation is to filter measurement sources based on angular constraints. Specifically, only those with angles $0 < \alpha < \text{atan2}(-10, 10)$, where α is the angular position of the source in polar coordinates, are accepted.



(a) Multi-extended object tracking.

(b) Single object tracking with a static obstacle.

Figure 13: Occlusions [PH3].

If a spatial model is used for the measurement likelihood, occluded regions could alternatively be identified using a heuristic that sets the measurement source density to zero in those areas. However, implementing such a heuristic still requires some form of physical occlusion modeling, such as ray tracing.

4. The virtual measurement model framework

The shape and measurement models presented in the previous chapter are ideal for precisely modeling the generation of measurements in a wide variety of scenarios based on physical properties. Moreover, since these models can be combined flexibly, they enable accurate representations even in complex scenarios that require a combination of contour measurements and uniformly or normally distributed measurements.
Such combined measurement scenarios typically arise in real-world lidar data. A practical example is tracking boats using 3D lidar data reduced to a 2D bird's-eye view. Here, certain lidar layers predominantly generate contour measurements of the boat's hull, whereas other layers produce measurements from objects located on deck, roofs, or cabin structures. Due to their 2D-position being inside the vessel's contour, these inner structures appear as if their measurements originate from the boat's surface.
Despite their flexibility, these realistic measurement models pose significant challenges in EOT Specifically, Bayesian filters, which form the basis of most tracking approaches, fundamentally depend on accurately evaluating the measurement likelihood function $p(\underline{z}_k|\underline{x}_k)$ (see Section 2.2). While closed-form expressions for the measurement likelihood are available for normally or uniformly distributed data, they become infeasible or computationally demanding for contour and combined measurement models due to their inherent nonlinearity and asymmetry.
To address uniform or normally distributed measurements efficiently, the random matrix approach summarized in section 2.5 is one of the most popular methods for EOT. It provides a computationally efficient, robust, and precise solution to the Bayesian filtering problem, yet it struggles with measurement scenarios that include significant contour data or asymmetrical shapes.
Motivated by these limitations, the virtual measurement model (VMM) framework proposed in this thesis aims to combine the robustness and computational efficiency of the random matrix approach with the flexibility of complex physics-based measurement models. In particular, the VMM framework is based on the random matrix method, inheriting its desirable properties such as robustness to measurement noise and computational efficiency, largely independent of the number of measurements. However, since directly applying the random matrix approach leads to mismatched modeling when contour or combined measurements are involved, additional correction steps are necessary. The proposed solution involves generating artificial, so-called *virtual*, measurements and subsequently comparing their statistical moments with the filtered statistical moments of the actual measurements obtained from the random matrix algorithm. The estimate is then iteratively adapted until the statistical moments of virtual and filtered actual measurements closely match. Alternatively, a regression model can be trained on virtual measurements to directly adjust the random matrix filter's output.

Own publications on this subject: The VMM framework was first introduced in the journal "IEEE Transactions on Signal Processing" [PH1] including the lidar sensor ray tracing method for elliptical-shaped objects from the previous section and the VMM input adaptation with proof of stability and convergence. This article has been read more than 2000 times and has been cited 32 times as of August 2025 (according to researchgate.net).
In the conference publications [PH3–PH6] the VMM framework was extended to the state of this dissertation: In [PH3], multi extended object tracking within the VMM framework and concealment of objects was considered using the same input adaptation from the original paper. As an alternative to this adaptation, a Gaussian processes regression model (GPRM) was presented in [PH4]. Finally, [PH5] and [PH6] focus on shape classification with virtual measurement models which is the topic of section 5. Reproduction of textual material, illustrations, and tables from those papers for this section is in accordance with the IEEE and ISIF copyright policies.

4.1. Reference model concept

The main motivation behind the reference model concept is to efficiently estimate the position and extent of targets without explicitly requiring a closed-form expression for the measurement likelihood. Thus, this approach conveniently accommodates all the shape-specific measurement models introduced in Section 3. The fundamental idea is to interpret the entire processing chain from the reference model up to the estimation step performed by the random matrix approach as a plant model. Figure 14 illustrates this plant representation in its upper part.

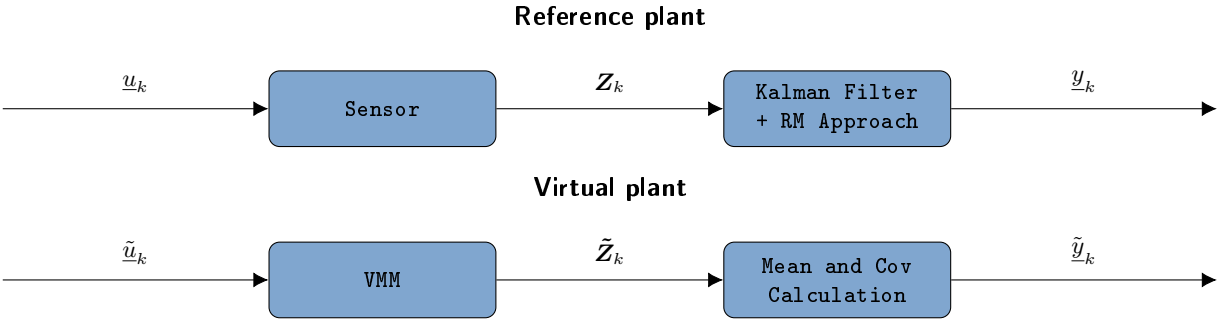


Figure 14: Reference model concept [PH1].

This plant will henceforth be referred to as the reference model, respectively reference plant.

Reference plant Its input vector $\underline{u}_k$ consists of the unknown position and extent parameters of the actual object at time step k:

$$\underline{u}_k = \begin{cases} (x_k, y_k, l_k, w_k)^T & \text{for 2D tracking,} \\ (x_k, y_k, z_k, l_k, w_k, h_k)^T & \text{for 3D tracking.} \end{cases} \tag{4.1}$$

For 2D tracking scenarios, the input vector $\underline{u}_k$ includes the position (x_k, y_k), the object's length l_k, and width w_k at time step k. In 3D scenarios, an additional vertical position z_k and the height h_k are included as further extent parameters. Based on $\underline{u}_k$, the sensor generates a set of measurements $\boldsymbol{Z}_k$. Afterwards, the Kalman filter with RM extension estimation provides the filtered kinematic state $\underline{x}_{k|k}$ and the filtered covariance matrix $\mathbb{E}(X_{k|k})$ of those measurements.
In the earliest works on VMMs [PH3–PH5], the larger and smaller eigenvalues of $\mathbb{E}(X_{k|k})$ are denoted as $e_{1_{k|k}}$ and $e_{2_{k|k}}$, respectively. It is assumed that the square roots of these eigenvalues are linearly related to the object's semi-axes, leading to the following proportional relationships:

$$l_{k|k} \propto 2\sqrt{e_{1_{k|k}}}, \tag{4.2}$$

$$w_{k|k} \propto 2\sqrt{e_{2_{k|k}}}. \tag{4.3}$$

The factor 2 is not decisive for the proportionality and could theoretically be omitted. However, the relationship used in the random matrix approach, namely $l_{k|k} = 2\sqrt{e_{1_{k|k}}}$ and $w_{k|k} = 2\sqrt{e_{2_{k|k}}}$, also plays a role in the initialization of the adaptation process discussed in the next section. For the sake of consistency, it is therefore retained in this formulation. In practical applications, however, only the proportionality is relevant.
In 2D scenarios, the assumption $l > w$ is generally plausible. In contrast, a corresponding assumption such as $l > w > h$ is difficult to justify in 3D due to the added spatial dimension. Real-world objects may also exhibit proportions like $l > h > w$ with similar likelihood, making fixed ordering assumptions problematic.
As an alternative that remains independent of the specific ordering of extent parameters, an eigenvector-based matching approach is proposed. This method is outlined in Algorithm 1. and assumes small roll and pitch angles. For larger angles, the eigenvector with the largest absolute z-component may not correspond to the height, which may lead to incorrect assignments.

Algorithm 1 Eigenvector-based matching

Compute the eigenvalues and eigenvectors of the extent matrix $X_{k|k}^{(Z)}$.
Identify the eigenvector with the largest z-component and assign its corresponding eigenvalue to the height.
Determine the yaw angles of the remaining two eigenvectors.
Compute the yaw angle from the kinematic state.
Assign the eigenvalue whose eigenvector's yaw angle is closest to the kinematic yaw to the length.
Assign the remaining eigenvalue to the width.

The eigenvalue assigned to the length is now denoted by $e_{l_{k|k}}$, that for the width by $e_{w_{k|k}}$, and that for the height by $e_{h_{k|k}}$. Based on this assignment, the output vector $\underline{y}_k$ of the reference plant can now be defined as follows:

$$\underline{y}_k = \begin{cases} \left(z_{x_{k|k}}, z_{y_{k|k}}, 2\sqrt{e_{l_{k|k}}}, 2\sqrt{e_{w_{k|k}}}\right)^T & \text{for 2D tracking,} \\ \left(z_{x_{k|k}}, z_{y_{k|k}}, z_{z_{k|k}}, 2\sqrt{e_{l_{k|k}}}, 2\sqrt{e_{w_{k|k}}}, 2\sqrt{e_{h_{k|k}}}\right)^T & \text{for 3D tracking.} \end{cases} \tag{4.4}$$

Here, $\left(z_{x_{k|k}}, z_{y_{k|k}}\right)$, respectively $\left(z_{x_{k|k}}, z_{y_{k|k}}, z_{z_{k|k}}\right)$ represent the filtered center of gravity in 2D and 3D. According to Singer's ODM, the center of gravity corresponds to the first two or three components of the kinematic state, depending on the tracking dimensionality.

Other kinematic quantities such as velocity or acceleration are assumed to be available from the Kalman filter's kinematic state, as they are largely unaffected by the choice of measurement model.

Furthermore, it is assumed that the object is aligned with its direction of motion. Therefore, the 2D orientation angle $\psi_{k|k}$ of the object's extent can be derived directly from the velocity components of the estimated kinematic state:

$$\psi_{k|k} = \arctan\left(\frac{v_{y_{k|k}}}{v_{x_{k|k}}}\right). \tag{4.5}$$

In the 3D case, the pitch angle $\theta_{k|k}$ can be estimated similar using:

$$\theta_{k|k} = \arctan\left(\frac{v_{z_{k|k}}}{\sqrt{v_{x_{k|k}}^2 + v_{y_{k|k}}^2}}\right), \tag{4.6}$$

whereas the roll angle φ remains undefined in this framework and is treated as arbitrary.
This assumption holds for simple motion models, such as Singer's ODM. However, for more complex scenarios, e.g. when the target exhibits drift or rapid orientation changes, dedicated methods for estimating the roll angle $\varphi_{k|k}$ may be required (see, e.g., [65]).

Difficulties also arise when the target comes to a stop and becomes static. In such cases, even slight inaccuracies in ego motion compensation can lead to sign changes in the estimated velocity components, which may cause unintentional rotations of the estimated orientation.

As an alternative, the orientation angles can be derived directly from the covariance matrix estimated by the random matrix approach [85, p. 44]:

$$\psi_{k|k} = \arctan\left(\frac{d - (\Phi_{1,1} - \Phi_{2,2})}{2\Phi_{2,1}}\right), \tag{4.7}$$

whereas Φ is the expected value for the covariance matrix $\mathbb{E}\left(X_{k|k}\right)$, $d = \sqrt{(\Phi_{1,1} - \Phi_{2,2})^2 + 4\Phi_{2,1}^2}$

and $\Phi_{2,1} \neq 0$. Equation (4.7) is derived by first identifying the eigenvector of the extent matrix Φ that corresponds to its largest eigenvalue. This eigenvector is then used analogously to the velocity vector in Equation (4.5) to estimate the orientation.
In 3D tracking scenarios, both pitch θ and yaw ψ can be extracted from this dominant eigenvector. However, computing the eigenvalues of Φ requires solving a cubic characteristic equation, which complicates the derivation of a closed-form analytical solution.
By additionally evaluating the remaining eigenvectors, it is also possible to estimate the roll angle φ, which is otherwise not accessible via the kinematic state alone.

Virtual plant Parallel to the reference plan, a virtual plant is introduced, designed to emulate its behavior. In this plant, the sensor is replaced by a virtual measurement model (VMM), and the filtering process is substituted by a direct computation of the sample mean and covariance. The input vector $\underline{\tilde{u}}$ follows the structure from Equation (4.41) and represents a suggested estimate.

$$\underline{\tilde{u}}_k = \begin{cases} (\tilde{x}_k, \tilde{y}_k, \tilde{l}_k, \tilde{w}_k)^T & \text{for 2D tracking,} \\ (\tilde{x}_k, \tilde{y}_k, \tilde{z}_k, \tilde{l}_k, \tilde{w}_k, \tilde{h}_k)^T & \text{for 3D tracking.} \end{cases} \tag{4.8}$$

The output vector $\underline{\tilde{y}}_k$ of the virtual plant is defined in the same form as in Equation (4.4):

$$\underline{\tilde{y}}_k = \begin{cases} \left(\tilde{z}_{x_k}, \tilde{z}_{y_k}, 2\sqrt{\tilde{e}_{l_k}}, 2\sqrt{\tilde{e}_{w_k}}\right)^T & \text{for 2D tracking,} \\ \left(\tilde{z}_{x_k}, \tilde{z}_{y_k}, \tilde{z}_{z_k}, 2\sqrt{\tilde{e}_{l_k}}, 2\sqrt{\tilde{e}_{w_k}}, 2\sqrt{\tilde{e}_{h_k}}\right)^T & \text{for 3D tracking.} \end{cases} \tag{4.9}$$

Since no filtering is applied within the virtual plant, the resolution of the VMM must be sufficiently high to ensure an accurate estimation of all components of $\underline{\tilde{y}}_k$. Notably, increasing the sensor resolution does not influence the estimated center of gravity or the measurement variance. The VMM can be interpreted as a function that generates artificial measurements. To formally describe the input–output relationship of the virtual plant, the mapping

$$\underline{\tilde{y}}_k = \tilde{f}(\underline{\tilde{u}}_k) \tag{4.10}$$

is defined. However, due to the complexity of the VMM, the function $\tilde{f}$ is not available in closed form.

Optimization problem The objective of the reference model concept is to find an input $\underline{\tilde{u}}_k$ for the VMM plant such that the output $\underline{\tilde{y}}_k = \tilde{f}(\underline{\tilde{u}}_k)$ matches the output $\underline{y}_k$ of the reference plant. If both outputs are equal and the system is fully observable, then the corresponding inputs must also be equal. In this case, $\underline{\tilde{u}}_k$ becomes an estimate of the system state, respectively the input $\underline{u}_k$.
This condition can be formulated as the following optimization problem:

$$\underline{\tilde{u}}_k^* = \arg\min_{\underline{\tilde{u}}_k} \left(\underline{y}_k - \tilde{f}(\underline{\tilde{u}}_k)\right) \left(\underline{y}_k - \tilde{f}(\underline{\tilde{u}}_k)\right)^\top, \tag{4.11}$$

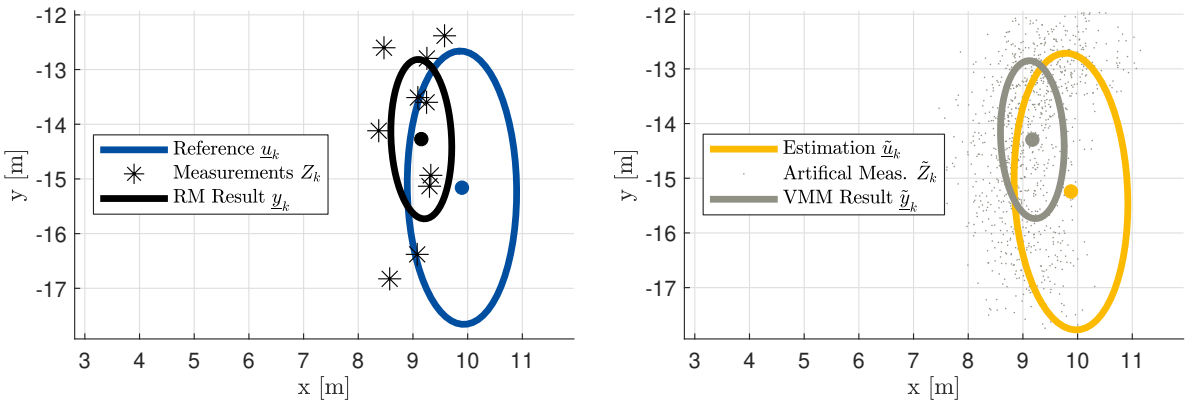
for which the following error vector and corresponding error norm are defined:

$$\underline{\epsilon}_k = \underline{y}_k - \underline{\tilde{y}}_k, \tag{4.12}$$

$$\epsilon_k = \underline{\epsilon}_k \, \underline{\epsilon}_k^\top. \tag{4.13}$$

This work proposes an adaptation algorithm for the input $\underline{\tilde{u}}_k$, which is based on an integral adaptation law with a variable gain K_i. The algorithm is presented in detail in Section 4.2.
As an alternative to the adaptation, the Nelder–Mead algorithm [86] can be employed. It is a numerical optimization method that uses a simplex-based approach to iteratively search for an input that minimizes the error.

Practical example To illustrate the reference model concept, a 2D tracking example is considered. A noisy lidar sensor, positioned at the origin of the coordinate system, is used to track an elliptically shaped object using the random matrix approach. Figure 15a corresponds to the processing chain within the reference plant, from the true object parameters to the filtered result obtained via the random matrix method. The output of the RM approach is visualized as a σ-ellipse, where the semi-axes are defined by $l_{k|k} = 2\sqrt{e1_{k|k}}$ and $w_{k|k} = 2\sqrt{e2_{k|k}}$.



(a) Reference plant.

(b) Virtual plant.

Figure 15: Introductory example at time step $k = 50$ [PH1].

The input and output of the virtual plant are illustrated in Figure 15b. It is assumed that an adaptation algorithm provides an estimate $\tilde{\underline{u}}_k$, which is then used as input to the VMM in order to generate a large number of artificial measurements.
The mean and covariance of these artificial measurements are visualized using a σ-ellipse and is denoted in Figure 15b as VMM Result $\tilde{\underline{y}}_k$. To assess the quality of the estimate $\tilde{\underline{u}}_k$, a comparison with the reference $\underline{u}_k$ (Figure 15a) would be informative. However, in real-world applications, the true object parameters $\underline{u}_k$ are not available.
Instead, the comparison can be made between the output of the random matrix approach and the empirical mean and covariance of the artificial measurements. These two results are visualized as black and grey ellipses in Figure 15. In this example, the alignment between both ellipses is very good, indicating that the artificial measurements closely match the filtered output. Consequently, the error ϵ_k defined in Equation (4.13) is expected to be small, and $\tilde{\underline{u}}_k$ can be considered a good estimate of the object's position and extent.

Measurement noise When using the random matrix approach as described in Section 2.5 and based on [30], measurement noise must be considered in the VMM. This can be achieved either by adding noise directly to the generated measurements, or by incorporating the measurement noise covariance matrix into the empirical covariance of the artificial measurements.
In contrast, advanced approaches such as those proposed in [31, 32, 34, 35] use an inverse Wishart distribution in which the expected value exclusively represents the object's extent. In such cases, the VMM must be designed without adding measurement noise.

4.2. Adaptation algorithm

The goal of the adaptation algorithm is to iteratively determine an input $\underline{\tilde{u}}_{k,i}$ that minimizes the corresponding error $\epsilon_{k,i}$, as defined in Equation (4.13), where i denotes the iteration index. Figure 16 illustrates the structure of the proposed approach.

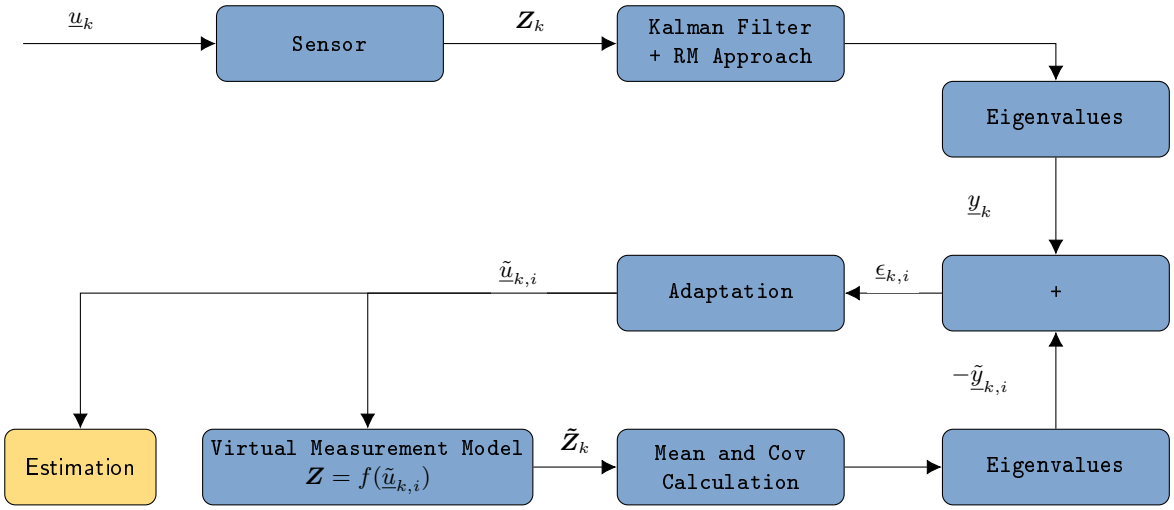


Figure 16: Reference model concept with RM approach and an adaptation algorithm [PH1].

The first part of the flow chart, from the real-world parameters $\underline{u}_k$ to the output of the random matrix approach $\underline{y}_k$, is already known from Figure 14, including the virtual plant depicted in the lower section. For each time step k, the adaptation algorithm is executed over I iterations. In each iteration $i \in \{1, 2, \ldots, I\}$, the input parameters $\underline{\tilde{u}}_{k,i}$ are updated using the following adaptation law:

$$\underline{\tilde{u}}_{k,i+1} = \underline{\tilde{u}}_{k,i} + K_i \, \underline{\epsilon}_{k,i}, \tag{4.14}$$

where $\underline{\epsilon}_{k,i} = \underline{y}_k - \underline{\tilde{y}}_{k,i}$ is the error vector as defined in Equation (4.12).

Closed-loop stability The lower part of Figure 16, consisting the VMM, the computation of mean and covariance, the eigenvalue calculation, and the adaptation mechanism, is a closed feedback loop.
The most critical requirement for the adaptation law is that this feedback loop is asymptotically stable. To analyze stability, all components except the adaptation algorithm are grouped into a single plant, denoted by A. This abstraction leads to the simplified control structure shown in Figure 17. For better readability, time and iteration indices are only written for the vectors and neglected for their elements.

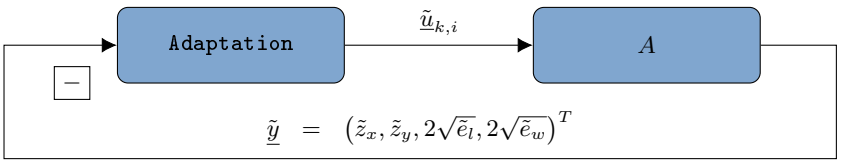


Figure 17: Adaptation in closed-loop [PH1].

A central challenge in this approach is that the measurement likelihood, and therefore consequently the plant A, cannot be expressed in closed form. Nevertheless, upper and lower bounds can be assumed in order to analyze stability.
For the center of gravity of the measurements, it is assumed that it may shift from the geometric center of the object by at most half the largest extension in the direction of the sensor. This

shift can be modeled as:

$$\tilde{z}_x = \tilde{x} - \operatorname{sgn}(\tilde{x}) \max(\tilde{l}, \tilde{w}) a, \tag{4.15}$$

$$\tilde{z}_y = \tilde{y} - \operatorname{sgn}(\tilde{y}) \max(\tilde{l}, \tilde{w}) b, \tag{4.16}$$

with $a \in [0,\, 0.5]$ and $b \in [0,\, 0.5]$. Time and iteration indices are neglected for better readability. The values of a and b are considered constant within a given time step k, and their variation is assumed to be negligible during the adaptation iterations.

For the eigenvalues, it is assumed that the square root of the larger eigenvalue corresponds linearly to half the object length, and the square root of the smaller eigenvalue corresponds linearly to half the object width. Since both expressions must be non-negative and cannot exceed the actual object dimensions, the following relationships are assumed:

$$2\sqrt{\tilde{e}_l} = c\,\tilde{l}, \tag{4.17}$$

$$2\sqrt{\tilde{e}_w} = d\,\tilde{w}, \tag{4.18}$$

with $c \in [0,\, 1]$ and $d \in [0,\, 1]$.

For the first quadrant ($\tilde{x}, \tilde{y} > 0$), and under the assumption that the length is by definition larger than the width, Equations (4.15)–(4.18) can be rearranged into a system of linear equations of the form $\underline{\tilde{y}}_{k,i} = A\underline{\tilde{u}}_{k,i}$:

$$\begin{pmatrix} \tilde{z}_x \\ \tilde{z}_y \\ 2\sqrt{\tilde{e}_l} \\ 2\sqrt{\tilde{e}_w} \end{pmatrix} = \begin{pmatrix} 1 & 0 & -a & 0 \\ 0 & 1 & -b & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \end{pmatrix} \begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{l} \\ \tilde{w} \end{pmatrix}. \tag{4.19}$$

Using the adaptation law (4.14) leads to a closed-loop dynamic given by:

$$\underline{\tilde{u}}_{k,i+1} = \underline{\tilde{u}}_{k,i} - K_i A \underline{\tilde{u}}_{k,i} + K_i \underline{y}_k, \tag{4.20}$$

$$= (I - K_i A)\, \underline{\tilde{u}}_{k,i} + K_i \underline{y}_k. \tag{4.21}$$

To prove that the closed loop is asymptotically stable, it must be shown that all eigenvalues of the matrix $(I - K_i A)$ lie strictly inside the unit circle. That means, the absolute values of all eigenvalues must satisfy:

$$|\lambda_j\,(I - K_i A)| < 1 \quad \text{with } j = 1, \ldots, 4. \tag{4.22}$$

The matrix $(I - K_i A)$ is given by:

$$(I - K_i A) = \begin{pmatrix} 1 - K_i & 0 & aK_i & 0 \\ 0 & 1 - K_i & bK_i & 0 \\ 0 & 0 & 1 - cK_i & 0 \\ 0 & 0 & 0 & 1 - dK_i \end{pmatrix}. \tag{4.23}$$

Since this matrix is upper triangular, its eigenvalues are given by the entries on the main diagonal. Consequently, the parameters a and b have no influence on the eigenvalues and thus do not affect stability.

For the first two eigenvalues, the condition $|1 - K_i| < 1$ must hold, which leads to the constraint:

$$0 < K_i < 2. \tag{4.24}$$

For the third eigenvalue, stability requires the following condition:

$$1 - cK_i < 1, \tag{4.25}$$
$$cK_i > 0, \tag{4.26}$$

which is automatically satisfied, as both c and K_i are positive by previous assumptions, respectively restrictions (4.24). The second condition is:

$$1 - cK_i > -1, \tag{4.27}$$
$$K_i < \frac{2}{c}. \tag{4.28}$$

Inserting the maximum value $c = 1$ leads to $K_i < 2$. Hence, no additional constraint arises for K_i from the third eigenvalue. Since c and d are defined in the same intervals and due to symmetry, the same argument applies to the fourth eigenvalue.
To summarize, the adaptation law given in Equation (4.23) ensures asymptotic closed-loop stability for all gain values in the interval: $0 < K_i < 2$.
This stability proof also applies to the 3D tracking case, as the same structural considerations hold. However, the assumption made in Equation (4.18), that the length corresponds to the larger eigenvalue and the width to the smaller one, is not always valid in 3D.
In particular, when the eigenvalues are close to each other, misassignments may occur, which can negatively affect stability. To address this issue, the eigenvector-based matching procedure described in Algorithm 1 can be used to ensure a consistent assignment of the different spatial dimensions.

Initialization There are three options to initialize the adaptation algorithm in each time step:

1. Initialize with the result of the Kalman filter with RM extension estimation using the following relations (mandatory in the first time step):

$$\tilde{x}_{k,1} = z_{x_{k|k}} \tag{4.29}$$
$$\tilde{y}_{k,1} = z_{y_{k|k}}, \tag{4.30}$$
$$\tilde{l}_{k,1} = 2\sqrt{e_{l_{k|k}}}, \tag{4.31}$$
$$\tilde{w}_{k,1} = 2\sqrt{e_{w_{k|k}}}. \tag{4.32}$$

2. Prediction of the VMM depending on the selected ODM:

$$\tilde{x}_{k,1} = f_x(\tilde{x}_{k-1,I}), \tag{4.33}$$
$$\tilde{y}_{k,1} = f_y(\tilde{y}_{k-1,I}), \tag{4.34}$$
$$\tilde{l}_{k,1} = \tilde{l}_{k,I}, \tag{4.35}$$
$$\tilde{w}_{k,1} = \tilde{w}_{k,I}. \tag{4.36}$$

3. Continue with the last VMM input of the previous time step:

$$\underline{\tilde{u}}_{k,1} = \underline{\tilde{u}}_{k-1,I}. \tag{4.37}$$

The most precise option is a prediction from the previous VMM parameters to the current time step. This makes the initial error $\epsilon_{k,1}$ smaller than with the other two methods. Thus, the number of iterations per time step can be kept small.

Implementation It was previously shown that the adaptation algorithm for the VMM input is asymptotically stable for all gain values in the interval $0 < K_i < 2$. However, in practical implementations, deviations from the theoretical assumptions may occur, potentially violating the conditions required for stability.
One such issue arises when, after a VMM update, the estimated width exceeds the length. If the eigenvector-based assignment procedure from Algorithm 1 is not applied, this leads to a misassignment of eigenvalues, as it is otherwise always assumed that the larger eigenvalue corresponds to the length. This problem can be addressed by enforcing correct ordering after the adaptation step or by reducing the gain K_i.
Moreover, it must be ensured that length and width remain positive after the update.
Due to stochastic fluctuations, parameters such as c and d may occasionally fall outside their intended ranges. According to Equation (4.28), an increased value of c implies a smaller allowable K_i to maintain asymptotic stability.
To mitigate all these potential sources of instability, a variable gain K_i with step-size control is proposed. In cases where the system temporarily violates assumptions, this approach adapts K_i to remain within the valid range. As an indicator for reducing K_i, the error norm ϵ_i defined in Equation (4.13) is monitored.
Algorithm 2 outlines a robust implementation of the VMM adaptation using variable gain control.

Algorithm 2 Random matrix approach with VMM adaptation

```
for k = 1, ..., K do
  Kalman filter prediction and update of kinematic and extent state (see Sections 2.4-2.5)
  if k == 1 then
    Initialize VMM input using Equations (4.29)–(4.32)
  else
    Predict VMM input using Equations (4.33)–(4.36)
  end if
  Compute initial error ε_0 from Equation (4.13)
  Initialize gain K_i such that 0 ≪ K_i ≪ 2
  for i = 1, ..., I do
    Store current VMM input parameters
    Update input using adaptation law (4.14) and run VMM
    Compute error ε_i from Equation (4.13)
    if ε_i > ε_{i-1} then
      Restore previous VMM input
      Reduce K_i (e.g., halve it)
    end if
  end for
  Return final VMM input as state estimate
end for
```

4.3. Expansion to multi-extended object tracking

In this section, the VMM Approach is extended to address MEOT. The content is based on the own conference paper [PH3]. The focus is on scenarios where objects may obscure one another but remain sufficiently separated, allowing for straightforward solutions to clustering and data association challenges using density-based spatial clustering of applications with noise (DBSCAN) [87] and a nearest neighbor approach. The overall approach continues to rely on Kalman filtering with RM extension estimation, where each track is managed by its own filter. Since object transitions are assumed to be independent, the prediction step remains unchanged, and the method described in 4.2 can be applied to each track alongside the Kalman filter prediction equations [17]. Figure 18 illustrates the adapted VMM framework.

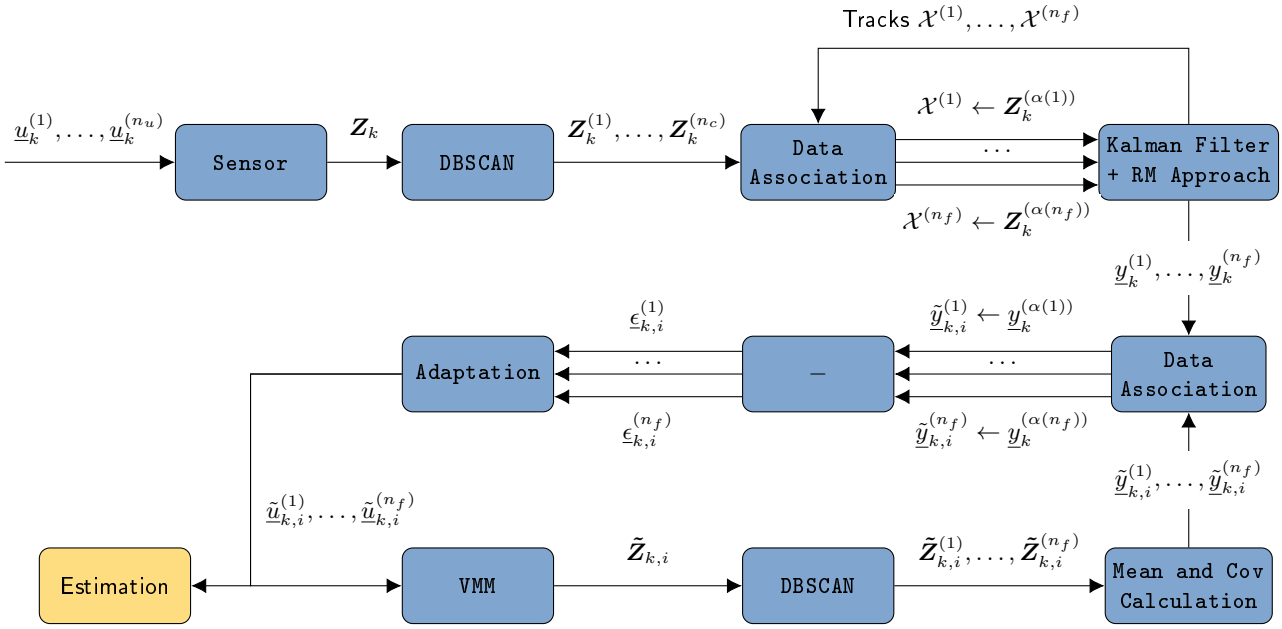


Figure 18: Reference model concept for multi-extended object tracking with an adaptation algorithm [PH3].

In MEOT, the input to the reference plant is a set of vectors $\underline{u}_k^{(1)},...,\underline{u}_k^{(n_u)}$, where n_u is the number of objects present and each vector $\underline{u}_k$ follows the definition given in (4.1). After the sensor generates a set of measurement $\boldsymbol{Z}_k$, the set is clustered into subsets $\boldsymbol{Z}_k^{(1)},...,\boldsymbol{Z}_k^{(n_c)}$ using DBSCAN [87], where n_c represents the number of clusters. DBSCAN requires the set of measurements and two tuning parameters as input: the maximum distance (Eps) and the minimum number of points (MinPts)[87, p. 230]. With appropriate parameter settings, each subset contains measurements generated by one single object.
The MEOT filter (Kalman Filter + RM Approach in Figure 18), maintains a set of Tracks $\mathcal{X}^{(1)},...,\mathcal{X}^{(1)}$, where each track includes the parameters of a Gaussian distribution for kinematic state estimation and the parameters of an inverse Wishart distribution for extension estimation. For the current iteration's update step, the measurement clusters must be assigned to the predicted tracks from the previous iteration. This assignment is performed by calculating the average Mahalanobis distance between all measurements in cluster j and track e, defined as:

$$c_{e,j} = \frac{1}{n_j}\sum_{ii=1}^{n_j}\left(H_k\underline{x}_{k|k-1}^{(e)} - \underline{z}_{j,ii}\right)\left(S_{k|k-1}^{(e)}\right)^{-1}\left(H_k\underline{x}_{k|k-1}^{(e)} - \underline{z}_{j,ii}\right)^{\top}. \tag{4.38}$$

This allows to build up a cost matrix for the total number of n_f tracks and n_c clusters:

$$C = \begin{pmatrix} c_{1,1} & c_{1,2} & \dots & c_{1,n_c} \\ c_{2,1} & c_{2,2} & \dots & c_{2,n_c} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n_f,1} & c_{n_f,2} & \dots & c_{n_f,n_c} \end{pmatrix}. \tag{4.39}$$

The maximum number of cluster-to-track associations is given by $a = \min(n_c, n_f)$. During each of the I iterations, the minimum value in the cost matrix C is evaluated, and the corresponding cluster with index j is assigned to track e if $c(e, j)$ does not exceed a predefined threshold. After the assignment, the values in C for row e and column j are set to infinity, ensuring that each cluster is assigned to only one track, and each track receives at most one cluster. This method for solving the data association problem closely resembles the suboptimal nearest-neighbor (SNN) approach [88, p. 293]. In the update step, each track is updated with its assigned cluster or, if no cluster is assigned, treated as a miss detection track. For the notation of the assignment in Figure 18, the function $\alpha(e)$ is used and returns the index of the cluster that is assigned to track e by the SNN. Note that this is a very simple approach to the MOT problem, more modern tracking methods do not make a fixed assignment but consider multiple hypotheses. For example, the GLMB filter [89] considers every global assignment hypothesis. The disadvantage, however, is the enormous computational effort, since the expansion estimation within each global hypothesis would have to be adapted with VMMs. For this section, it is therefore assumed that the measurement data can be easily clustered and that an assignment is unambiguous. The focus is on the occlusions.

The virtual plant consists of the blocks *VMM*, *DBSCAN* and *Mean and Cov Calculation* and is the Multi-Object version of the second line in Figure 14. The input to the VMM is a set of suggested estimates $\underline{\tilde{u}}_{k,i}^{(1)}, \dots, \underline{\tilde{u}}_{k,i}^{(n_f)}$ and the VMM outputs a set of the artificial measurements $\tilde{Z}_{k,i}$. The artificial measurements are then clustered and for each cluster, the mean and covariance matrix are calculated as well as the eigenvalues of this covariance matrix. The result is structured as a set of vectors $\underline{\tilde{y}}_{k,i}^{(1)}, ..., \underline{\tilde{y}}_{k,i}^{(n_f)}$, with each vector following the definition given in (4.4).

The adaptation law for the multi-object case is analogous to the single-object case and is given by:

$$\underline{\tilde{u}}_{k,i+1}^{(e)} = \underline{\tilde{u}}_{k,i}^{(e)} - K_i^{(e)} \underline{\epsilon}_{k,i}^{(e)}. \tag{4.40}$$

In this equation, the error $\underline{\epsilon}_{k,i}^{(e)}$ represents the difference between the VMMs result $\underline{\tilde{y}}_{k,i}^{(e)}$ and the corresponding reference result assigned that has been assigned to this VMM output $\underline{y}_k^{(\alpha(e))}$. The same SNN method can be employed for data association.

The adaptation and the evaluation of the virtual plant are done in I iterations for every time step k. In the last iteration, the errors $\epsilon_{k,I}$ is expected to be small and $\underline{\tilde{u}}_{k,I}^{(1)}, \dots, \underline{\tilde{u}}_{k,I}^{(n_f)}$ becomes an estimate for position and extension.

VMM input initialization and prediction At each time step k, an initial set of VMM input set $\underline{\tilde{u}}_{k,1}^{(1)}, \dots, \underline{\tilde{u}}_{k,1}^{(n_f)}$ is required. For the initial time step $k = 1$, the VMM inputs can be initialized using the outputs of the Kalman filters with RM extension estimation:

$$\underline{\tilde{u}}_{1,1}^{(1)}, \dots, \underline{\tilde{u}}_{1,1}^{(n_f)} = \underline{y}_1^{(1)}, \dots, \underline{y}_1^{(n_f)}. \tag{4.41}$$

For subsequent time steps, simply initializing the VMM inputs in the same manner would cause the adaptation process to restart from scratch at each step. To enhance efficiency, it is more effective to predict the VMM inputs for the next time step using the results from the previous step:

$$\underline{\tilde{u}}_{k,1}^{(1)}, \dots, \underline{\tilde{u}}_{k,1}^{(n_f)} = f\left(\underline{\tilde{u}}_{k-1,I}^{(1)}, \dots, \underline{\tilde{u}}_{k-1,I}^{(n_f)}\right), \tag{4.42}$$

where f represents the transition function. To perform this transition, e.g. for a constant velocity (CV) model, the velocities v_x and v_y resp. v_z in 3D must either be integrated into the vector $\underline{\tilde{u}}$ or extracted from the corresponding track.

Implementation Algorithm 3 presents a suggested implementation of the MEOT filtering approach. To ensure the safe convergence of the adaptation procedure, the adaptation gain K_i is adjusted: if the error increases after an iteration, the gain is reduced accordingly. This dynamic adjustment helps stabilize the adaptation process, preventing divergence and improving overall robustness.

Algorithm 3 Multi-extended object tracking with a virtual measurement model

for k=1,...,K **do**
  Prediction of kinematic [17] and extent using (2.13)-(2.14) for each track
  Measurement clustering using DBSCAN [87]
  Cluster-to-track assignment using the cost matrix (4.39)
  Update of kinematic [17] and extent using (2.17)-(2.23) for each track
  VMM input initialization (4.41) resp. prediction (4.42)
  Get initial error $\epsilon_0^{(e)} = || \left(\underline{y}_k^{(e)} - \underline{\tilde{y}}_{k,1}^{(e)} \right) ||$ for each object
  Initialize $K_i^{(e)}$ $(0 << K_i^{(e)} << 2)$ for each object
  for i=1,...,I **do**
    Store VMM input for each object
    VMM input adaptation (4.40)
    Run VMM to generate artificial measurements
    Artificial measurement clustering using DBSCAN [87]
    Evaluate mean and cov for each cluster
    Data Association $\underline{\tilde{y}}_{k,i}^{(e)} \leftarrow \underline{y}_k^{(\alpha(e))}$
    get error $\epsilon_i^{(e)} = || \left(\underline{y}_k^{(e)} - \underline{\tilde{y}}_{k,i}^{(e)} \right) ||$ for each object
    for e=1,...,n_f **do**
      if $\epsilon_i^{(e)} > \epsilon_{i-1}^{(e)}$ **then**
        restore previous VMM input parameters for $\underline{\tilde{y}}_{k,i}^{(e)}$
        decrease K_i (e.g. divide by 2)
      end if
    end for
  end for
  provide last VMM input parameters as state estimation
end for

4.4. Gaussian processes regression model with VMMs

The reference model approach from section 4.1 involves the use of a VMM to generate artificial measurements for a suggested estimate $\underline{\tilde{u}}_k$ and a comparison of their statistical characteristics $\underline{\tilde{y}}_k$ with the result of the RM Kalman filter $\underline{y}_k$. If $\underline{y}_k$ and $\underline{\tilde{y}}_k$ match, then the suggested estimate $\underline{\tilde{u}}_k$ should also match the reference $\underline{u}_k$. Previously, this estimate was adapted multiple times within each time step using the integral-based adaptation law (see Eq. (4.14)). Each iteration required the generation of artificial measurements, which, depending on the measurement and shape model, could become computational expensive.
In this section, an alternative approach is proposed: Using a virtual measurement model to generate training data for a GPRM [62] during offline preprocessing. During the actual filtering process, the trained GPRM's prediction function is employed to estimate the target's position and extent based on the filtered CoG and the covariance matrix of the measurements. This approach eliminates the need for multiple adaptation iterations per time step and allows for the prediction of uncertainties. This section is based on the own conference paper [PH4] where the GPRM-VMM method was applied to 2D-tracking scenarios. While the approach could, in theory, be extended to 3D tracking, the curse of dimensionality significantly increases the number of required samples, making the GPRM less efficient. Therefore, this section focuses on explaining the GPRM approach with VMMs specifically for 2D tracking.
Figure 19 illustrates the processing chain of the proposed extension estimation and tracking approach, which consists of two main steps: the training of the GPRM during preprocessing and online EOT using a Kalman filter with random matrix extension estimation followed by the *correction* with the GPRM's prediction.

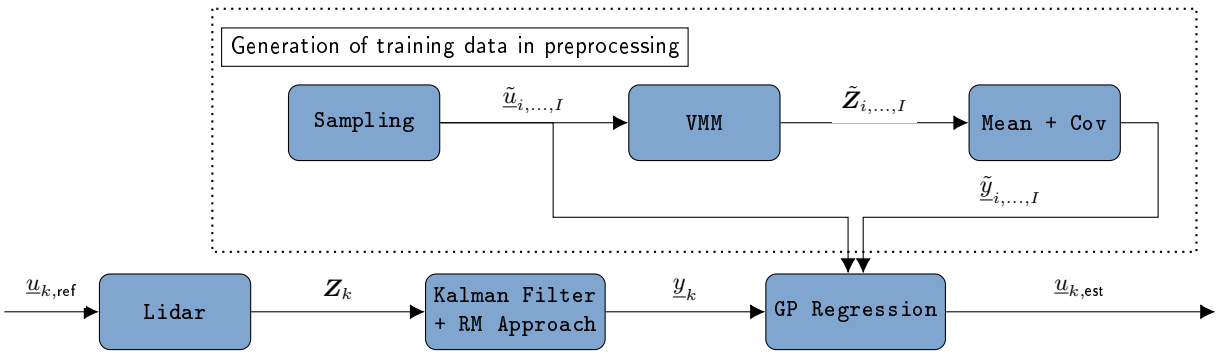


Figure 19: Processing chain of the proposed extension estimation approach [PH4].

Generation of training data The training of the GPRM requires a total of I VMM input samples and their corresponding outputs:

$$\underline{\tilde{u}}_i = \left(\tilde{x}_i, \tilde{y}_i, \tilde{l}_i, \tilde{w}_i, \tilde{\psi}\right), \tag{4.43}$$

$$\underline{\tilde{y}}_i = \left(\tilde{z}_{x,i}, \tilde{z}_{y,i}, z_{el,i}, z_{ew,i}\right), \tag{4.44}$$

whereas $\tilde{z}_x = 2\sqrt{e_l}$ and $\tilde{z}_x = 2\sqrt{e_w}$.
To account for objects with all possible orientations, $\tilde{\psi}_i$ is included in the input vector $\underline{\tilde{u}}_i$ for the training data. However, the orientation $\psi_{k,\text{est}} = \text{atan2}\left(v_{y_{k,\text{est}}}, v_{x_{k,\text{est}}}\right)$ is already available from the object's kinematics and thus not estimated by the GPRM. The positions $(\tilde{x}_i, \tilde{y}_i)$ of the input samples must cover the entire observation area, and their parameters $\tilde{l}_i$ and $\tilde{w}_i$ must span all potential objects extents.

These samples are drawn from uniform distributions with $i = 1, \ldots, I$:

$$\tilde{x}_i \sim \mathcal{U}(x_{min}, x_{max}), \quad (4.45) \qquad \tilde{\psi}_i \sim \mathcal{U}(0, 2\pi), \quad (4.48)$$
$$\tilde{y}_i \sim \mathcal{U}(y_{min}, y_{max}), \quad (4.46) \qquad c_{w,i} \sim \mathcal{U}(0.1, 0.9), \quad (4.49)$$
$$\tilde{l}_i \sim \mathcal{U}(l_{min}, l_{max}), \quad (4.47) \qquad \tilde{w}_i = c_{w,i}\tilde{l}_i. \quad (4.50)$$

The scaling factor $c_{w,i}$ ensures that the width is always smaller than the length. For each input sample $\underline{\tilde{u}}_i$, artificial measurements $\tilde{\boldsymbol{Z}}_i$ are generated using the VMM and the elements of the output vector $\underline{\tilde{y}}_i$ are computed as described in Section 4.1, Equation (4.4).

Linear correlations in training data Before designing the GPRM, an analysis of the training data is performed to select suitable hyperparameters and kernel functions. It is expected that each VMM input parameter in $\underline{\tilde{u}}$ primarily influences a specific output parameter in $\underline{\tilde{y}}$. For instance, the position $(\tilde{x}, \tilde{y})$ corresponds to the CoG $(\tilde{z}_x, \tilde{z}_y)$, the length $\tilde{l}$ to the square root of the larger eigenvalue $\tilde{z}_{el}$, and the width $\tilde{w}$ to the square root of the smaller eigenvalue $\tilde{z}_{ew}$.
Table 1 presents the linear correlation coefficients between the input parameters $\underline{\tilde{u}}$ and the output parameters $\underline{\tilde{y}}$.

Table 1: Linear correlation coefficients between training data. High correlations (≥ 0.8) are highlighted in yellow resp. blue (≥ 0.5). 10,000 samples [PH4].

		$\underline{\tilde{u}}$				
		$\tilde{x}$	$\tilde{y}$	$\tilde{l}$	$\tilde{w}$	$\tilde{\psi}$
$\underline{\tilde{y}}$	$\tilde{x} - \tilde{z}_x$	0.80	0.00	0.00	0.02	0.00
	$\tilde{y} - \tilde{z}_y$	0.00	0.80	0.00	−0.01	0.02
	$\tilde{z}_{el}$	0.00	0.00	0.96	0.66	0.01
	$\tilde{z}_{ew}$	0.00	0.00	0.61	0.90	0.02

Note that the correlation was calculated to the shift $\tilde{x} - \tilde{z}_x$ rather than for the CoG itself. This prevents the correlation from being overly close to 1, as high values of $\tilde{x}$ and $\tilde{y}$ would otherwise dominate the values in $\tilde{l}$ and $\tilde{w}$.

Fitting the GPRM In the following, two different GPRM approaches are parametrized, both following the structure shown in Figure 20.

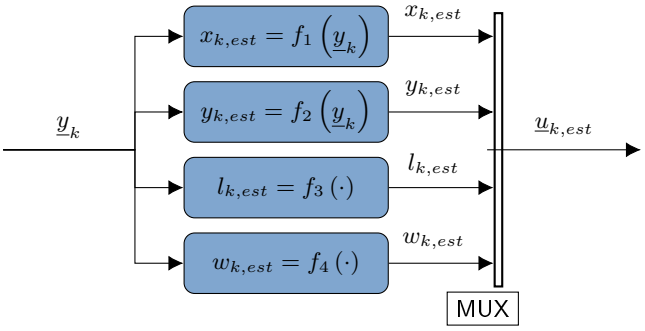
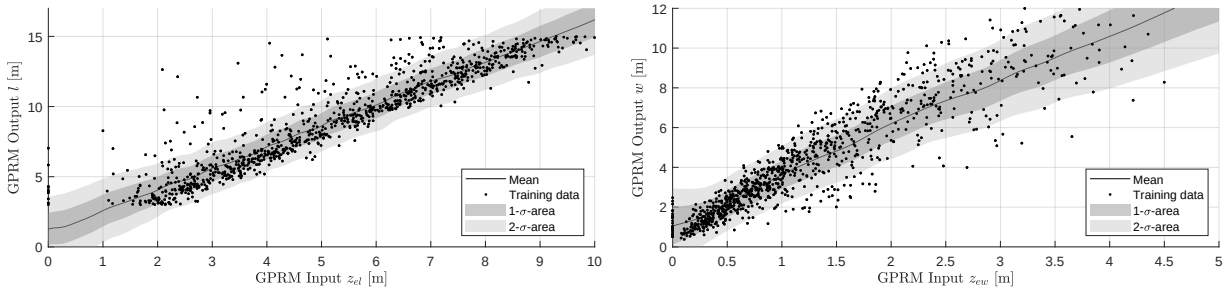


Figure 20: Gaussian processes regression. Full l/w predictors: $f_{3,4}(\underline{y}_k)$. Single l/w predictors: $f_3(z_{el})$ and $f_4(z_{ew})$ [PH4].

The most straightforward approach is to use a multivariate Gaussian process for each desired parameter in $\underline{u}_{k,est}$, using the entire vector $\underline{y}_k$ as input for each prediction. This approach is referred to as *full predictors*. However, due to the strong correlations between z_{el} and length and between z_{ew} and width, an alternative is to predict length and width using only z_{el}, respectively z_{ew}. This approach is referred to as *single predictors*.

Single l/w predictors The parameterization begins with the single predictors, as the one-dimensional Gaussian process used in this approach allows for straightforward visualization. Figure 21 shows 1,000 data points sampled within the range $x_{\min} = y_{\min} = -50\,\mathrm{m}$, $x_{\max} = y_{\max} = 50\,\mathrm{m}$, with lengths varying between $l_{\min} = 3\,\mathrm{m}$ and $l_{\max} = 15\,\mathrm{m}$. The sensor is located



(a) length $l = f_3(z_{el})$.

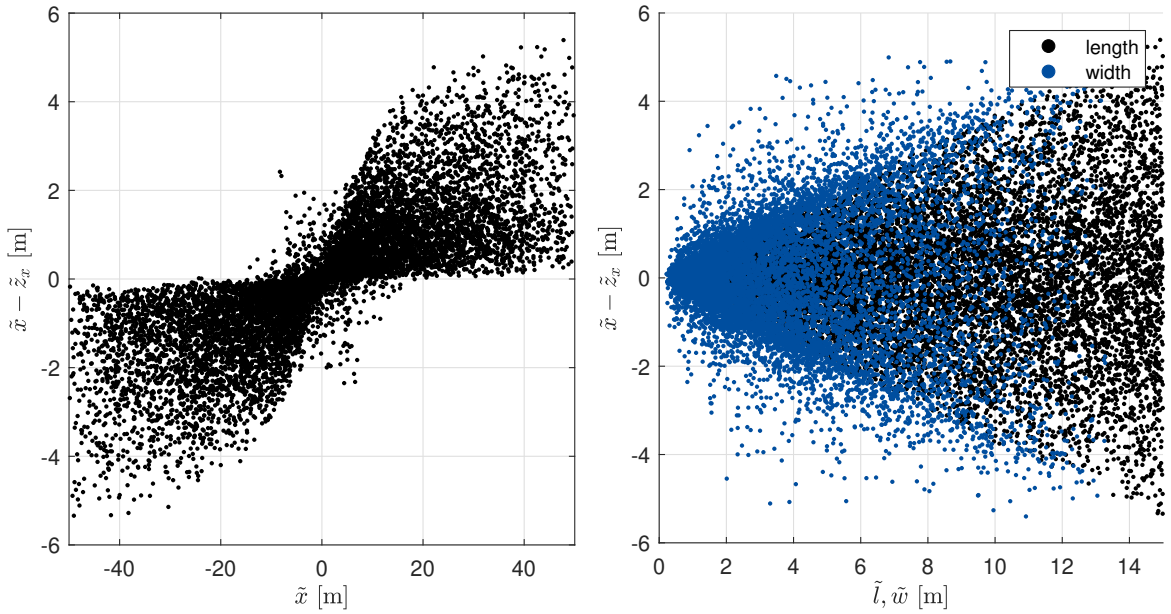
(b) length $w = f_4(z_{ew})$.

Figure 21: Posterior Gaussian processes for single predictors [PH4].

at $(0, 0)\,\mathrm{m}$ and any objects overlapping with this location are resampled until they no longer do. For the Gaussian processes, a kernel composed of the sum of a squared exponential kernel and a dot product kernel is used:

$$k(x_p, x_q) = \sigma_f^2 \exp\left(-\frac{1}{2}\frac{(x_p - x_q)^2}{2\ell^2}\right) + \sigma_n^2 \delta_{pq} + x_p x_q \tag{4.51}$$

with signal variance $\sigma_f^2 = 1\,\mathrm{m}^2$, length scale $\ell = 0.1\,\mathrm{m}$ and noise variance $\sigma_n^2 = 100\,\mathrm{m}$. These parameters were found iteratively to ensure that the posterior Gaussian processes in Figure 21 (calculated using (2.83)) fitted well with the training data. A good fit is indicated by approximately 95% of the samples falling within the 2σ confidence interval. The advantage of the single predictor is its user-friendly evaluation by visualization of the Gaussian process as well as the computational efficiency. However, it has a significant disadvantage: As shown in Figure 21, the uncertainty in the posterior Gaussian process remains high and cannot be reduced by adding more training data. The high noise term is caused by the unknown and varying position and orientation, the VMM itself works without noise and is deterministic. Without considering the object's orientation and position, the estimation will not become more precise. This is even more evident in the estimation of the position: If a CoG (z_x, z_y) is provided by the Kalman filter, the distance this CoG has shifted from the actual object position (x, y) towards the sensor must be determined. This shift depends on the position and on the extension parameters, as Figure 22 shows. If an object is placed at $x = 0\,\mathrm{m}$, the shift is mainly in the y-direction. The further away the object is in the x-direction, the larger the x-shift becomes which is expressed by the linear correlation coefficient of 0.8 in Table 1. The size of this shift is mainly determined by the expansion parameters. Due to the symmetry to the x-axis, the linear correlation coefficient is 0 and cannot express this relationship. Consequently, to estimate the position and for a better expansion estimate, multivariate Gaussian processes are used in the following.



(a) $\tilde{x}$ position. (b) Length $\tilde{l}$ and width $\tilde{w}$.

Figure 22: Correlations to the shift $\tilde{x} - \tilde{z}_x$. 10,000 samples [PH6].

Multivariate predictors For the multivariate predictor the same combination of a squared exponential kernel with a dot product kernel in its multivariate notation (see (2.81)-(2.82)) is used, with signal variance $\sigma_f^2 = 1\,\mathrm{m}^2$, noise covariance $\sigma_n^2 = 1\,\mathrm{m}$, length scale $\ell_{1,2} = 10\,\mathrm{m}$ for x/y-position, $\ell_{3,4} = 1\,\mathrm{m}$ for length and width and $\ell_5 = 0.1$ for orientation. With these hyperparameters, the four Gaussian processes given in Figure 20 are implemented in the form $f_i\left(\underline{y}_k\right)$.

5. Shape classification with virtual measurement models

In EOT [81], the shape of objects is not always known. In group tracking, it is even possible that the shape changes if the individuals change their formation. In many situations, at least one of the shapes considered in this work (see section 3) can represent the object well, but which one fits best is not necessarily known in advance. Therefore the goal of the shape classification is to use virtual measurement models, to identify the shape of the object respectively to select the best fitting shape from a predefined list.
Shape classification has been previously considered e.g. in [90], where a Gaussian Processes-[77] based extension estimation algorithm was used, followed by a feature-based Bayesian classifier. However, in this work, arbitrary shapes are not considered.
Own publications on this subject: This section is based on the own conference papers [PH5] and [PH6]. In [PH5], the framework for shape classification with VMMs, that is shown in Figure 23 and explained in the further course of this section, was introduced. The shape classification was done by comparing real measurements with artificial measurements, generated by shape-specific VMM's using a histogram based approach. Whereas [PH5] was limited to 2D-EOT, 3D-EOT was considered in [PH6] and instead of the histogram-based classification, Chamfer distances were used.
The framework for EOT and shape classification with VMMs is shown in Figure 23.

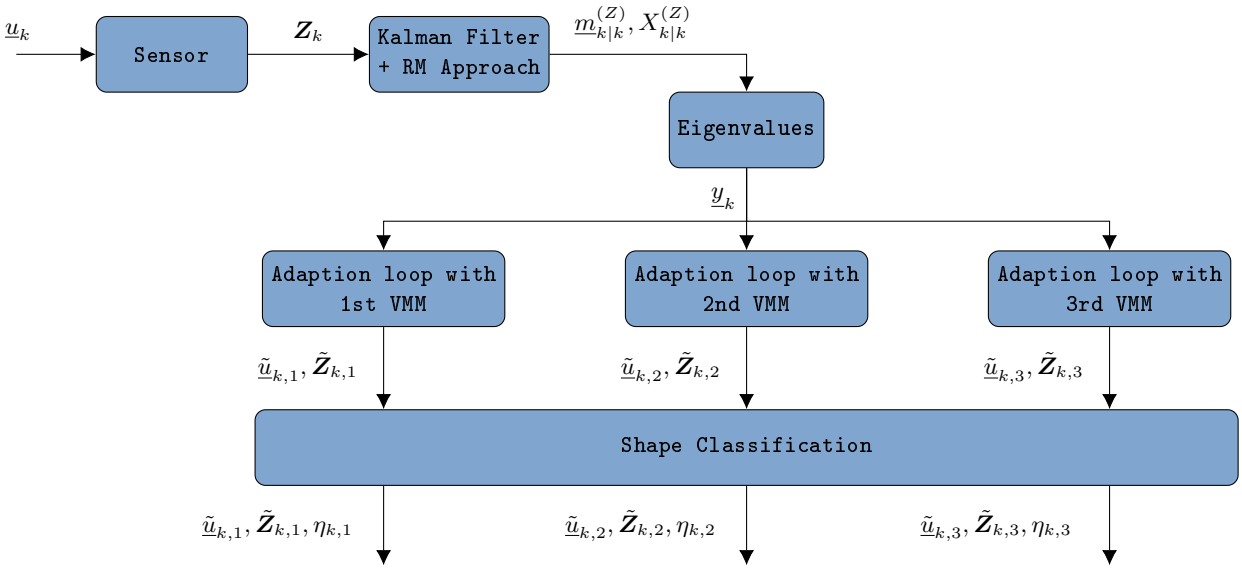


Figure 23: Extended object tracking and shape classification with virtual measurement models [PH5, PH6].

At the beginning of the framework shown in Figure 23, data acquisition is followed by filtering using a Kalman filter with random matrix extension estimation. Once the eigenvalues are calculated, the result is rearranged into the vector $\underline{y}_k$ as described in (4.4). So far, this is exactly the same structure from the previous section that was denoted as the reference plant. However, the reference plant is now followed by several parallel adaptation loops, each utilizing a different virtual measurement model corresponding to a specific shape. In [PH5], rectangular, triangular and elliptical shapes were used, while in [PH6] ellipsoids, cones and cuboids were employed.
Each adaptation loop involves generating artificial measurements $\tilde{\boldsymbol{Z}}_{k,i}$, evaluating them and denoting them as the virtual output $\underline{\tilde{y}}_{k,i}$. This output is then compared with the actual output $\underline{y}_k$ and used for the adaptation of the estimation $\underline{\tilde{u}}_{k,i}$ as illustrated in Figure 16. Note that the virtual output $\underline{\tilde{y}}_{k,i}$ can not be found in Figure 23 since it is an internal element of the adaptation loop and therefore only visible when visualizing the inside of the loop, as done in Figure 16.
The results of each adaptation loop can be interpreted as follows: If assumed that the shape

of the object is elliptical, the "Adaption loop with ellipse VMM" provides us the object's parameters $\underline{\tilde{u}}_{k,ell}$. These parameters cause the artificial measurements $\tilde{\boldsymbol{Z}}_{k,ell}$ and their statistical moments match the random matrix result. Note that this works for all shape models: Even if the wrong shape is used, it is still possible to fit the shape parameters so that the mean and the covariance matrix of artificial measurements match the random matrix result. However, this also means that the residual error in the adaptation loops (e.g. $\underline{\tilde{y}}_{k,ell} - \underline{y}_k$) is not a criterion for the classification, since it is minimized for each shape. However, for classification it is possible to compare the artificial measurements $\tilde{\boldsymbol{Z}}_{k,i}$ generated by the corresponding VMM in its last iteration with the real measurements $\boldsymbol{Z}_k$. Note that mean and covariance of those measurement sets can not be used for classification, since in all adaptation loops, the VMM was adapted until those fit. In the following, two comparison methods are presented for classification with VMMs.

5.1. Shape classification based on histograms

To estimate the shape, the real measurements $\boldsymbol{Z}_k$ are compared with the artificial measurements of the VMMs ($\tilde{\boldsymbol{Z}}_{k,1}$, $\tilde{\boldsymbol{Z}}_{k,2}$ and $\tilde{\boldsymbol{Z}}_{k,3}$). This is done in target coordinates which allows to accumulate measurements over several time steps. To check, how well a certain shape fits, the real and the artificial measurements are both transformed into the target coordinate system. Here, the measurements are divided into several bins according to Figure 24 based on the estimated extension parameters and the measurements in each bin are counted. The root mean square

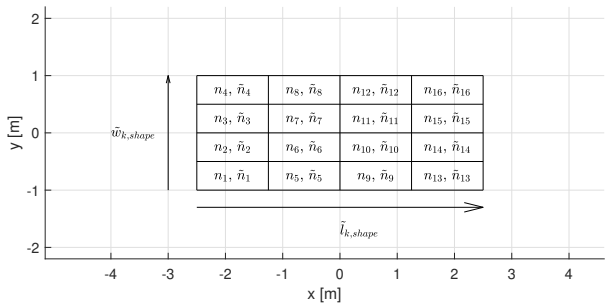


Figure 24: Histogram-based shape classification [PH5].

error of the normalized number of real measurements compared to the normalized number of artificial measurements is used as the criterion. The shape classification can be either done at the end of the scenario or in live tracking as a sliding window that uses the measurements of the L last time steps.

5.2. Shape classification based on Chamfer distances

Previously, shape classification within the VMM framework relied on comparing histograms of real measurements $\boldsymbol{Z}_k$ with those of artificial measurements for each shape hypothesis [PH5]. However, this method did not consistently achieve high classification quality, especially in scenarios with moderate to high measurement noise [PH5].
To enhance classification accuracy, the use of the Chamfer distance [91] as a metric to evaluate the similarity between the two sets of measurements is proposed. The Chamfer distance between two 3D point clouds S_1 and S_2 is given by [92, p. 4] [93, p. 4]:

$$d_C(S_1, S_2) = \frac{1}{|S_1|} \sum_{x \in S_1} \min_{j \in S_2} ||x - y||^2 + \frac{1}{|S_2|} \sum_{y \in S_2} \min_{x \in S_1} ||y - x||^2, \tag{5.1}$$

whereas $|S_i|$ is the number of elements in S_i. The classification scores for each shape hypothesis are then determined as follows:

$$\eta_{k,ell} = d_C\left(\tilde{\boldsymbol{Z}}_{k,ell}, \boldsymbol{Z}_k\right), \tag{5.2}$$

$$\eta_{k,con} = d_C\left(\tilde{\boldsymbol{Z}}_{k,con}, \boldsymbol{Z}_k\right), \tag{5.3}$$

$$\eta_{k,cub} = d_C\left(\tilde{\boldsymbol{Z}}_{k,cub}, \boldsymbol{Z}_k\right). \tag{5.4}$$

The shape with the lowest score is considered as the estimated shape. To improve accuracy further, real measurements from multiple time steps are accumulated using a sliding window approach with a window length L. These accumulated measurements are then transformed into the BODY coordinate system of the VMM's state estimation and compared with the virtual measurements generated by the corresponding VMM.

6. Results

This section presents the results of the proposed methods, based on both simulation studies and real-world lidar data, to evaluate their performance and applicability across a range of tracking scenarios.

6.1. Extension estimation with an elliptical object

In this section, the VMM framework is evaluated in its simplest form: observing a single elliptical object using a lidar sensor. These results were originally published in the journal paper [PH1], where the VMM framework was first introduced. For the modeling of the object's dynamic, a Singer's ODM is used. The scenario is 29 s long with a sampling period $T = 0.1\,\mathrm{s}$. The kinematic state $(x, y, v_x, v_y, a_x, a_y)'$ is initialized as $(-40\,\mathrm{m}, -10\,\mathrm{m}, 3\,\mathrm{m/s}, 0\,\mathrm{m/s}, 0\,\mathrm{m/s^2}, 0\,\mathrm{m/s^2})$. Acceleration is affected by white Gaussian noise with standard deviation $\sigma_{ax} = \sigma_{ay} = 0.01\,\mathrm{m/s^2}$. The object has a length of $l = 5\,\mathrm{m}$ and a width of $w = 2\,\mathrm{m}$. The lidar sensor is located at the origin, with a resolution of 1° and measurements are generated using the lidar sensor model for contour measurements as presented in section 3. The measurements are affected by white Gaussian noise with standard deviations $\sigma_x = \sigma_y = 0.5\,\mathrm{m}$. The measurements are first filtered using the random matrix algorithm resulting in the filtered center of gravity and the filtered covariance matrix. The Kalman filter is initialized by the measurements from the first time step with an initial uncertainty $P = \mathrm{diag}\,((10, 5, 1))$. For the random matrix algorithm, a parameter of $\tau = 0.5$ and an initial value $\nu = 4$ are used. Figure 25 shows the filtered center of gravity and the filtered $1\sigma-$ellipses of the measurements in black.

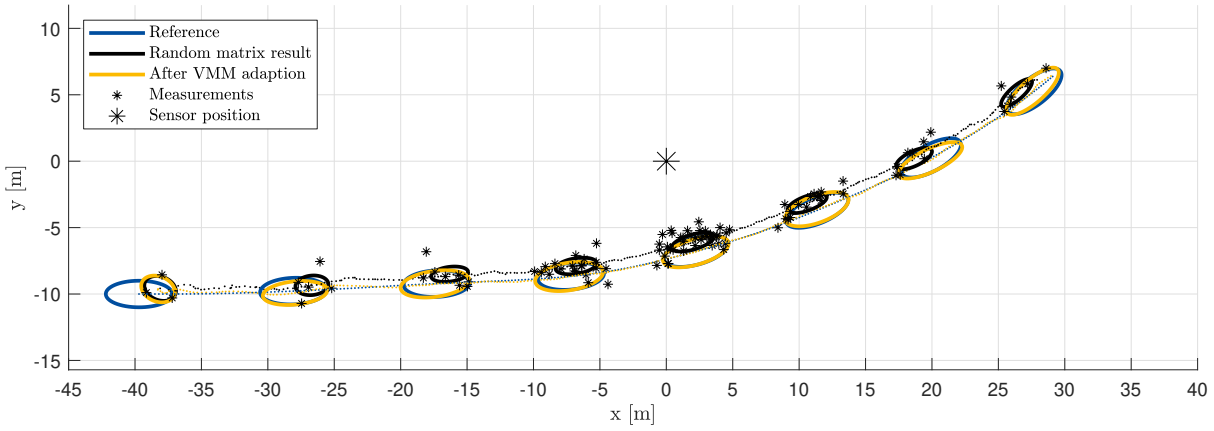
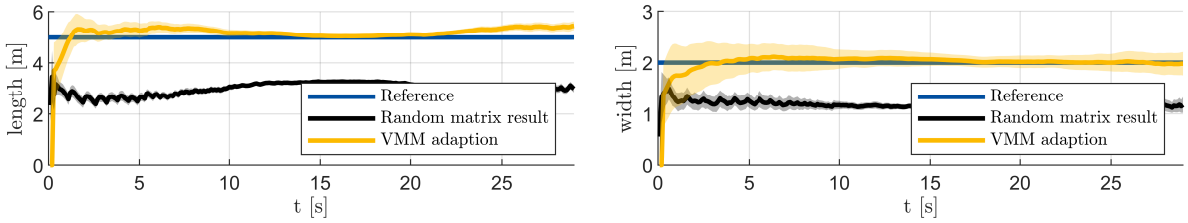


Figure 25: Tracking results for a Singer's ODM observed by a lidar sensor. Linear Kalman filter implementation. The extension estimation and the measurements are shown every 40 time steps [PH1, p. 235].

In Figure 25, it can be observed that the center of gravity of the measurements, respectively the position of the black ellipse, filtered by the random matrix algorithm, is shifted towards the sensor. This shift occurs because measurements only appear on the object's contour facing the sensor, causing a bias in the overall trajectory. Additionally, the filtered 1σ-ellipse is smaller in both directions compared to the object's extent and its size depends on the position of the object in the sensor coordinate frame, even though the actual size of the object remains constant. Therefore, using a constant scaling factor to estimate the extent from the filtered 1σ-ellipse (respectively the covariance matrix Σ_z) would not be appropriate.
The VMM has the same properties as the real sensor, only the resolution is set to 0.01° and no measurement noise is assumed. Using the VMM and the proposed adaptation algorithm, the extension estimation is adapted for two iterations in each time step, as proposed in algorithm 2, using a variable gain starting with $K_i = 0.1$. If the error ϵ increases after an iteration, the previous values are reconstructed and the gain K_i is divided by 2 for the next iteration. In the

next time step, K_i is set to 0.1 again. The results after the adaptation are shown in Figure 25 in red.
With the adaptation presented in this paper, the bias in the trajectory estimation is removed and the estimated extension matches the reference extension very precisely. However, the accuracy after adaptation depends on the precision of the random matrix results. In the first estimate (shown at $\approx (-38\,\mathrm{m}|-10\,\mathrm{m})$, the filter has not yet fully converged, so even the adaptation cannot give an accurate result.

Monte Carlo simulation - single trajectory The scenario shown in Figure 25 is evaluated over 100 Monte Carlo (MC) runs. While the trajectory is generated only once, measurements are independently generated for each MC run. The results of the extension estimation are shown in Figure 26.



(a) Length (b) Width

Figure 26: Extension estimation (mean and standard deviation) averaged over 100 MC runs [PH1].

After approximately $15-20$ seconds, the object passes the sensor, generating the highest number of measurements at this point. Consequently, the standard deviations of all extension estimations are at their lowest. The results of the random matrix approach are discussed first: During the drive-by at $15-20$ seconds, the sensor observes the entire length of the object, leading to a peak in the length estimation , as shown in Figure 26a. When mainly the front or the back of the object is visible, the length estimation is smaller. A similar but opposite behavior is observed for the width estimation in Figure 26b. When the front or the back of the object is heading towards the sensor, the entire width is observed, causing the width estimation to increase at the beginning and end of the scenario.
Another observation is that both the length and width estimation oscillate at the beginning of the scenario. This effect can be explained by considering the number of measurements received at each time step, as shwon in Figure 27.

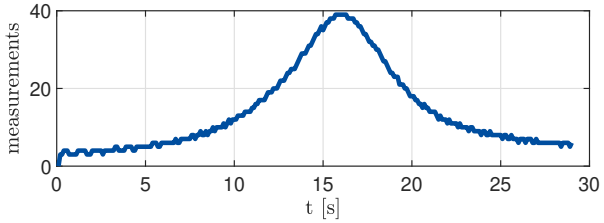


Figure 27: Number of measurements received at each time step. Since the trajectory is identical across all MC runs and the detection probability is 100%, the number of measurements in each MC run remains constant.

At $t = 1.6\,\mathrm{m}$, four noisy measurements are generated because four lidar beams intersect with the object's contour. As the object moves slightly to the right, only three measurements are generated, giving the appearance that the object is smaller. In the early time steps, this oscillation in the number of measurements results in a corresponding oscillation in the extension estimation. As more measurements are gathered and the distance between them decreases, this

effect diminishes.
After applying the VMM adaptation, the estimated extension aligns almost perfectly with the reference when the object is near the sensor. However, when fewer measurements are received, the VMM approach tends to slightly overestimate the extension.

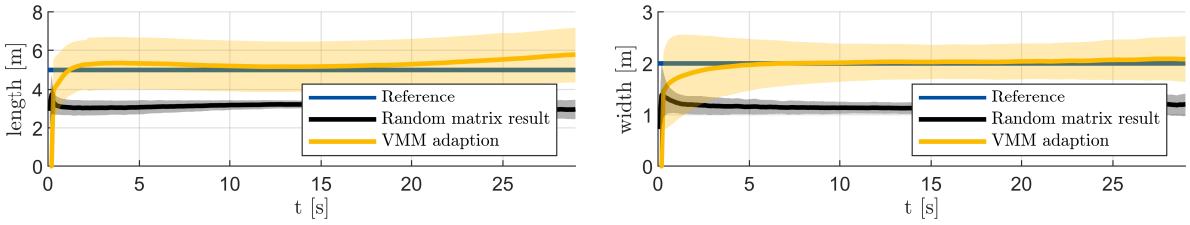
Monte Carlo simulation - random trajectories To demonstrate that the VMM and its adaptation are effective beyond a single trajectory, a MC simulation is performed with randomly generated trajectories. In each MC run, the initial position is sampled from a Gaussian birth density with an expected value of

$$\mathbb{E}(\underline{x}_0) = (-40, -20, 3, 0, 0, 0)' \tag{6.1}$$

and covariance matrix

$$\mathrm{Cov}(x_0) = \mathrm{diag}((10, 10, 0.01, 0.01, 0.001, 0.001)). \tag{6.2}$$

Process noise is generated independently for each trajectory. The results of the extension estimation are shown in Figure 28.



(a) Length (b) Width

Figure 28: Extension estimation averaged over 1000 MC runs with random trajectories [PH1]

.

The extension estimation shows a slight overestimation, with this bias increasing toward the end of the scenario when the target is typically farther from the sensor. The standard deviation is also higher compared to the previous simulation with a single trajectory. This increase is expected, as some trajectories pass close to the sensor while others do not. The oscillations in the number of measurements and the results of the random matrix approach, as explained in the previous section, are not observed in this scenario with random trajectories since it depends on geometric properties of a single trajectory.

Studies on stability and computational cost Section 4.2 shows that the adaptation algorithm is asymptotically stable for $0 < K_i < 2$. To demonstrate this, the scenario shown in Figure 25 is evaluated for different K_i values. The value of K_i is held constant throughout the scenario and is not adjusted, even if the error increases. At each time step, the VMM input is initialized with the random matrix result using (4.29)-(4.32) and nine iterations are performed. Figure 29 shows the error (4.13) per iteration, averaged over all time steps.
For $K_i < 2$, the error converges, as illustrated in Figure 29 for $K_i = 0.1$ and $K_i = 1$. However, when $K_i > 2$, the algorithm diverges with the first two eigenvalues from (4.23) falling outside the unit circle. In these cases, the third and fourth eigenvalues may occasionally remain stable, depending on whether c and d stay within their assumed limits. However, to ensure stability across all scenarios, using a variable K_i is strongly recommended, as detailed in section 4.2. By incorporating the VMM prediction from (4.33)-(4.36), a smaller K_i can be selected for robustness and the number of iterations can be reduced. This significantly lowers the computational cost, which is closely tied to the the number of iterations, as shown in Table 2. The Kalman Filter with random matrix extension estimation requires an average of 0.14 ms per time step. The main

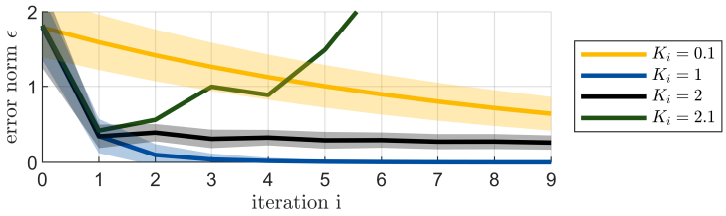


Figure 29: Error norm (mean and standard deviation), averaged over all time steps. The standard deviation for $K_i = 2.1$ is not shown since its region would cover the whole Figure [PH1].

Table 2: Computational cost per time step, averaged over 100 MC runs. Matlab-Implementation on an Intel Core i7-3720QM CPU @2.6GHz [PH1].

number of iterations	1	2	3	5	10
computational cost [ms]	7.0	15.7	23.5	40.7	52.1

reason for the high computational cost is the generation of numerous artificial measurements. Reducing the resolution of the VMM can therefore reduce the computational cost.

Comparison with Gaussian Processes The results of the extension estimation using a VMM and its adaptation are compared to a shape estimation based on Gaussian processes [77]. It is important to note that this shape estimation differs significantly from the GPRM approach proposed in section 4.4, where the shape remains elliptical. Results for the GPRM extension to VMMs are presented in the next section. In this section, the shape itself is modeled by a Gaussian process, resulting in a star-convex form.
The Gaussian processes algorithm employs a nonlinear motion model with coordinated turn. The kinematic state contains position (x, y), velocity v, orientation angle φ and turn rate ω. The state transition is given in [29, p. 2, Equation (4)]. This motion model has the advantage that the orientation φ is directly available. In the Gaussian processes algorithm, the measurements have to be transformed into the local target coordinate frame using the orientation [77, p. 4169]. A symmetric model is assumed. Therefore, the covariance function has a period of π. The kernel is defined as follows:

$$k(\theta, \theta') = \sigma_f^2 e^{-\frac{sin^2(|\theta-\theta'|)}{2l^2}}, \tag{6.3}$$

where θ represents the angle. The kernel function parameters are set as follows: $\sigma_f = 2$, $\sigma_r = 2$ and $l = \pi/4$ with a total of 100 radial function samples. An extended Kalman filter framework is used to accommodate the nonlinear model. Figure 30 compares the extension estimation results obtained with the VMM to those from the Gaussian process-based extension estimation.
Compared to the previous simulation study, the measurement noise has been reduced to $\sigma_x = \sigma_y = 0.2\,\text{m}$. With higher measurement noise (e.g. $\sigma_x = 0.5\,\text{m}$ as used before), the Gaussian processes algorithm may fail to converge as measurements are frequently assigned to wrong parts of the object's contour. Figure 31 shows the extension estimation at $k = 220$ for various levels of measurement noise.
In low-noise scenarios (see Figure 30 and Figure 31a), both algorithms achieve highly accurate results. The Gaussian processes-based approach tends to overestimates the width but lacks the VMM's prior information that the object is perfectly elliptical which also leads to a longer convergence time. However, if the shape is unknown and the measurement noise is low or moderate, Gaussian processes may be superior. By contrast, the VMM extension estimation remains largely unaffected by the increased measurement noise, as shown in Figure 31b. Table 3 provides numerical results with mean and root mean square error (RMSE) values evaluated over 100 Monte Carlo runs for both algorithms. Results where the Gaussian processes-based approach produces errors exceeding 5 m are omitted in the table due to divergence in some

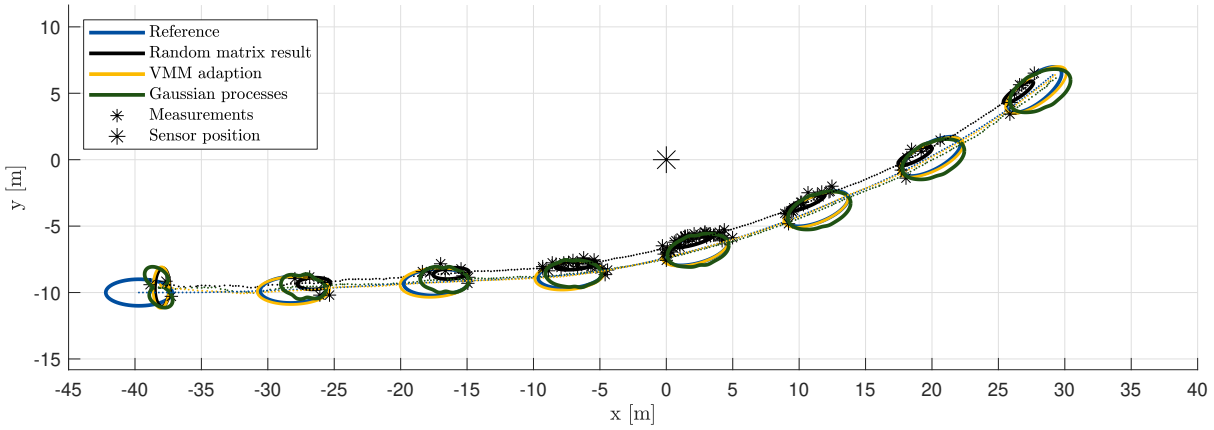
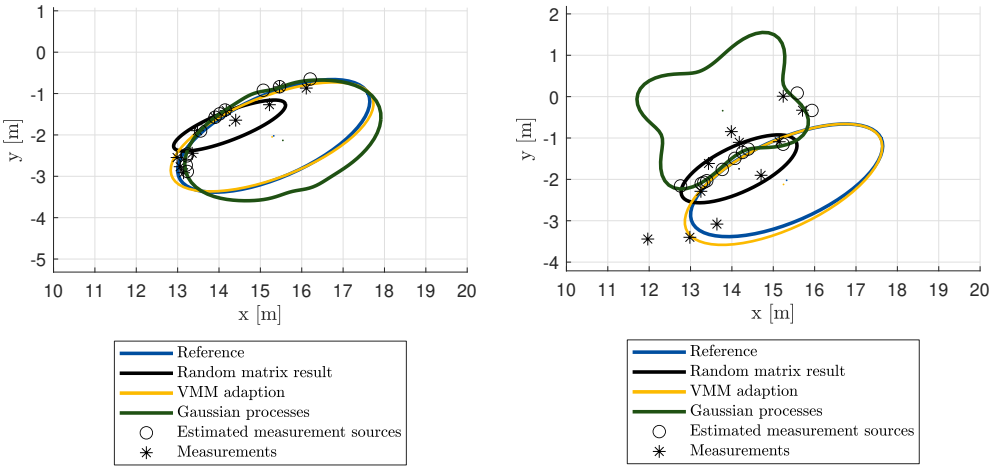


Figure 30: Tracking results for a Singer's model observed by a lidar sensor. Comparison with Gaussian Processes [PH1].



(a) $\sigma_x = 0.2$ m (b) $\sigma_x = 0.5$ m

Figure 31: Extension estimation at $k = 220$ for different measurement noise [PH1].

situations. The VMM approach generally outperforms Gaussian processes at the beginning and end of the scenario, where fewer measurements are available.
The Gaussian processes algorithm's average computational cost per time step is 36 ms, similar to that of the VMM with $4 - 5$ iterations (see Table 2).

Table 3: Mean and RMSE averaged over 100 Monte Carlo runs [PH1].

	Ref.	GP mean	VMM mean	GP RMSE	VMM RMSE
length	5 m	4.71 m	5.20 m	1.23 m	0.35 m
width	2 m	2.66 m	2.05 m	1.20 m	0.24 m

VMM adaptation with real lidar (Velodyne VLP-16) data In this scenario, a vessel on the Rhine river was observed using a Velodyne VLP-16 (Puck) Lidar. Figure 32 shows the observed vessel.

Figure 32: Observed vessel [PH1].

With real data, the challenges in extension estimation differ significantly from those in simulation studies. In the previous scenario, measurement noise was set to $\sigma_x = \sigma_y = 0.5\,\mathrm{m}$ whereas the VLP-16 offers a range accuracy of $\pm 0.03\,\mathrm{m}$. Measurement noise can easily be transformed in Cartesian coordinates using a linear approximation. Due to the high horizontal resolution ($0.1° - 0.4°$), numerous measurements are captured per time step when the target is nearby. However, the lower vertical resolution ($2.0°$) restricts measurement to a single layer, making 2D tracking methods sufficient.
A key challenge is that the assumptions of the measurement model are only partially met: The vessel's contour is not a perfect ellipse and measurements do not consistently appear only on the side of the vessel facing the sensor. Figure 33 shows the received Lidar measurements for every 15th time step.

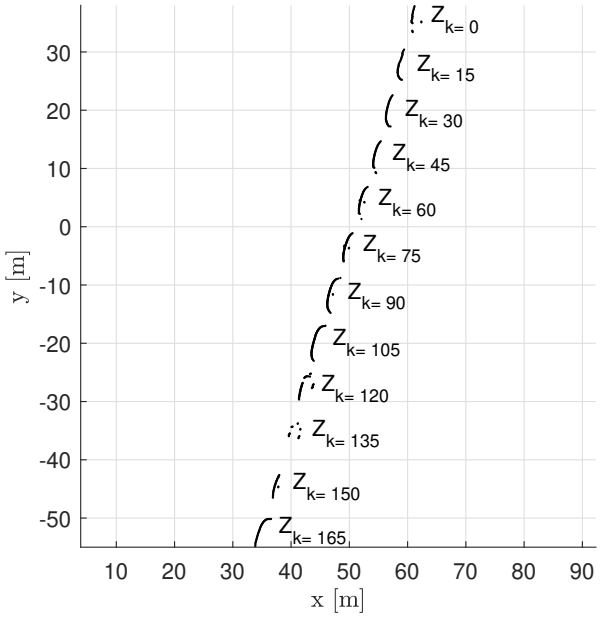
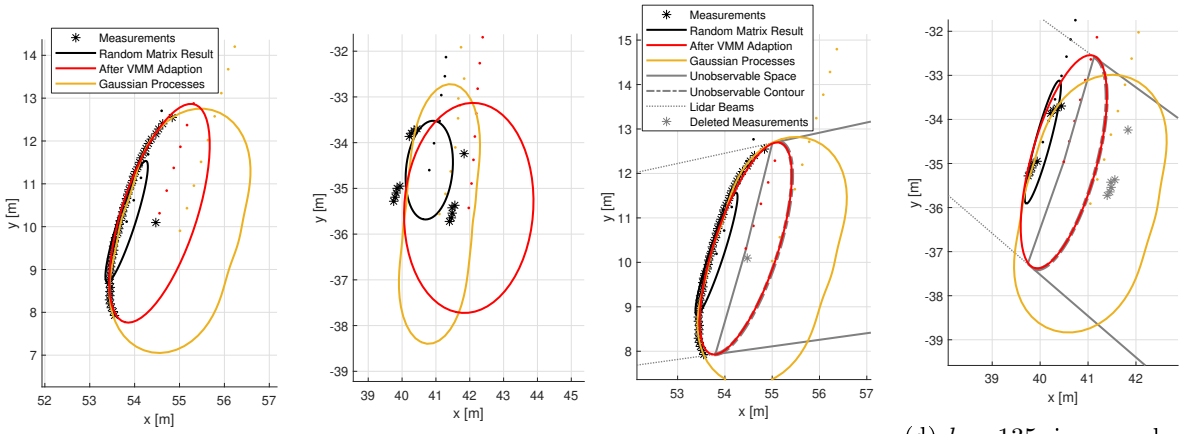


Figure 33: Lidar measurements shown for every 15 time steps [PH1].

From the beginning until approximately time step $k = 105$, the measurements align closely with the assumed model, with only a few outliers. Between $k \approx 120$ and $k \approx 135$, however, measurements also appear from the side facing away from the sensor. For the filter with VMM

adaptation, the same parameters as in the previous simulation study (Figure 25) were used, with the exception that the measurement noise was reduced to $\sigma_x = \sigma_y = 0.03\,\text{m}$. The tracking results at $k = 50$ and $k = 135$ are presented in Figure 34.



(a) $k = 50$ (b) $k = 135$ (c) $k = 50$, increased τ and gating (d) $k = 135$, increased τ and gating

Figure 34: Tracking results at selected time steps [PH1]

When measurements appear where expected, the extension can be estimated very precisely - see Figure 34a. However, if the measurements model does not align with the sensor's behavior, the extension estimation becomes significantly worse (see Figure 34b). The extension estimation using Gaussian Processes shows a different behavior: When measurements are only on the object's side facing the sensor, the object is estimated wider than with the VMM. When measurements appear on the other side, the Gaussian processes algorithm matches them to measurement sources at the backside. The width estimation gets smaller and close to its supposed reference value. There are several options to improve the VMM: The random matrix approach [30] uses a parameter τ which describes how fast the estimated covariance (and therefore the estimated extension) can change. Of course, the length and width of an object do not change, but a rotation of the covariance matrix can be caused by a turn of the object. Another option is to distinguish between measurements in the observable space (according to the assumed model) and those that seem to be outside. The observable space can be determined by the prediction of the VMM once it has converged. Measurements assigned to the non- observable part of the object's contour are deleted. The tracking results for this setting are shown in Figure 34c. The parameter τ has been increased from $\tau = 0.5$ up to $\tau = 50$ and only the measurements at the object's side facing towards the sensor are used. The boundary of the unobservable space is also shown. It is defined by the tangential lidar beams and the predicted center of the target. The estimation in Figure 34c has not changed since, in this time step, only a few measurements are affected by the gating. In Figure 34d, the estimation of extension and center has been improved. Without measurement gating, a length of $\approx 5.50\,\text{m}$ is estimated (see Figure 34a) and with measurement gating, a length of $\approx 5.00\,\text{m}$ (see Figure 34c). Both values seem to be reasonable for the observed vessel. The width is estimated as $\approx 2.00\,\text{m}$, resp. $\approx 1.50\,\text{m}$ with measurement gating. Assuming that the measurements shown in Figure 34d are from both sides of the object's contour, a width around 2.00 m seems reasonable. When there are measurements from both sides of the object, the extension estimation using Gaussian processes is close to this value (see Figure 34b). In this scenario, the object was mainly observed from the side. This made the length estimation easy and the width estimation challenging.

6.2. Extension estimation with the Gaussian processes regression model

In this section, the VMM is used to generate training data for a GPRM, which is then applied in live tracking to predict the extension parameters and the object position based on the random matrix result, following the approach outlined in section 4.4. All findings were previously published in the conference paper [PH4]. First, the GPRM approach is tested in the scenario presented in Figure 25 from the previous section, where a Singer's ODM moves in close proximity to lidar sensor. Following this, the real lidar data from Figure 33, captured by a Velodyne VLP-16 sensor observing a small vessel on the Rhine river, are evaluated. This allows to compare the results of both approaches.
Before tracking, the GPRM was trained with $1,000$ samples to predict the extension parameters $\underline{u}_{k,est}$ from the random matrix result $\underline{y}_k$ during live tracking. The object's orientation is not determined by the GPRM but from the Kalman filters kinematic. The tracking results are shown in Figure 35.

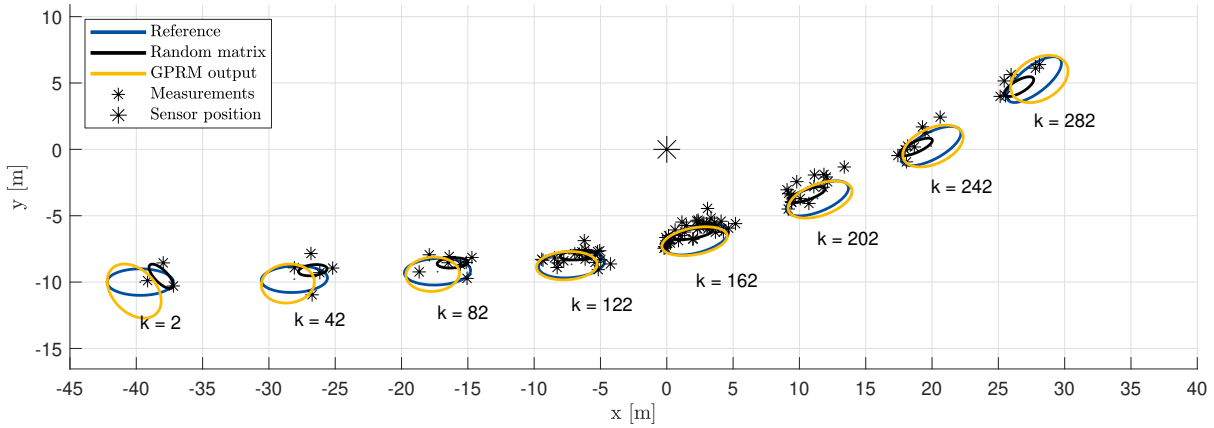
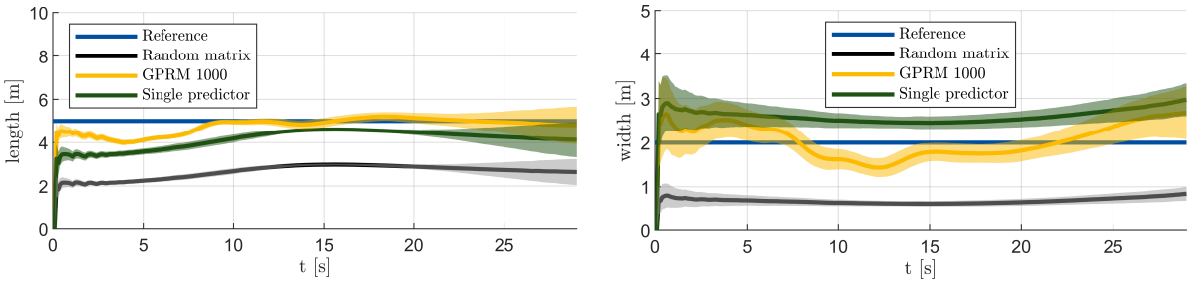


Figure 35: Tracking results of the random matrix algorithm and the GPRM approach for a constant acceleration model observed by a lidar sensor. The extension estimations and the measurements are shown every 40 time steps [PH4].

The results of the random matrix algorithm have not changed from those explained in the previous section, where the estimated position shifts toward the sensor and both length and width are underestimated. The GPRM functions as a corrective layer for the random matrix results, providing high accuracy, especially when the object is close to the sensor, at the midpoint of the scenario. However, at the beginning and end of the scenario, when measurements are sparse, the width estimation tends to lose accuracy.
To compare the multivariate GPRM results with those of single predictors for length and width, as well as with the VMM adaptation approach from the previous section, a Monte Carlo simulation is performed. The results are shown in Figure 36 and Table 4. In each run, a new trajectory and new measurements are generated, while the trained GPRMs and the initial states remain unchanged. The length is clearly underestimated by the single predictor while the width is overestimated. The single predictor takes the values of the random matrix approach (z_{el}, z_{ew}) and determines the corresponding l and w values (see Figure 21). As explained in chapter 4.4, this is not very accurate. However, good results can be obtained with the multivariate GPRM, especially for the length. At the beginning of the scenario, the length is slightly underestimated since it can not be observed well from this angle. The best precision is achieved when the object is close to the sensor and observed from the side. In the same section, however, the width is difficult to observe, which leads to an underestimation here. Table 4 shows the RMSE and computational time for GPRMs with different numbers of training data, as well as the comparison to the original approach via VMM adaptation[PH1].



(a) Length (b) Width

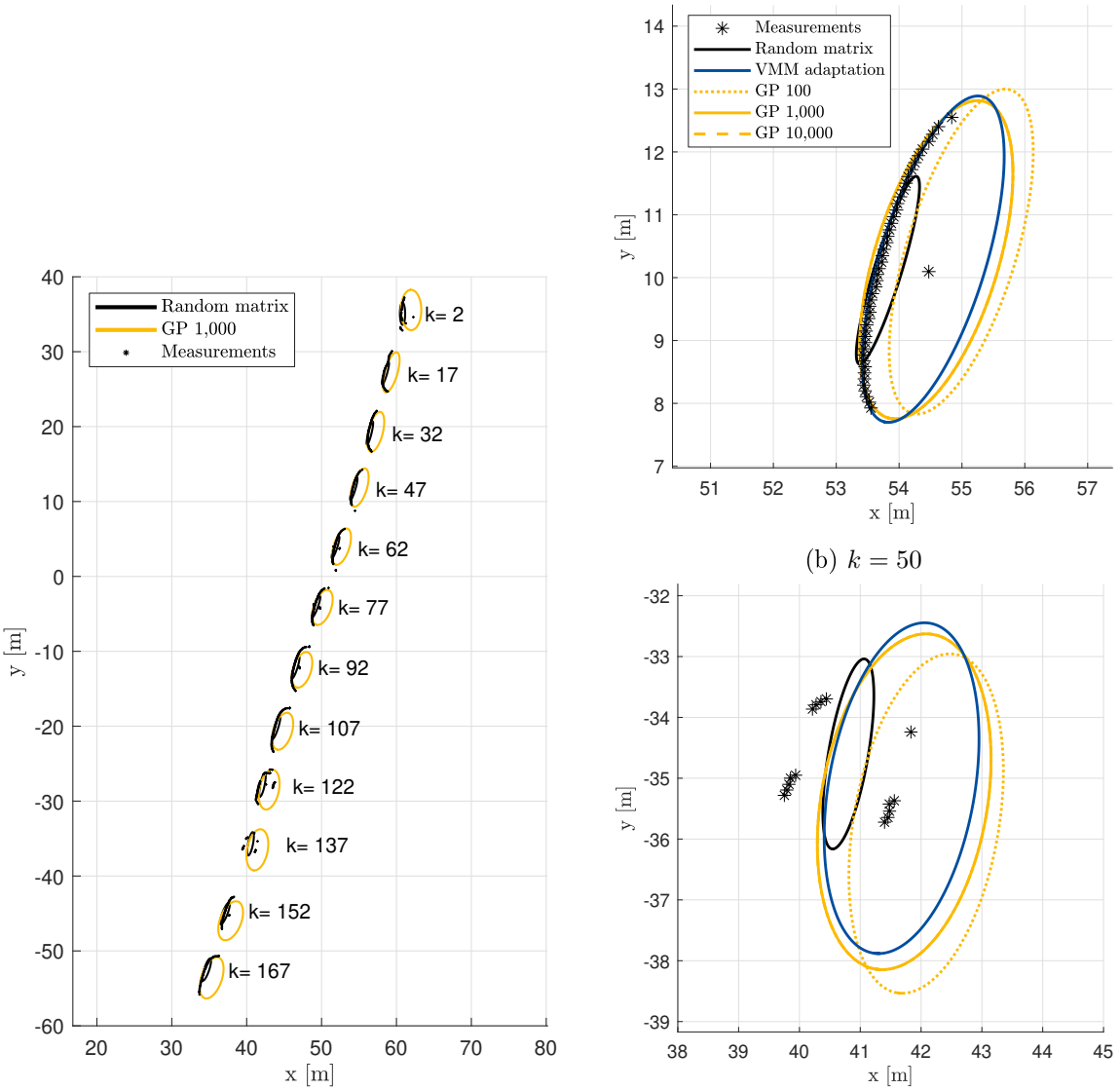
Figure 36: Extension estimation averaged over 100 MC runs [PH4].

Table 4: RMSE and calculation time for different GPRM settings [PH4].

	n	x[m]	y[m]	l[m]	w[m]	t[ms]
GPRM	100	0.72	0.88	0.51	0.68	0.15
GPRM	1,000	0.46	0.27	0.40	0.41	2.5
GPRM	5,000	0.49	0.40	0.48	0.47	3.9
[PH1]	n.a.	0.15	0.08	0.39	0.30	10.2

Not surprisingly, more training data improves the results but also means more computational effort. Compared to the adaptation, the performance is still slightly worse even with 5,000 training data, but the computational effort is significantly lower.

GPRM with real Lidar (Velodyne VLP-16) data The GPRM extension estimation is evaluated using real lidar data of a small vessel (see Figure 32) on the Rhine river, captured with a Velodyne VLP-16. Given that the VLP-16 has a range of 100 m, 1,000 pairs of training data are sampled with $r_{\max} = 100\,\text{m}$, $r_{\min} = 5\,\text{m}$, $l_{\min} = 2\,\text{m}$, $l_{\max} = 15\,\text{m}$. The linear Kalman filter with random matrix extension estimation is applied with the same settings as in the previous section, but with an increased value for τ set to 50. Figure 37 shows the tracking results of this scenario. For most time steps, the expansion estimate is reasonable, indicated by the measurements aligning with the estimated contour. One such time step is $k = 50$, where the estimation is straightforward because nearly all measurements were generated from the side of the vessel facing the sensor. This, therefore, aligns perfectly with the assumptions made in the VMM used to create the training data. However, at time step $k = 135$, this assumption is violated, as measurements are also received from the side of the vessel facing away from the sensor. The lidar beams hit the boat at a different angle or height and possibly pass the window at the rear of the boat as shown in Figure 32. Since this could not be included in the training data, there is a substantial discrepancy in the estimation of extension and position, as the measurements do not correspond to the object's contour.
At both time steps, the GPRM with only 100 pairs of training data has a larger mismatch (especially in position) compared to the others. This indicates that 100 is too little training data. The length and width estimation of the whole scenario are plotted in Figure 37.
It shows that there are no major differences between the GPRM with different training data and the VMM input adaptation for both extension parameters, but the length estimation of the GPRM converges faster. The lengths estimates are varying around a constant value, while the widths are increasing more and more. This is because the object is observed almost continuously from the side which makes length estimation easy and width estimation difficult. The averaged length and width estimation per time step are shown in Table 5. The Kalman filter with random matrices, is very efficient with 0.3 ms/time step, but cannot provide a precise expansion estimate on its own. The determined values for length and width seem plausible, even if no reference for the ship is available. In addition, the vessels shape is not perfectly elliptical.



(a) all time steps

(b) $k = 50$

(c) $k = 135$

Figure 37: Tracking results of the Kalman Filter with Random Matrix covariance estimation followed by a Gaussian processes regression. The sensor is located at $(0, 0)$ [PH4].

Table 5: Averaged extension estimation for the Kalman Filter with Random Matrices, GPRM regression and the VMM input adaptation from [PH1] with 2 iterations per time step.

	KF+RM	GP 100	GP 1,000	GP 10,000	[PH1]
Length [m]	3.1	5.4	5.3	5.4	5.2
Width [m]	0.5	2.0	2.3	2.3	2.0
calc. time [ms]	0.3	1.2	1.4	1.5	2.2

6.3. Multi-extended object tracking and occlusions

In this section, the VMM with the integral-based adaptation law is evaluated in the context of multi-extended object tracking. The implemented framework is based on the methodology in section 4.3, initially presented in the conference paper [PH3]. This results section focuses on occlusions, as they can be effectively modeled within the ray tracing methods presented in section 3. The first scenario, though focused on a single object, is included to illustrate a temporary occlusion as the object moves behind a static obstacle, rather than involving occlusions caused by other objects. In the second scenario, however, multiple objects are considered, with frequent occlusions occurring throughout. The results of both scenarios were as well published in the conference paper [PH3].
For the first scenario, a constant velocity model is used for the object's kinematic. The lidar sensor is simulated with a resolution of 1° and measurements are affected by white Gaussian noise with $\sigma_x = \sigma_y = 0.5\,\mathrm{m}$. To consider the wall, the occlusion is modeled as described in section 3.7. The Kalman filter with the random matrix algorithm is used with the same parameters as in previous sections, followed by the VMM adaptation with an initial gain of $K_i = 0.1$ and an increased resolution for the virtual sensor of 0.01°. The results are shown in Figure 38.

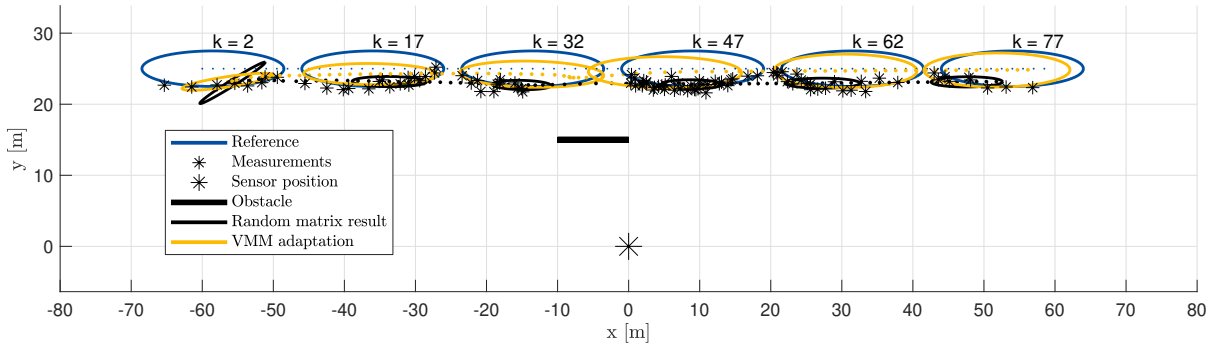


Figure 38: Single object tracking with occlusions: Object passing behind an obstacle observed by a lidar sensor. Reference ellipse, Measurements and results visualized every 15 time steps [PH3].

At the beginning and the end of the scenario, the object is fully visible and the results are in general the same that were observed in section 6.1, where an elliptical object passed nearby a lidar sensor without any occlusion. At time step $k = 32$, the occlusion's effect becomes evident: although measurements are generated only from the object's left side, the object's extent is still accurately determined, as the obstacle-induced occlusion is accounted for by the VMM. At time step $k = 47$, however, a limitation of the method emerges as a leftward bias in the estimation. When the object passes behind the obstacle, it seems that the center of gravity of the measurements "decelerates" (see Figure 39). Once measurements reappear on the other side of the obstacle, the estimated velocity increases, causing the estimate to lag behind the reference for a few time steps. Since the total distance moved by the center of gravity in Figure 38 is shorter than the distance moved by the object, there is generally a bias in the velocity estimate in Figure 39.

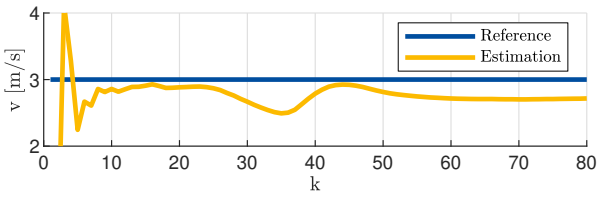


Figure 39: Velocity estimation

In the second scenario, four extended objects of varying sizes are simulated, as shown in Figure 40. Measurements are generated using ray tracing, with occlusions from other objects accounted for as outlined in section 3.7. Clustering of measurements is performed using the DBSCAN algorithm [87], with parameters MinPts=1, defining the minimum required measurements within a cluster, and eps=15 m, specifying the minimum separation between clusters. Following clustering, the MEOT framework for VMM adaptation is applied as described in section 4.3, with an individual Kalman filter and random matrix extension algorithm assigned to each object, followed by the VMM adaptation. The results are presented in Figure 40.

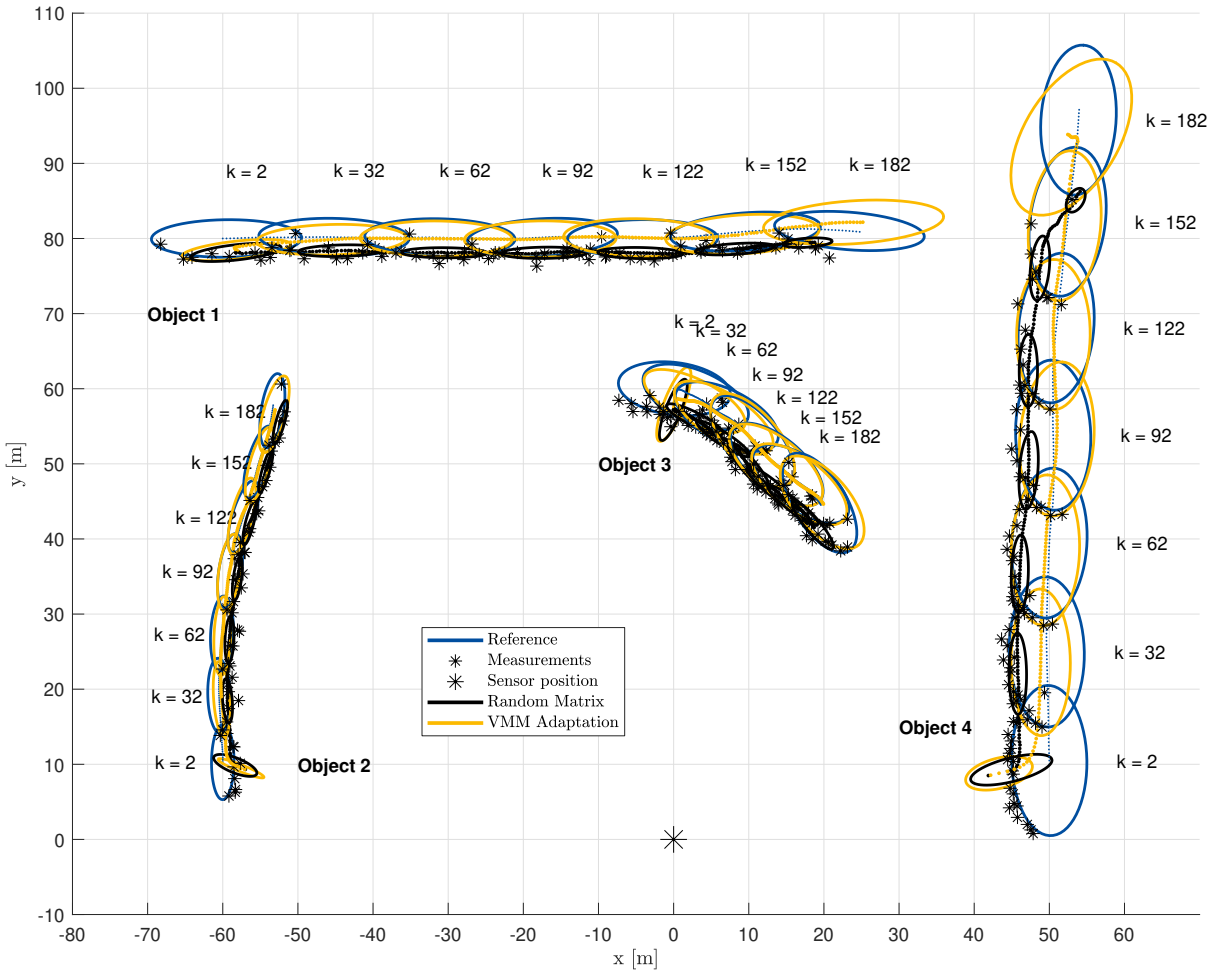


Figure 40: Multi-extended object tracking results with occlusions. Reference, extension estimation and measurements are visualized every 30 time steps.

For most time steps, no occlusions occur, allowing the extents of all four objects to be estimated. The results are basically the same as in previous sections with single elliptical objects, since object's do not influence each other. However, at time step $k = 182$, object 3 is in such a position that it partially covers objects 2 and 4. While there are still sufficient measurements for an accurate adaptation of object 2, this is not the case for object 4. In this situation, a *miss detection hypothesis* could be introduced as an enhancement, where the adaptation would be paused when measurements are insufficient respectively not available at all.

6.4. Comparison with other EOT methods

In previous sections, the VMM adaptation and the GPRM approach have been compared against each other and to a Gaussian process-based shape estimation. Although some comparisons were also made with the original random matrix method, these are less informative since the random matrix approach does not account for the ray tracing measurement model.
In their work [C4, C5] on EOT using superellipses, Baur et al. conducted extensive simulation studies and evaluations on real lidar data from automotive and maritime scenarios. They compared their superellipse model with multiple other EOT approaches, including the multiplicative error model extended Kalman filter (MEM-EKF*) [73], elliptical and star convex (Fourier) RHM [71, 72, 94], Gaussian processes-based extent estimation [77] and the VMM method. For these studies, a VMM Matlab implementation was provided, in which the VMM generates artificial measurements from the contour using ray tracing with a resolution of 1° and artificial measurements from the object's interior with a density of $2/\mathrm{m}^2$. These results reveal additional strengths and weaknesses of the VMM method; thus, the most significant findings are summarized and reevaluated here with a focus on VMM. All simulations in [C4, C5] used a coordinated turn model [29], with computational effort and intersection over union (IoU) [95] as the main evaluation criteria.
In the first simulation study, a reference trajectory around a lidar sensor is generated, with the object's shape modeled using a CAD model of the research vessel Solgenia [C4, 96]. A simulated lidar sensor captures surface points and measurement noise is added with a standard deviation of $\sigma_x = 0.05\,\mathrm{m}$. Further details about this scenario can be found in [C4, 96]. The results for the simulated maritime scenario from Baur et al. are illustrated in Figure 41.

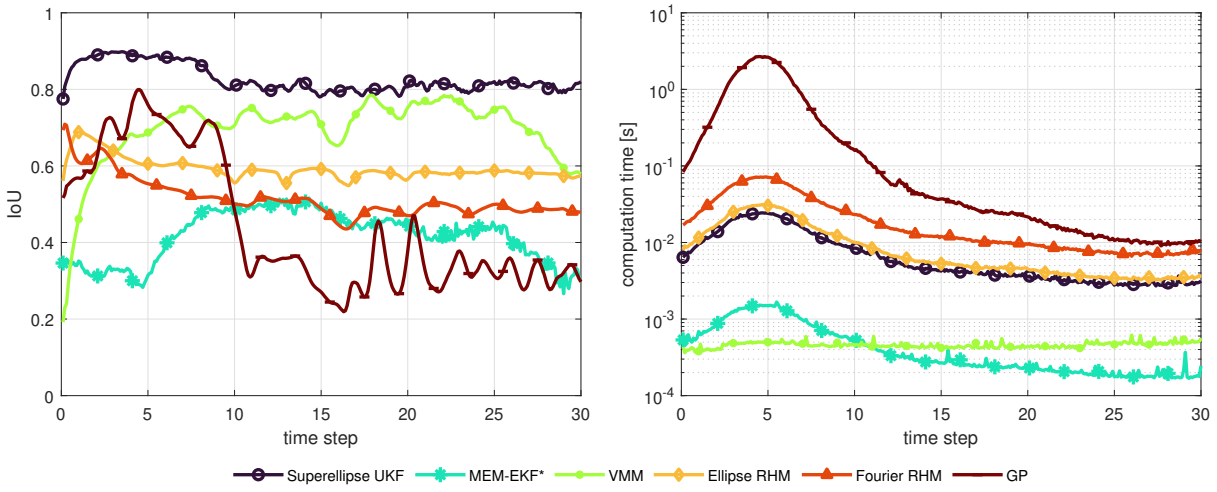
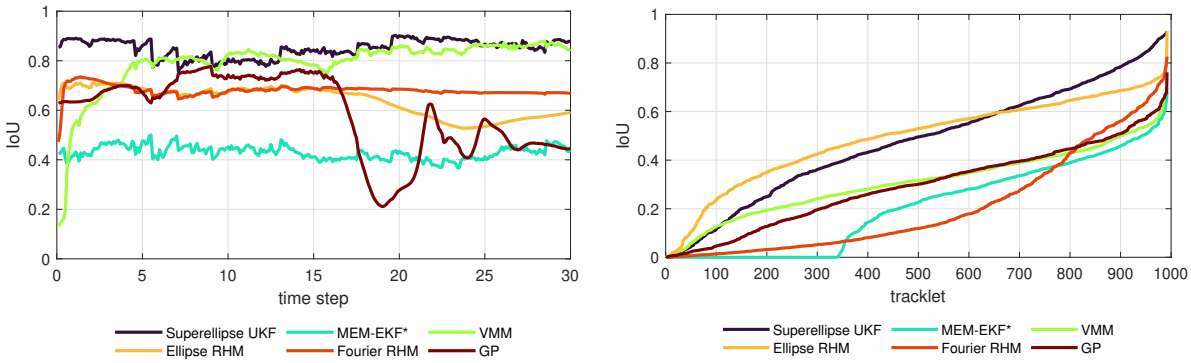


Figure 41: Intersection over Union and computational effort of the comparison study [C4, p. 6]. Other parameter settings for superellipses, elliptical and Fourier RHMs and Gaussian processes are presented in [C4, p. 6].

In terms of IoU, the VMM algorithm performs well, ranking in the upper range, although it falls slightly short compared to the superellipses, which offer greater shape flexibility: The Solgenia's shape can be reasonably approximated by an ellipse in 2D, but superellipses with optimal parameterization provide a closer fit. When considering computational efficiency, the VMM algorithm is among the best, together with the MEM-EKF*, although the latter performs poorly in accuracy. Another key advantage of the VMM algorithm is its consistent computational demand throughout the scenario. At time step $k = 5$, the object is closest and generates numerous measurements, the computational effort rises for all EOT methods except VMM. The primary reason for this efficiency is that the random matrix method, a core component of the VMM, requires only the prediction and update of mean and covariance, which themselves are independent of the number of measurements. However, calculating the mean and covariance of the current measurement set,

which is necessary for the update step, depends slightly on the number of measurements. When generating artificial measurements with the VMM, lidar beams are considered in all directions and intersections are calculated regardless, as it is unkown in advance if they exist or not. If a solution and therefore a measurement source are found, the computational effort remains almost stable. Furthermore, all other steps, such as comparing with the random matrix results and performing the adaptation, are completely independent of the number of measurements. Other EOT methods, do the update step individually for each measurement and have therefore an almost linear relation between number of measurements and computational effort.
Besides the simulation studies, Baur et al. also compared EOT methods in a maritime scenario and on an automotive data set [C5]. For the 2D real-world maritime scenario, the Solgenia vessel was tracked from a stationary lidar sensor positioned at the river bank, with an RTK-GPS and CAD model as reference [C5]. For the automotive scenarios, the KITTI data set [4] was used. Further details are provided in [C5]. Figure 42 shows the IoUs for each EOT method in both real-world applications. In the maritime scenario, IoUs are shown for each time step, while for the automotive data set, IoUs are displayed for every tracklet and sorted to improve visualization.



(a) Maritime (b) Automotive

Figure 42: Extension estimation results from real-world applications [C5]

In the maritime scenario, the VMM performs very well, being only slightly behind the superellipses. However, on the automotive data set, the VMM encounters some challenges but still outperforms both MEM-EKF* and the star convex RHM. Since many targets in the automotive data set are rectangular rather than elliptical, this could account for the superior performance of superellipses. However, the ellipse RHM scores surprisingly well, suggesting that VMM's limitations may not stem solely from wrong shape assumptions. On the other hand, it is notable that the ellipse RHM did not perform as well in the maritime scenario. To further improve the VMM method, particularly for automotive scenarios, greater flexibility is required. This could be done by incorporating additional shapes, such as rectangles and triangles, as introduced in section 3, which will be evaluated in the following section. Another option is to make the VMM ray tracing resolution and spatial measurement density adaptive. This could be beneficial, especially if the density varies across individual tracklets within the automotive data set, caused by different height and geometry.

6.5. Non-elliptical shapes and shape classification in 2D and 3D

In previous results sections, the object's shape was limited to 2D ellipses. In this section, the parallel application of multiple VMMs to represent a range of 2D and 3D shape models, as proposed in section 5, is evaluated. This also includes the shape classification method, based on either histograms or Chamfer distances, to select the best-fitting shape. First, this approach is tested in a 2D simulation study, published in the conference paper [PH5]. Following this, a 3D simulation study and experiments with real lidar data from maritime scenarios are presented. These results were published in [PH6].

Simulation studies for 2D EOT and Shape Classification with Histograms For the 2D simulation study with shape classification, the same constant acceleration model is applied, with state vector $\underline{x} = (x, y, v_x, v_y, a_x, a_y)^\top$ and the transition affected by white Gaussian process noise of $\sigma_{wx} = \sigma_{wy} = 0.5\mathrm{m/s}^3$. The sampling period is set to $T = 0.1\,\mathrm{s}$. To keep the object near the sensor for an extended period and ensure it passes close to the sensor multiple times, the trajectory is not determined solely by process noise but is also directed by navigation commands. These commands specify a certain acceleration, subsequently affected by the process noise. The navigation commands are listed in Table 6, resulting in the trajectory shown in Figure 43.

Table 6: Navigation commands for the desired trajectory [PH5].

k	50	140	150	180	200	320	350	450
a_x [m/s^2]	2	0	0	0	-1	-1.5	0	-1
a_y [m/s^2]	-1.5	2	0	2	-2	2	0	0

Because acceleration affects position and velocity with a delay, the direction changes observed in Figure 43 lag behind the control commands listed in Table 6. Additionally, these control commands introduce unexpected disturbances, providing an extra challenge for the filtering approach.

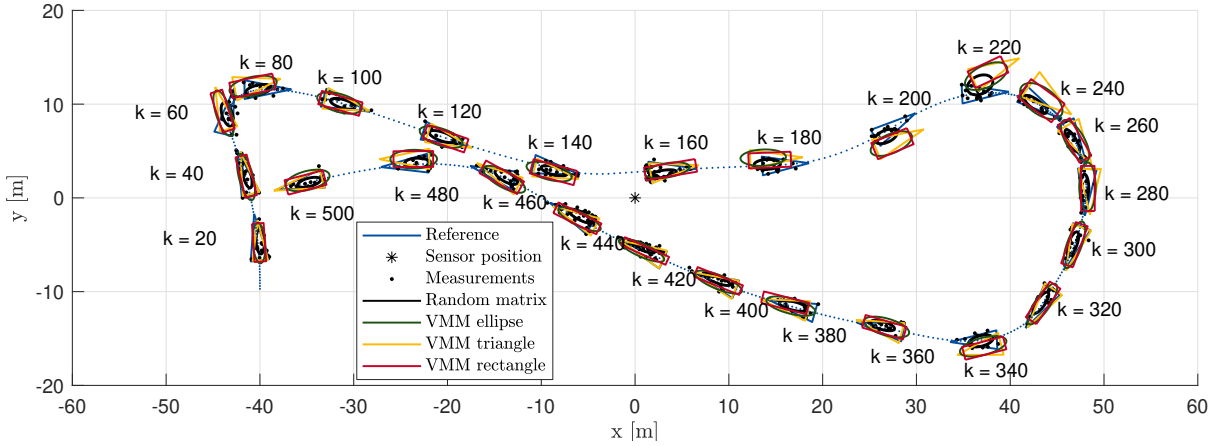


Figure 43: Tracking results for a controlled constant acceleration model with triangular shape observed by a lidar sensor. Extension estimation with three VMMs (Ellipse, Triangle, Rectangle). The extension estimations and the measurements are shown every 20 time steps [PH5].

The reference object has a triangular shape with length $l = 5\,\mathrm{m}$ and width $w = 2\,\mathrm{m}$. The sensor is located at $(0, 0)$ and generates contour measurements with a resolution of 1° as well as measurements uniformly distributed over the object's extent with a density of $2/\mathrm{m}^2$. All measurements are affected by white Gaussian noise with $\sigma_x = \sigma_y = 0.2\mathrm{m}$. Since both the true shape of the object, which could be an ellipse, triangle, or rectangle, and its size in terms of length and width are unknown, three VMMs run in parallel to perform simultaneous extension

estimation. Each VMM follows its own adaptation loop, based on the framework presented in section 5. The VMMs operate under known sensor specifications: the resolution of 1° and the density of $2/\mathrm{m}^2$ are known. Tracking results from the linear Kalman filter with random matrix extension estimation, as well as from the three VMMs, are shown in Figure 43. The triangular VMM aligns closely with the reference shape at most time steps, while the other VMMs shapes also produce reasonable estimates of the object's expansion. One such example is at time step $k = 160$, shown in Figure 44.

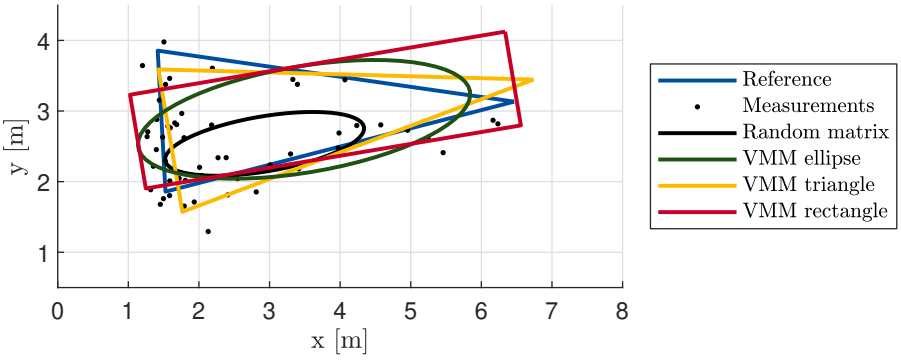
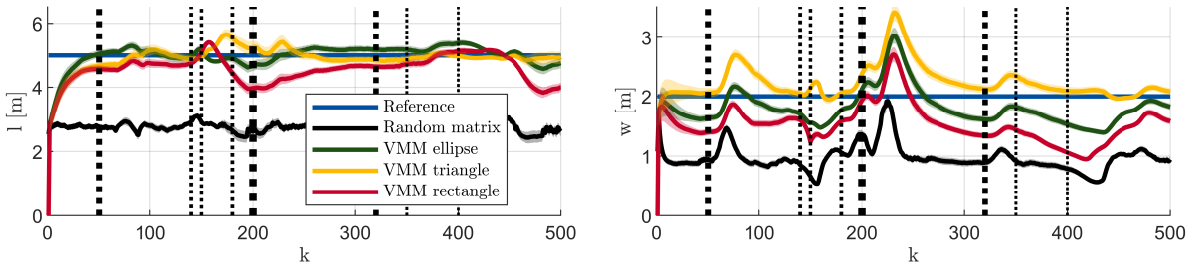


Figure 44: Extension estimation results at $k = 160$ [PH5]

The VMM builds on the random matrix estimate: due to unevenly distributed contour measurements, the random matrix ellipse often shifts slightly toward the sensor, an effect consistently observed in all result sections. This shift is later corrected by the VMMs. Temporary divergence in the estimate may only occur immediately after turning maneuvers, caused by the Kalman filter requiring time to respond to the unexpected disturbance. It is important to note that the orientation angle remains consistent across all estimates, as it is derived directly from the Kalman filter's kinematic model using $\varphi = \mathrm{atan2}(v_y, v_x)$.
To obtain more conclusive results, a Monte Carlo simulation with 500 runs was conducted using the trajectory shown in Figure 43. For each run, measurements were regenerated, while the process noise was sampled only once to maintain a consistent trajectory. Figure 45 shows the mean and standard deviation of the extension estimates.



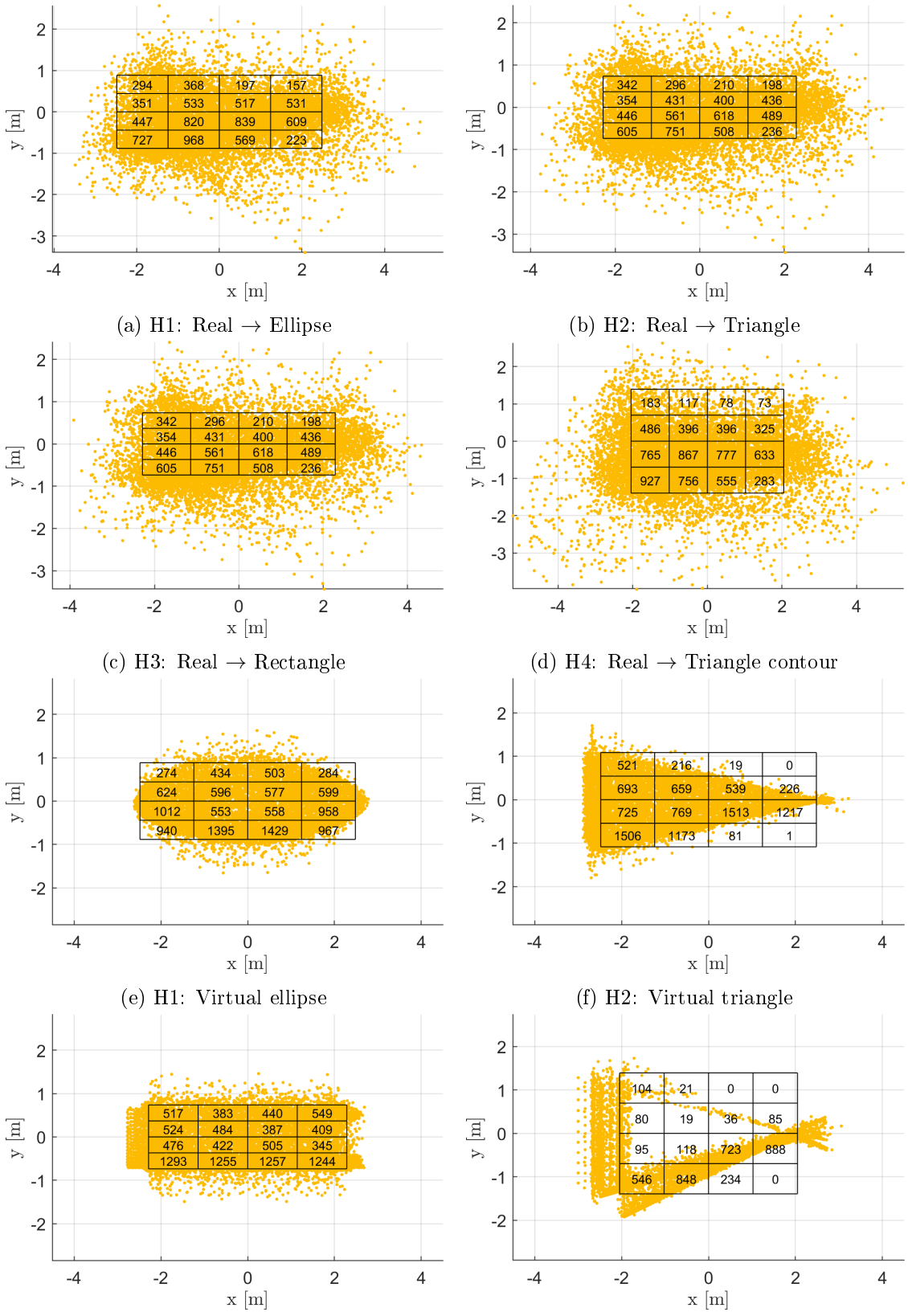
(a) Length (b) Width

Figure 45: Extension estimation. Averaged over 500 MC runs with the same trajectory but different measurements. The dotted lines show the navigation commands and their thicknesses correspond to the change in acceleration [PH5].

Up to time step $k = 50$, where no distractions from navigation commands are present, all shape models produce an accurate length estimate, though only the correct triangle model captures the width accurately. It should be noted that, due to the orthogonal observation angle in this section, the length is generally easier to estimate than the width. For later time steps, width estimates increase with a slight delay after each navigation command, and this increase corresponds to the intensity of the kinematic change. Subsequently, the width estimation consistently converges back to the true value, while alternative shape models tend to underestimate the width in sta-

tionary periods. The low standard deviation observed across the Monte Carlo runs suggests that the main sources of error are due to the navigation commands rather than measurement noise.

Although all VMM shape models provide reasonable results, it is unquestionable, that the best performance is expected from the right shape models. Likewise, for a real target that does not precisely conform to one of these basic geometric shapes, the VMM that best approximates the target shape is likely to yield the most accurate results. Consequently, implementing a shape classification method offers a distinct advantage. This is now evaluated using the histogram-based approach introduced in section 5.1.
In addition to the VMM adaptation for elliptical, triangular, and rectangular shapes with known sensor specifications, a triangular VMM adaptation is also considered, where mainly contour measurements are generated. This is done to investigate whether the same approach can be used not to classify the shape, but rather the measurement likelihood, such as distinguishing between contour measurements, interior measurements, or a mixture of both. In summary, four hypotheses are considered: H1 for an elliptical shape, H2 for a triangular shape, H3 for a rectangular shape, and H4 for a triangular shape with a spatial density in the VMM reduced to 1/10. For each hypothesis, the real measurements are compared with the virtual measurements generated by the corresponding VMM during the last adaptation of each time step. This comparison requires alignment in the same coordinate system; therefore, to evaluate H1, for example, both real and virtual measurements are transformed into the target coordinate system of the estimated elliptical object. This is illustrated for all hypotheses in Figure 46. Next, the measurements are divided into 16 segments of equal size, with the estimated length and width setting the segment boundaries. For each segment, the measurements are counted. Then, the number of real measurements in each segment is compared with the number of artificial measurements in the corresponding segment using a normalized RMSE across all segments.



(a) H1: Real → Ellipse

(b) H2: Real → Triangle

(c) H3: Real → Rectangle

(d) H4: Real → Triangle contour

(e) H1: Virtual ellipse

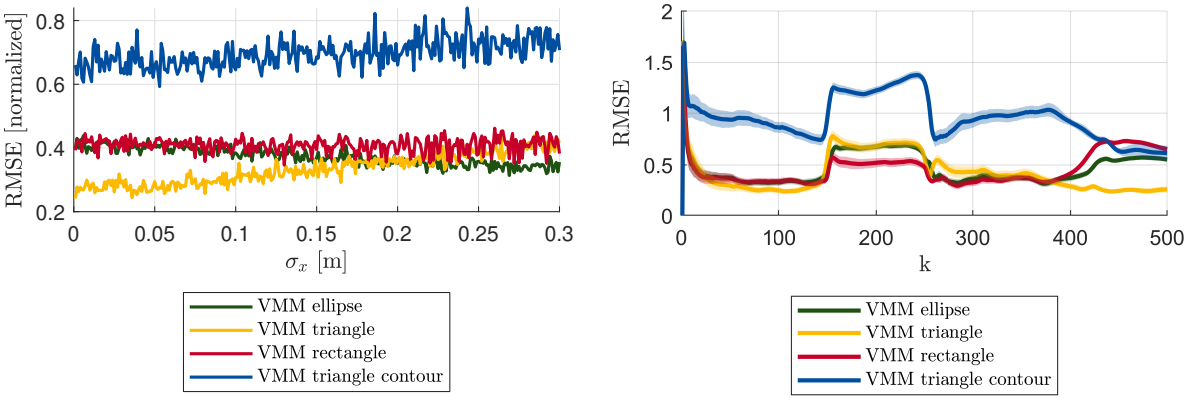
(f) H2: Virtual triangle

(g) H3: Virtual rectangle

(h) H4: Virtual triangle contour

Figure 46: Histogram-based shape classification. Real and virtual measurements transformed in the target coordinate system of the estimation [PH5].

The scenario shown in Figure 43 is repeated with 300 different noise realizations, and measurements are collected throughout the entire scenario. Figure 47a presents the RMSE results from the histogram-based comparison between real and artificial measurements, depending on the noise level.



(a) RMSE over measurement noise

(b) Sliding Window with $L = 100$

Figure 47: RMSE results for shape classification [PH5].

Up to a noise level of $\sigma_x = 0.15\,\text{m}$, the true triangular shape produces the lowest RMSE and is clearly identifiable. At higher noise levels, however, the elliptical shape becomes more fitting, since Gaussian distributed measurements imitate an elliptical shape. Across all noise values, the VMM model that primarily relies on contour measurements yields the highest RMSE, demonstrating that this approach can also be used effectively for likelihood classification.
Figure 47b presents results from a Monte Carlo simulation using the sliding window approach. In this setup, the RMSE at each time step depends on measurements received over the previous 100 time steps. This simulation reveals the sections of the trajectory where the triangle shape is classified correctly. The triangle is accurately identified only between $k = 50$ and $k = 150$ and from $k = 400$ until the end of the scenario. When examining these segments in Figure 43, it becomes evident that only in these intervals is the triangle's tip directed toward the sensor. It is unsurprising that the triangle cannot be distinguished from a rectangle when observed from the rear. Here, spatial measurements also appear to offer no additional classification advantage.

Simulation studies for 3D EOT and Shape Classification with Chamfer distances The first 3D shape investigated is the ellipsoid. In the following scenario, an ellipsoidal object is tracked, with kinematic transitions modeled by a 3D constant acceleration (CA) model and a state vector $\underline{x} = (x, y, z, v_x, v_y, v_z, a_x, a_y, a_z)^\top$. The model is influenced by white Gaussian process noise with $\sigma_{wx} = \sigma_{wy} = \sigma_{wz} = 0.5\ \mathrm{m/s^4}$, and a sampling period of $T = 0.1\,\mathrm{s}$. Measurements were generated through 3D ray tracing, as described in section 3 for the ellipsoid, with a sensor resolution of 1° in both vertical and horizontal directions. All measurements were also affected by white Gaussian measurement noise with $\sigma_x = \sigma_y = \sigma_z = 0.5\,\mathrm{m}$. Figure 48 shows the reference trajectory, the measurements and the results from the random matrix algorithm and from the VMM algorithm.

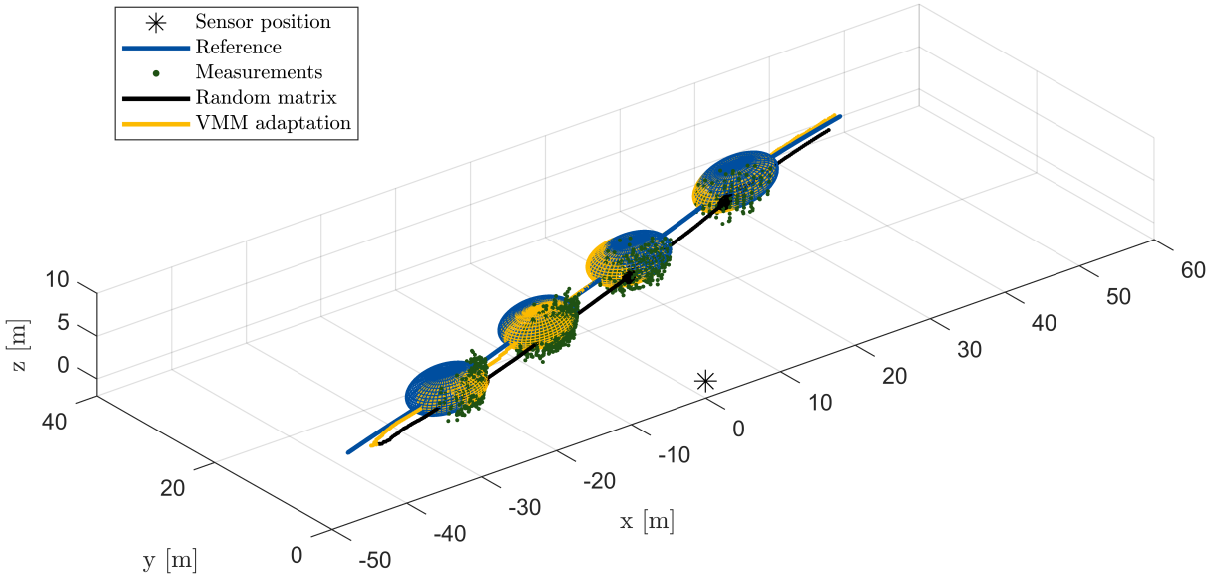


Figure 48: Single run. Extension estimation and measurement shown every 60 time steps [PH6].

The trajectory estimated by the linear Kalman filter with random matrix extension estimation shows a noticeable shift towards the sensor, an observation consistent with previous 2D simulation studies. Additionally, the 1σ ellipsoid is smaller than the actual object's extent and is therefore barely visible, as it lies within the reference ellipsoid. In contrast, the VMM adaptation effectively compensates for the trajectory shift with only two iterations per time step. The estimated ellipsoid size closely matches the reference ellipsoid. Notably, the enhanced cost function to prevent extension growth due to unobservability was not required in this scenario, as the VMM adaptation remained stable on its own.
To evaluate robustness and obtain more conclusive results, a Monte Carlo simulation with 100 runs was conducted, with process noise and measurement noise independently sampled for each run. Figure 49 and Table 7 show the mean and standard deviation of the extension estimation from the Monte Carlo simulation.

Table 7: Mean and std. deviation from 100 MC runs. Ellipsoidal reference object [PH6].

	Reference	Ellipsoid	Cone	Cuboid
length [m]	11	11.1 ± 1.4	12.9 ± 2.6	7.3 ± 0.9
width [m]	7	7.1 ± 1.4	6.2 ± 2.3	4.2 ± 1.8
height [m]	4	4.2 ± 0.4	4.7 ± 0.9	3.3 ± 0.3

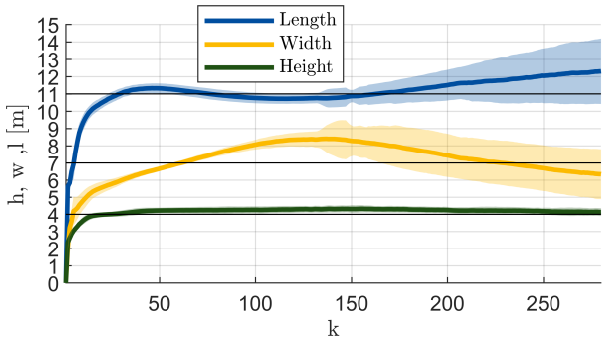


Figure 49: Extension estimation averaged over 100 MC runs [PH6]

Height estimates were the most accurate, benefiting from a favorable angle between the sensor and the object. Toward the end of the scenario, estimating the object's length became more difficult as it was viewed exclusively from behind. Similarly, width estimation proved challenging when the object was close but only visible from a side angle.
Assuming that the ellipsoidal shape is initially unknown and needs to be identified, the cone and cuboid VMMs are applied simultaneously. Their results, shown in Table 7, are clearly less accurate, reflecting the consequences of assuming an incorrect shape. However, the generated measurements from each VMM, $\tilde{\boldsymbol{Z}}_k$, can be compared with the actual measurements, $\boldsymbol{Z}_k$, using Chamfer distances to help determine the correct shape. Figure 50 presents the Chamfer distances for a sliding window of length $L = 5$, averaged over 100 Monte Carlo runs. On average,

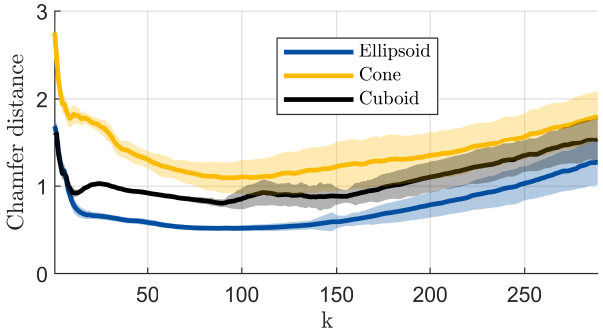
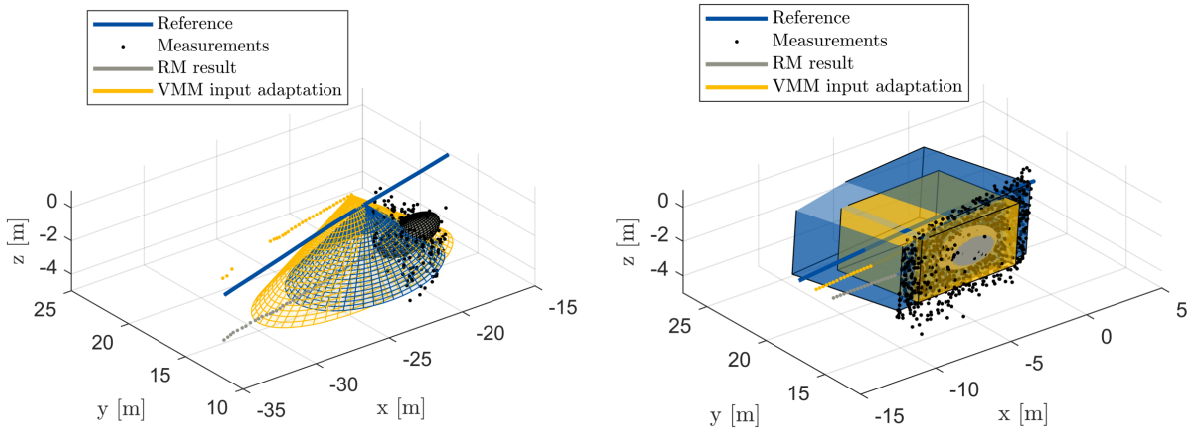


Figure 50: Chamfer distances for three VMMs. Monte Carlo results [PH6].

the Chamfer distance between the ellipsoid's virtual measurements and the real measurements is the lowest, confirming accurate shape classification. The 1σ ranges show consistent correct classification overall, although some overlap occurs toward the end of the scenario, indicating occasional misclassifications in a few Monte Carlo runs. These misclassifications, however, were infrequent: the ellipsoid was correctly identified in 97.9% of all time steps, while the cuboid achieved the lowest Chamfer distance in 1.9% of instances, and the cone in only 0.2%
To determine if objects with different shapes could be tracked with comparable accuracy and effectively identified, the same scenario was simulated with a cone and a cuboid as reference objects. Figure 51 shows the results at time step $k = 60$. Overall, the cone yielded tracking results comparable to those for the ellipsoid. In contrast, the cuboid showed significantly degraded tracking performance at certain time steps, with notable underestimation of its size, especially when only a single face was visible, resulting in observability challenges (see, e.g., Figure 51b). Classification results for all shapes are summarized in Table 8, based on 100 Monte Carlo runs per shape. While the ellipsoid and cone were consistently classified accurately, the cuboid was frequently misclassified as an ellipsoid.



(a) Cone

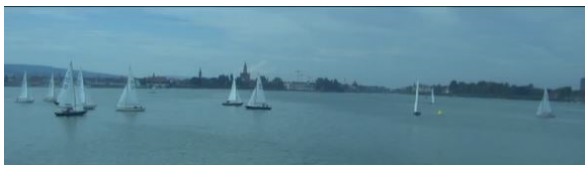
(b) Cuboid

Figure 51: Tracking results at $k = 60$ [PH6].

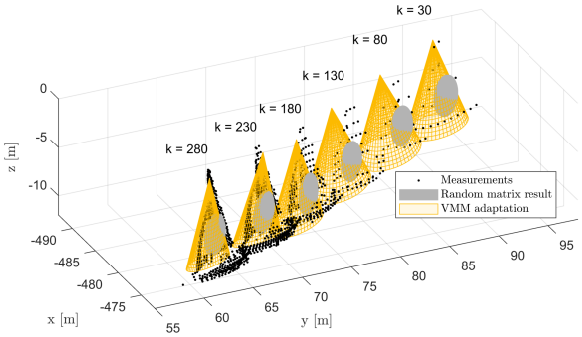
Table 8: Classification results from 100 Monte Carlo runs per shape.

	classified as		
Reference	Ellipsoid	Cone	Cuboid
Ellipsoid	97.9%	0.2%	1.9%
Cone	2.9%	94.7%	2.4%
Cuboid	46.6%	0.1%	53.3%

Real-world experiments with a sailing boat To evaluate the EOT approach in a real-world setting, a scenario with sailing boats on Lake Constance is considered. The camera image in Figure 52a shows an overview of the scenario. Measurements were captured by a Velodyne Alpha Prime (VLP-128) lidar mounted on a catamaran traveling approximately 32 km/h from Friedrichshafen to Constance. Differential global positioning system (GPS) provided position and yaw data, while an inertial measurement unit (IMU) measured roll and pitch to account for the catamaran's ego motion. Figure 52b shows the lidar measurements, random matrix result, and extension estimation after VMM adaptation for a single sailing boat. An elliptical cone was identified by the VMM classification with Chamfer distances as the best-fitting shape.



(a) Camera image.

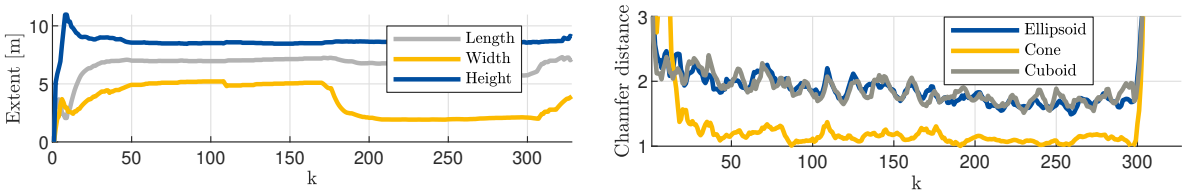


(b) Measurements and tracking results.

Figure 52: Real world experiments with a sailing boat [PH6].

At the start of the scenario, the sailing boat is approximately 240 meters away, resulting in a low number of measurements. As the catamaran gets closer, the measurement count steadily increases, reaching its peak when the distance narrows to about 70 meters at the end of the scenario. Around time step $k = 300$, the measurement count quickly drops to zero as the object moves into a blind spot, where the lidar becomes obstructed by parts of the catamaran's roof.

As shown in Figure 52b, the VMM adaptation aligns the cone reasonably well with the measurements, though not perfectly, as some measurements fall distinctly outside the cone. Figure 53a shows the estimated extension parameters over the time steps.



(a) Extension estimation (Cone-VMM).

(b) Chamfer distances

Figure 53: Tracking results from the sailing boat [PH6].

The length and height estimates appear plausible, remaining nearly constant throughout the scenario. However, estimating width poses several challenges due to various factors: first, the wind positions the sail at the back of the boat, whereas ray tracing assumes measurements come from the front, leading to a mismatch in modeling. Second, while the boat tilts slightly, ignoring roll and pitch results in an estimated cone that remains upright, misrepresenting the boat's orientation. Third, since the boat is consistently viewed from the side, estimating width becomes inherently more challenging than estimating length or height.
The classification results, based on Chamfer distances, are shown in Figure 53b. With the exception of a few initial time steps (when the Kalman filter has not fully converged) and the final steps, the cone consistently achieves the lowest Chamfer distance. This outcome is intuitive, as a cone visually appears to be the best fit for the shape of a sailing boat.

Real-world experiments with a motor boat In the second real-world scenario, lidar data from the motorboat "Solgenia" are analyzed. For this scenario, both a CAD model and reference extension parameters are available. Figure 54 presents the lidar measurements, tracking results from the VMM adaptation with ellipsoidal shape, and the CAD model, manually aligned with the measurements for comparison.

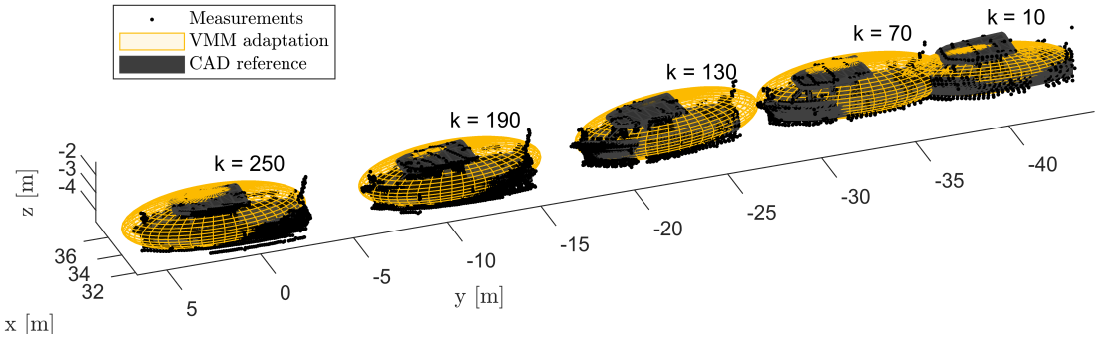
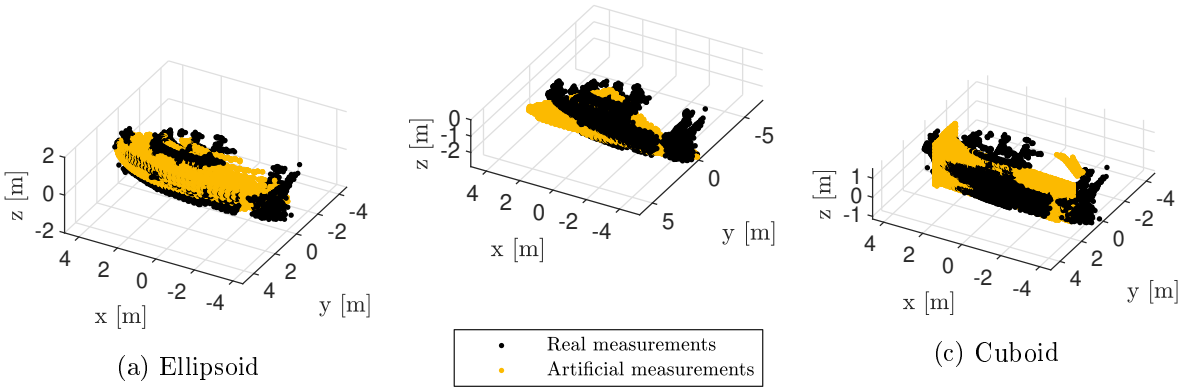


Figure 54: Tracking results for the motorboat "Solgenia". Reference parameters: $l = 8.4\,\mathrm{m}$, $w = 2.5\,\mathrm{m}$, $h = 2.0\,\mathrm{m}$. Estimated parameters (mean): $\tilde{l} = 8.5\,\mathrm{m}$, $\tilde{w} = 2.4\,\mathrm{m}$, $\tilde{h} = 2.5\,\mathrm{m}$ [PH6].

The length and width of the "Solgenia" can be estimated with high precision with an error of only 10 cm. This accuracy is due to the fact that the base, or 2D top view, of the "Solgenia" closely approximates a perfect ellipse. However, the height is somewhat overestimated, particularly toward the bottom, where the estimated ellipsoid diverges significantly from the CAD model. Regarding the height parameter, the elliptical assumption is not entirely met, posing a challenge to precise estimation.
If the ellipsoidal shape is not assumed to be known, the shape must first be classified. For shape classification in this scenario, real and artificial measurements from three different VMMs

(ellipsoid, cuboid, and cone) are accumulated within a sliding window of length $L = 5$. All measurements are transformed into the target coordinate system of the corresponding VMM estimate. For illustration, Figure 55 shows the real and artificial measurements for time steps $k = 10$ through $k = 15$.



(a) Ellipsoid

(b) Cone

(c) Cuboid

Figure 55: Real and artificial measurements for classification [PH6].

It is immediately apparent that the point clouds in Figure 55a are the most similar and that therefore the ellipsoid should be the best-fitting shape for the "Solgenia". Table 9 provides the mean Chamfer distances and the empirical probabilities of each shape achieving the lowest Chamfer distance per time step. In 91.4% of all time steps, the ellipsoid proved to be the best-fitting shape, yielding the lowest Chamfer distance.

Table 9: Classification results for the motor boat [PH6].

	Ellipsoid	Cone	Cuboid
Mean d_C	0.50	0.67	0.59
classified as ... [%]	91.4	4.8	3.8

6.6. Discussion and conclusion for virtual measurement models

In the first part of this thesis, a novel approach for extended object tracking (EOT) was proposed, in which the output of the random matrix algorithm is adjusted using virtual measurement models (VMMs). In Section 6.1, it was shown that the proposed VMM approach leads to unbiased estimates of both object position and spatial extent when tracking elliptical objects with a lidar sensor that produces asymmetric contour measurements. The associated adaptation algorithm, introduced in Section 4.2, was proven to be asymptotically stable in theory (see Section 4.2) and confirmed in practice (see Section 6.1), provided the adaptation gain remains within the range $0 < K_i < 2$.

A comparison with Gaussian process-based shape estimation methods revealed a key strength of the VMM approach: its robustness to measurement noise. This advantage stems from the fact that the VMM operates on the filtered outputs—namely, the estimated mean and covariance—of the random matrix algorithm. As a result, measurement noise is effectively suppressed prior to adaptation. In contrast, methods based on Gaussian processes must associate individual measurements with assumed measurement sources on the object contour, which can lead to significant estimation errors or divergence in high-noise scenarios when measurements are assigned to the wrong part of the object's contour. However, Gaussian processes offer greater flexibility in modeling arbitrary shapes, whereas the VMM framework assumes a set of predefined shape classes. It is worth noting that the VMM adaptation does not affect the filtering process itself. It functions purely as a post-correction stage, preserving the stability and consistency of the underlying random matrix Kalman filter.

In Section 6.4, the VMM approach was compared with various state-of-the-art methods from the literature. Based on the intersection-over-union (IoU) metric, the VMM approach performed competitively in terms of estimation accuracy. More importantly, it stood out for its computational efficiency. This is largely due to the fact that the number of measurements has minimal influence on the adaptation cost, allowing the method to maintain low and predictable computational requirements across the whole scenario.

As an alternative to the iterative VMM adaptation, Section 6.2 explored a regression-based approach using Gaussian processes regression model (GPRM). Here, a GPRM was trained on synthetic data generated by the VMM and used during online tracking to directly map the random matrix output—i.e., centroid and spread—back to object position and shape parameters. This approach led to substantial computational savings, though it also introduced larger estimation errors. The primary challenge was to cover the entire state space with a limited number of training samples. Since this issue becomes more severe in higher-dimensional (3D) applications, the GPRM method was used exclusively in 2D, while the iterative VMM adaptation was retained for 3D scenarios.

In Section 6.3, it was demonstrated that VMMs can also be used to account for occlusions in multi-extended object tracking (MEOT). In particular, ray tracing within the VMM framework enables precise modeling of partial visibility. This approach proved effective as long as objects were only partially occluded. However, as the estimates of the kinematic depends on the center of gravity, they tend to drift during periods with occlusion, resulting in apparent velocity changes that are not yet compensated within the current framework.

Section 6.5 explored various geometric shape models for both 2D and 3D scenarios. Each shape class, such as ellipses, cones, and cuboids, was associated with its own VMM and adaptation loop. After the adaptation, the resulting virtual measurements were accumulated over several time steps and compared with real measurements to enable shape classification. The results indicate that accurate knowledge of the underlying object shape is critical for precise extent estimation and therefore, shape classification is beneficial. Shape classification using Chamfer distances proved to be robust and reliable, even under noisy conditions. Only rectangular cuboids observed strictly from the side could not be classified reliably due to limited observability. In real-world lidar data, a sailing boat was consistently classified as an elliptical cone, while a motorboat was correctly identified as an ellipsoid.

Beyond shape classification, the VMM framework can also be used to classify the underlying measurement likelihood. This includes distinguishing between surface/contour, and interior measurements, or estimating the relative contribution of each in mixed measurement scenarios, an important consideration in practical applications where lidar beams may generate also measurement from the inside of objects. In summary, the proposed framework offers a computationally efficient solution for EOT that flexibly accommodates a wide range of sensor modalities and geometric base shapes. By using VMMs, it becomes possible to model complex measurement model behavior, such as occlusions or mixed measurement sources, within a unified framework.

6.7. Future work

In this work, each object was represented by a single predefined shape. A natural extension is the use of multiple shape components that can be combined, thereby increasing the flexibility of the VMM and enabling more precise physics-based modeling. One illustrative example is the refinement of the sailing boat representation. While an elliptical cone was shown to provide a sufficiently accurate approximation in many scenarios, a more detailed model that combines an ellipsoid for the hull with a half-cone for the sail can further enhance the accuracy, as demonstrated in Figure 56. Virtual measurements can also be generated for this representation by

Figure 56: Sailing boat representation using a combination of a half-ellipsoid for the hull and a half-cone for the sail[6]

applying ray tracing and sampling interior measurements, which can then be compared with real sensor measurements. Assuming that the yaw angle is known from kinematic information and roll and pitch can be neglected, a single ellipsoid requires six parameters (its length, width, height, and position in x, y, and z) to be uniquely defined. These parameters can be determined directly from the random matrix estimate, namely from the filtered center of gravity in x, y, and z together with the three eigenvalues of the filtered covariance matrix.
In contrast, the combined representation requires eleven parameters to be specified (the height of the second component is omitted since the two shapes are stacked). The random matrix result alone is therefore insufficient, and additional metrics must be taken into account, for example Chamfer distances between virtual and real measurements. Furthermore, deriving an intuitive adaptation law is unlikely to be feasible; instead, more advanced machine learning approaches such as neural networks or reinforcement learning will be required.

[5]Sailboat CAD model, available at `https://sketchfab.com/3d-models/sailboat-d59913994a44444b8d054f4de5f519bc`, downloaded on July 30, 2025.

Part II.
Adaptive Birth Densities

7. Detection-driven adaptive birth density

The virtual measurement model (VMM) framework introduced in Part I was primarily concerned with the physical modeling of measurement models in extended object tracking (EOT). In contrast, multi-extended object tracking (MEOT) was addressed only in the context of occlusions, as discussed in Section 4.3. The classical multi object tracking (MOT) challenges presented in Section 4, including, for example, clutter, clustering, data association, and object birth and death, were not complex enough to require advanced solutions. Part II now focuses on MOT scenarios that necessitate more sophisticated modeling of object birth and more advanced filtering techniques, such as Bernoulli and probability hypothesis density (PHD) filters.
Bernoulli filters [42, 43, 45, 47, 50, 97] represent hypothetical objects as tracks, jointly estimating their kinematic states and probabilities of existence. An overview of Bernoulli filters [45] has been provided in Section 2.7. The labeled multi-Bernoulli (LMB) filter [47], which is employed later in this section, was briefly summarized in Section 2.8.
In addition to Bernoulli filters, PHD filters are widely used in MOT. These filters represent the multi-object state using the PHD function, which is defined over the single-object state space. PHD filters, introduced in Section 2.9, are among the most popular MOT approaches and have been applied and extended in numerous publications [54, 56, 98–100].
To enable the detection and initialization of new objects, Bernoulli filters incorporate birth tracks during the prediction step [42, p. 670], while PHD filters introduce a birth intensity [56, p. 4095]. Birth tracks form a multi-Bernoulli random finite set (RFS), where each component is characterized by a birth probability and a spatial birth density. In a Gaussian mixture (GM)-based implementation, if each birth track is represented by a single GM component, the conversion between birth tracks and birth intensity is straightforward.
Modeling a suitable birth density presents several challenges: the initial states of new objects are generally unknown and may vary significantly. For example, larger objects might be detected earlier at greater distances from the sensor and may exhibit higher velocities than smaller ones. Furthermore, the expected initial states can vary over time due to factors such as changing weather or lighting conditions. To address these challenges, adaptive birth densities or intensities have been proposed for both Bernoulli and PHD filters [47, p. 3256],[101, 102]. These adaptive models are typically measurement-driven; that means, they are based on measurements from the previous time step [47, 101, 102].
While measurement-driven models allow the detection of new objects anywhere in the surveillance region without prior information, they come with several drawbacks. In scenarios with high clutter density, both the computational cost and false alarm rate can increase substantially. Additional limitations include delayed detection (by at least one time step) [103, p. 3], sensitivity to user-defined parameters [103, p. 1], and the so-called "spooky action at a distance" phenomenon, wherein measurements from unrelated regions affect the existence probabilities of potential targets [103, p. 4].
The key contribution of this section is a detection-driven adaptive birth density model that can be integrated with existing MOT filters. The term "detection-driven" indicates that the birth density, modeled as a single Gaussian distribution, is updated based on the filter's detections.

Own publications on this subject The novel approach of adapting a birth density using the detections of the filter was first introduced in the conference paper [PH7], specifically in the context of the LMB filter. In this first publication, the birth density was modeled as a single Gaussian distribution. The initial states of the targets detected by the LMB filter were calculated using the Rauch-Tung-Striebel (RTS) recursion, with the goal to recursively estimate the mean and covariance of the target's initial state density, that is ultimately the birth density.
The implementation of the LMB filter used in this work is based on the author's master's thesis [M1], in which various Bernoulli filters were compared across multiple scenarios. That study identified the modeling of the birth density as a promising area for further research. A comprehensive literature review of measurement-driven adaptive birth densities was also published

in [M2, p. 43–45] and [PH9], and is summarized in the next section. Reproduction of textual material, illustrations, and tables from those papers for this section is in accordance with the IEEE and ISIF copyright policies.

7.1. Related work

Measurement-driven adaptive birth densities have been considered in [47, 101, 102, 104–107], primarily due to the limitations of static birth models. Static birth densities with small spatial uncertainties require prior knowledge [47, p. 3256], which is not always available [101, p. 96]. Furthermore, reliable and prompt track initiation becomes challenging if the object has already moved significantly away from the expected location of the birth density [47, p. 3256]. On the other hand, static birth densities with large uncertainties, designed to cover the entire state space, are inefficient [102, p. 1658] and lead to increased false alarm rates, particularly in cluttered environments.

A measurement-driven adaptive birth model was proposed in [47, p. 3256] for the LMB filter. The core idea is to use the current measurement set $\boldsymbol{Z}_k$ to define the multi-Bernoulli birth distribution at the next time step. Measurements that cannot be associated with any existing track are assumed likely to originate from newborn targets [47, p. 3256]. Accordingly, the probability that a given measurement corresponds to a new object is derived from the weights of the association hypotheses. This concept was later extended in [101] for application to the generalized labeled multi-Bernoulli (GLMB) filter.

A related method based on measurement-driven birth intensities was introduced in [102] for PHD and cardinalized probability hypothesis density (CPHD) filters. In this approach, targets are explicitly labeled as either persistent or newborn, and the labels are incorporated into both the prediction and update steps [102, p. 1659]. The labeled state is defined as

$$\underline{x} = \left(\underline{y}, \beta\right), \tag{7.1}$$

where

$$\beta = \begin{cases} 0 & \text{for a persistent target,} \\ 1 & \text{for a newborn target.} \end{cases} \tag{7.2}$$

This leads to a birth intensity of the form [102, p. 1658]:

$$D^{(b)}_{k|k-1}(\underline{x}) = D^{(b)}_{k|k-1}(\underline{y}, \beta) = \begin{cases} D^{(b)}_{k|k-1}(\underline{y}) & \text{if } \beta = 1, \\ 0 & \text{if } \beta = 0. \end{cases} \tag{7.3}$$

Here, $\underline{y}$ denotes the kinematic state, previously referred to as $\underline{x}$. Newborn targets labeled with $\beta = 1$ are considered persistent in the following time step, which is incorporated in the transition model [102]. The update step is also adjusted: since newborn targets must have produced a measurement, their detection probability is set to 1, whereas for persistent targets, it remains p_d [102].

This model was later refined in [104] for use in low-detection-probability environments. Additional variants of measurement-driven adaptive birth densities for PHD filters—incorporating labeling or variable flags to distinguish between newborn and persistent targets—have been explored in [105–107].

An alternative to adaptive birth models is track initiation via a Poisson point process, as implemented in Poisson multi-Bernoulli mixture filters [103]. This approach offers a Bayesian foundation for handling new targets.

7.2. Detection-driven adaptation

In many practical tracking scenarios, the initial states of objects are not known, but can be constrained using contextual information. For example, in air surveillance, newly appearing objects are typically located near runways. To capture such prior knowledge, a birth density can be defined to cover these high-probability regions. However, parameters describing a takeoff, such as position, velocity, and acceleration, remain uncertain. While a large aircraft like an Airbus A380 may take off at the end of the runway at speeds exceeding 300 km/h, lighter aircraft generally lift off earlier and at lower velocities. The true initial states can therefore be modeled as Gaussian-distributed, with unknown mean and covariance. When a new object is detected, its observed kinematic state provides valuable information that can be used to adapt the birth density of the tracking filter. The framework for adaptation is shown in Figure 57.

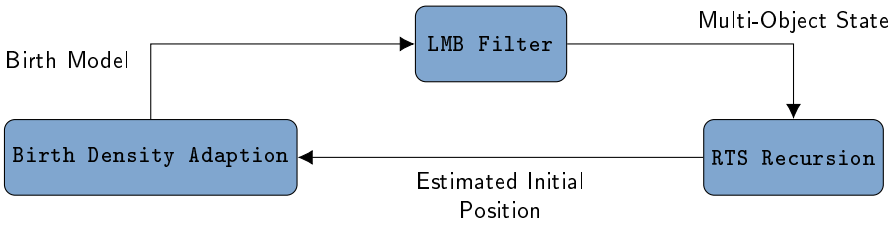


Figure 57: Detection driven adaptive birth density [PH7].

The LMB filter from Section 2.8 is used with an initial birth model to estimate the multi-object state. This birth model consists of a single Gaussian density, defined by its expected value and covariance, a birth probability and a label that contains the current time step k. This ensures that the initialization time of every track is explicitly known.
If an object originating from a birth track initialized at time step k is still alive at time step $k+\Delta k$, the RTS recursion is applied to estimate its initial position. This backward smoothing step provides a refined estimate of the object's state at the time of birth, which is then used to update the birth density. The following two sections summarize the RTS recursion and introduce the method for adapting the birth density based on the estimated initial states.

7.3. Rauch-Tung-Striebel (RTS) recursion

The Rauch-Tung-Striebel (RTS) recursion is a method used for smoothing or retrodiction [58]. The Kalman filter provides state estimates $\underline{x}_{k|k}$ and corresponding covariances $P_{k|k}$ at each time step k. However, its best estimate of the initial state $\underline{x}_0$ considering only the measurement received at $k = 0$ is denoted $\underline{x}_{0|0}$. When observations are available over multiple future time steps, up to $k = 0 + \Delta k$, these additional measurements can be used to refine the estimate of the initial state. This refined estimate is denoted by $\underline{x}_{0|0+\Delta k}$, with associated covariance $P_{0|0+\Delta k}$ and it considers all measurements from time steps 0 to Δk. An overview of the Kalman filter and RTS recursion structure is illustrated in Figure 58.

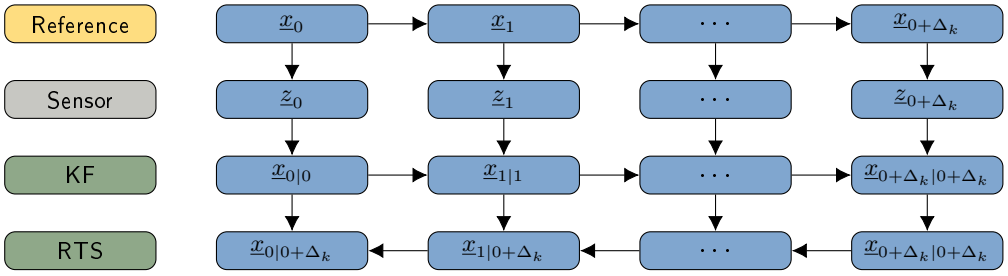


Figure 58: Relationship between the Kalman filter and the RTS recursion [PH7].

The RTS recursion is executed in a loop over all time steps at each time step k. The recursion is initialized with the latest estimate $(\underline{x}_{k|k}, P_{k|k})$ from the Kalman filter. Afterwards, the states of previous time steps l with $l < k$ can be reconstructed in reverse order [58, p. 2203]:

$$\underline{x}_{l|k} = \underline{x}_{l|l} + W_{l|l+1}(\underline{x}_{l+1|k} - \underline{x}_{l+1|l}), \tag{7.4}$$

$$P_{l|k} = P_{l|l} + W_{l|l+1}(P_{l+1|k} - P_{l+1|l})W_{l|l+1}^{\top}, \tag{7.5}$$

with retrodiction gain [58, p. 2203]:

$$W_{l|l+1} = P_{l|l}F^{\top}P_{l+1|l}^{-1}. \tag{7.6}$$

The RTS recursion, as presented here, is specifically designed for the standard Kalman filter, which models a single object's state as a unimodal normal distribution. To enable its application in multi-object filtering scenarios, the RTS recursion must be extended by constructing object trajectories based on label information. In the case of a GM implementation, the component with the highest weight is selected at each time step to estimate the initial position of the corresponding track. The weights of the Gaussian mixture components from earlier time steps remain unchanged throughout this backward smoothing process.

The detection-driven adaptive birth density can also be used without the RTS recursion by taking the state at the time of detection as an estimate of the initial position. This simplification increases compatibility, as it removes the dependency on label information and allows the approach to be used with any filter. However, bypassing the RTS recursion results in a loss of accuracy, since the estimated initial state no longer benefits from the information contained in future observations.

7.4. Birth density adaptation

The true birth density is defined by its mean $\underline{m}_{b,\text{true}}$ and covariance matrix $P_{b,\text{true}}$. Since these true parameters can not be determined directly, they must be estimated. In this context, n denotes the number of detected targets, and $\underline{m}_{0,i}$ represents the initial position of target i. The estimated mean and covariance of the birth density are denoted by $\underline{m}_b$ and P_b, respectively. Using a batch processing approach, standard statistical estimators can be applied to determine these parameters:

$$\underline{m}_b = \frac{1}{n}\sum_{i=1}^{n}\underline{m}_{0,i}, \tag{7.7}$$

$$P_b = \frac{1}{n-1}\sum_{i=1}^{n}(\underline{m}_{0,i} - \underline{m}_b)(\underline{m}_{0,i} - \underline{m}_b)^{\top}. \tag{7.8}$$

By some rewriting, the estimation process can be expressed in a recursive manner. The derivation of this recursive formulation is detailed in [108], and the resulting equations are provided below.

$$\underline{m}_{b|k} = \underline{m}_{b|k-1} + \frac{\underline{m}_{0,k} - \underline{m}_{b|k-1}}{n+1}, \tag{7.9}$$

$$P_{b|k} = \left(1 - \frac{1}{n}\right)P_{b|k-1} + (n+1)\left(\underline{m}_{b|k} - \underline{m}_{b|k-1}\right)\left(\underline{m}_{b|k} - \underline{m}_{b|k-1}\right)^{\top}, \tag{7.10}$$

with the initial condition $\underline{m}_{b|0} = 0$, $P_{b|0} = 0$. The disadvantage of this estimator is that newly detected targets have the same influence as older ones. As the number of updates increases, this leads to a loss of adaptability. To maintain the dynamic behavior of the estimate over time, a

parameter $\tau \geq 1$ is introduced.

$$\underline{m}_{b|k} = \underline{m}_{b|k-1} + \frac{\underline{m}_{0,k} - \underline{m}_{b|k-1}}{\tau + 1}, \tag{7.11}$$

$$P_{b|k} = \left(1 - \frac{1}{\tau}\right) P_{b|k-1} + (\tau + 1) \left(\underline{m}_{b|k} - \underline{m}_{b|k-1}\right) \left(\underline{m}_{b|k} - \underline{m}_{b|k-1}\right)^{\top}. \tag{7.12}$$

As a result, the algorithm always behaves as if τ targets have already been incorporated into the birth density estimate. This means that the weight of a newly detected target remains fixed at $1/(\tau + 1)$, independent of the actual number of previously processed detections.

Proposition 1: The estimation $\underline{m}_b$ is unbiased If the iterative update equation is reformulated as a batch computation, it results in a weighted sum. To express this formally, a weighting factor λ is introduced, leading to the following expression for the estimated birth density mean:

$$\underline{m}_b = \sum_{i=0}^{n} \lambda_i \, \underline{m}_{0,i}, \tag{7.13}$$

$$\lambda_i = \frac{1}{\tau + 1} \left(1 - \frac{1}{\tau + 1}\right)^i. \tag{7.14}$$

Note that $i = 0$ refers to the most recently detected target. As $n \to \infty$, the sum of the weighting factors converges to 1. This can be shown by rewriting the sum:

$$\sum_{i=0}^{n} \lambda_i = \frac{1}{\tau + 1} \sum_{i=0}^{n} \left(1 - \frac{1}{\tau + 1}\right)^i. \tag{7.15}$$

Since the common ratio $1 - \frac{1}{\tau+1} < 1$, this is a convergent geometric series. Applying the formula for the sum of such a series for $n \to \infty$ yields:

$$\sum_{i=0}^{n} \lambda_i \to \frac{1}{\tau + 1} \cdot \frac{1}{1 - \left(1 - \frac{1}{\tau+1}\right)} = 1. \tag{7.16}$$

This allows to compute the expected value of the estimate $\underline{m}_b$ from Equation (7.13):

$$\mathbb{E}(\underline{m}_b) = \mathbb{E}\left(\sum_{i=0}^{n} \lambda_i \, \underline{m}_{0,i}\right) = \sum_{i=0}^{n} \mathbb{E}(\lambda_i \, \underline{m}_{0,i}). \tag{7.17}$$

Since the weighting factors λ_i are deterministic and independent of the random variables $\underline{m}_{0,i}$, it follows that:

$$\mathbb{E}(\underline{m}_b) = \sum_{i=0}^{n} \lambda_i \, \mathbb{E}(\underline{m}_{0,i}). \tag{7.18}$$

Assuming $\mathbb{E}(\underline{m}_{0,i}) = \underline{m}_{b,\text{ref}}$ for all i, and using the result from Equation (7.16), this simplifies to:

$$\mathbb{E}(\underline{m}_b) = \underline{m}_{b,\text{ref}}. \tag{7.19}$$

Proposition 2: The estimation $\underline{m}_b$ is not consistent The covariance of the estimator, denoted as $\text{Cov}(\underline{m}_b)$, does not converge to zero as $n \to \infty$, as expected when using Equation (7.7). The lack of consistency arises due to the unequal weighting factors λ_i applied to the initial positions $\underline{m}_{0,i}$.

The covariance of the estimator (not to be confused with the estimated covariance matrix P_b of the birth density) depends on the true covariance $P_{b,\text{ref}}$ and the parameter τ. According to Bienaymé's identity [109], the covariance of a weighted sum $X = \sum_{i=0}^{n} \lambda_i X_i$ is given by:

$$\text{Cov}(X) = \sum_{i=0}^{n} \lambda_i^2 \, \text{Cov}(X_i). \tag{7.20}$$

In the case of the proposed estimator, it holds that $\text{Cov}(X_i) = \text{Cov}(\underline{m}_{0,i}) = P_{b,\text{ref}}$, leading to:

$$\text{Cov}(\underline{m}_b) = \sum_{i=0}^{n} \lambda_i^2 \, P_{b,\text{ref}}. \tag{7.21}$$

Inserting the expression for λ_i from Equation (7.14) gives:

$$\text{Cov}(\underline{m}_b) = P_{b,\text{ref}} \left(\frac{1}{\tau+1}\right)^2 \sum_{i=0}^{n} \left(1 - \frac{1}{\tau+1}\right)^{2i}. \tag{7.22}$$

Applying the standard formula for the sum in the limit $n \to \infty$ results in:

$$\text{Cov}(\underline{m}_b) = P_{b,\text{ref}} \left(\frac{1}{\tau+1}\right)^2 \cdot \frac{1}{1 - \left(1 - \frac{1}{\tau+1}\right)^2} = \frac{P_{b,\text{ref}}}{2\tau+1}. \tag{7.23}$$

Thus, the covariance of the estimate $\underline{m}_b$ does not vanish as $n \to \infty$, but instead converges to a fixed value determined by the true covariance $P_{b,\text{ref}}$ and the memory parameter τ.

Optimal choice for parameter τ As the value of τ increases, the stationary value can be estimated more precisely according to Equation (7.23). However, if the expected value of the underlying distribution changes over time, a smaller τ is advantageous. It allows the estimator to react more quickly, increasing adaptability at the cost of higher variance. This trade-off is illustrated in the example shown in Figure 59. Initially, samples are drawn from a normal distribution $\mathcal{N}(x; \mu_1 = 10, \sigma = 2)$. At time step $k = 100$, the expected value abruptly changes to $\mu_2 = 20$. Estimators with a small value of τ respond more quickly to sudden changes in the

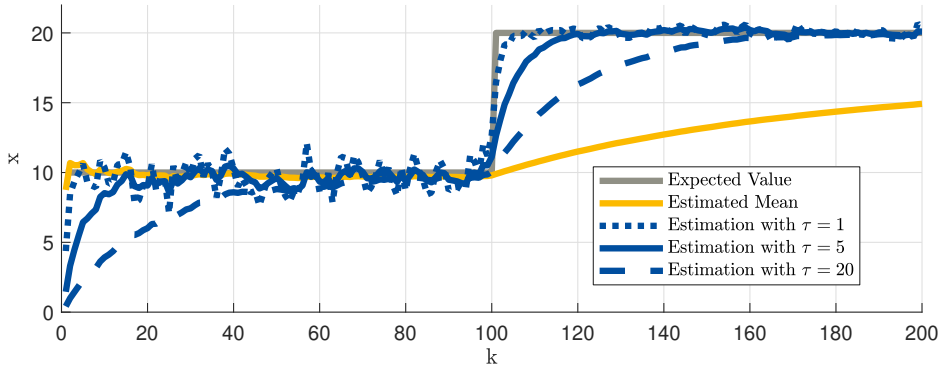


Figure 59: Estimation behavior for different values of τ [PH7].

distribution, but exhibit increased variance in the stationary regime. In contrast, larger values of τ yield smoother estimates but adapt more slowly to dynamic changes. As a compromise between adaptability and estimation precision, a value of $\tau = 5$ was selected in this work.

8. Circular birth density

The approach for an adaptive birth density described in the previous section was limited to a single Gaussian component. This restriction limits the method to scenarios in which targets emerge from only one predominant direction. In environments where targets may originate from multiple directions, the method can be extended to incorporate several Gaussian birth components. This extension is feasible provided that the initial positions of newly detected objects can be reliably assigned to one of these components. A representative example of such a setting is vessel tracking on a river (see results in Section 9.1), where targets can appear either from upstream or downstream.

In this section, the birth density is refined using polar coordinates, resulting in a circular birth density that covers the entire horizon of the observation area. This generalization is particularly relevant for lidar or radar sensors with 360° coverage, in scenarios with open fields of view, such as maritime or aerial environments. Assuming the sensor is positioned at the origin of the coordinate system, the birth density can be described in polar coordinates, incorporating all point-symmetrical quantities like radius r, radial velocity v_r and tangential velocity v_t of newborn objects. To simplify notation, these quantities are combined into a vector $\underline{r} = (r, v_r, v_t)^\top$. It is further assumed that the birth density follows a normal distribution over $\underline{r}$ but is uniform over the orientation angle θ. This results in a cirular birth density given by

$$p_b(\underline{r}, \theta) = \mathcal{N}(\underline{r}; \hat{\underline{r}}, P_b)\mathcal{U}(\theta; 0, 2\pi) \tag{8.1}$$

with expected value $\hat{\underline{r}}$ and covariance matrix P_b. The covariance matrix P_b can be given in diagonal form (e.g. $P_b = \text{diag}\left(\sigma_r^2, \sigma_{v_r}^2, \sigma_{v_t}^2\right)$) but can also consider correlations. Figure 60 shows a circular birth density for $\hat{r} = 100$ m and $\sigma_r = 20$ m as an example.

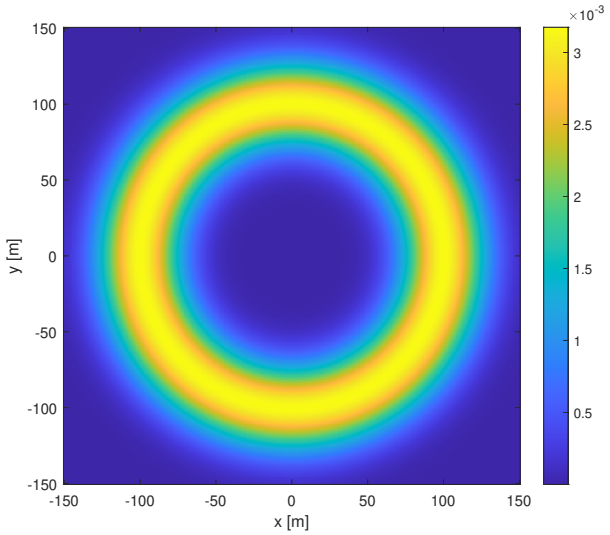


Figure 60: Circular birth density [PH8, PH9, M2]

However, in most tracking scenarios, filters operate in Cartesian coordinates. This is typically the case because motion models are defined for Cartesian state vectors. Therefore, the polar birth density from (8.1) must be approximated in Cartesian coordinates. The approximation is carried out in two steps: First, the circular birth density is represented as a Gaussian mixture in polar coordinates. Second, this mixture is transformed into Cartesian coordinates.

Own publications on this subject The idea of a circular birth density was first introduced in the master's thesis [M2] and further developed in the conference paper [PH8]. In both publications, the birth density is described in polar coordinates and approximated by a Gaussian mixture in Cartesian coordinates. The adaptation can be performed similarly to the previous section, but

carried out in polar space. While the Cartesian approximation proposed in [PH8, M2] was largely heuristic, initial simulation results were promising and are summarized in Section 9.2.
In this work, the entire approach is extended with a more rigorous mathematical foundation for the Cartesian transformation and the Gaussian mixture approximation, with special attention to unimodality and Kullback-Leibler divergences (KLDs). Furthermore, the influence of ego motion is explicitly addressed, and new simulation studies and real-world experiments with ego motion are presented. This section is based on and contains adapted material from the journal paper [PH9], reproduced here in accordance with the ISIF copyright policies.

8.1. Gaussian mixture approximation

In this section, the birth density from Equation (8.1) is approximated by a Gaussian mixture with n_j components, indexed by j. This approximation is necessary, as Gaussian mixtures can be efficiently transformed in the subsequent conversion to Cartesian coordinates. The details of this transformation are discussed in the next section, with Figure 63 providing an illustrative example.
As a general principle, increasing the number of Gaussian components n_j reduces the approximation error introduced during the transformation into Cartesian space. The expected values and standard deviations of the radial and tangential velocities can be directly used in the Gaussian mixture approximation, as their distributions are point-symmetric and thus identical for all GM components. Specifically, we have $\hat{v}_{r_j} = \hat{v}_r$, $\sigma_{v_{r_j}} = \sigma_{v_r}$, $\hat{v}_{t_j} = \hat{v}_t$, and $\sigma_{v_{t_j}} = \sigma_{v_t}$.
However, for radius r and orientation angle θ, the following Gaussian mixture approximation for the circular birth density (as shown in Figure 60) is required:

$$\mathcal{N}\left(r;\hat{r},\sigma_r^2\right)\mathcal{U}\left(\theta;0,2\pi\right) \approx \frac{1}{n_j}\sum_{j=1}^{n_j}\mathcal{N}\left(\begin{bmatrix} r \\ \theta \end{bmatrix};\begin{bmatrix} \hat{r} \\ \hat{\theta}_j \end{bmatrix},\begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\theta^2 \end{bmatrix}\right), \tag{8.2}$$

whereas $\hat{r}$ and σ_r^2 do not differ from the values used in the original density (8.1). With n_j Gaussian components, the expected values of the angular positions are defined as

$$\hat{\theta}_j = \frac{2\pi}{n_j}(j-1), \tag{8.3}$$

with $j = 1,\ldots,n_j$. As a result, the Gaussian components are uniformly distributed over the angular range θ.
Determining the optimal variance σ_θ^2 for an accurate approximation in (8.2) is a more challenging task. In [110], ridgeline and ridgeline elevation plots are used to investigate the topographic properties of multivariate distributions projected onto a one-dimensional space. In this work, the concept of ridgeline elevation is likewise used to evaluate the birth density and to identify a suitable approximation.
In Figure 60, the ridgeline is defined over the interval $\theta = 0$ to $\theta = 2\pi$ at a fixed radius $\hat{r}$. The corresponding ridgeline elevation function is given by

$$p(\theta, r = \hat{r}) = \mathcal{N}\left(\hat{r};\hat{r},\sigma_r^2\right)\mathcal{U}\left(\theta;0,2\pi\right), \tag{8.4}$$

which is constant, with its value determined by $\hat{r}$ and σ_r^2.
Marginalizing over the radius yields the one-dimensional ridgeline elevation density:

$$p(\theta) = \mathcal{U}\left(\theta;0,2\pi\right) = \frac{1}{2\pi} \approx 0.1592. \tag{8.5}$$

Since $p(\theta)$ does not exhibit any extrema, it is a unimodal distribution. Ideally, the Gaussian mixture should approximate this uniform density such that $p(\theta) = \frac{1}{2\pi}$ holds.
In general, however, the Gaussian mixture ridgeline probability density function (pdf) is given

by

$$p_{\mathrm{GM}}(\theta) = \frac{1}{n_j} \sum_{j=1}^{n_j} \frac{1}{\sqrt{2\pi}\sigma_\theta} \exp\left(-\frac{(\theta - \hat{\theta}_j)^2}{2\sigma_\theta^2} \right). \tag{8.6}$$

Proposition 1: With increasing σ_θ, the Gaussian mixture converges to a unimodal uniform distribution When standard Gaussian distributions are applied to a periodic variable such as θ, the periodicity is not inherently taken into account. As a result, the pdf assigns different values to equivalent angles, such as 0, 2π, 4π, and so on. To properly model periodic behavior, a mixture of von Mises distributions can be used. The corresponding pdf is given in [111, Eq. (4)]:

$$p_{\mathrm{VM}}(\theta) = \frac{1}{n_j} \sum_{j=1}^{n_j} \frac{1}{2\pi I_0(\kappa)} \exp\Big(\kappa \cos(\hat{\theta}_j - \theta) \Big), \tag{8.7}$$

where $\kappa = 1/\sigma_\theta^2$, and $I_0(\kappa)$ denotes the modified Bessel function.
For small values of σ_θ, i.e., $\sigma_\theta \ll 2\pi$, the von Mises distribution closely resembles a Gaussian distribution. The key differences compared to the Gaussian pdf are the cosine term in the exponent and the normalization factor, which ensures that $\int_0^{2\pi} p_{\mathrm{VM}}(\theta)\, d\theta = 1$.
To prove Proposition 1, we compute the limit of the pdf as follows:

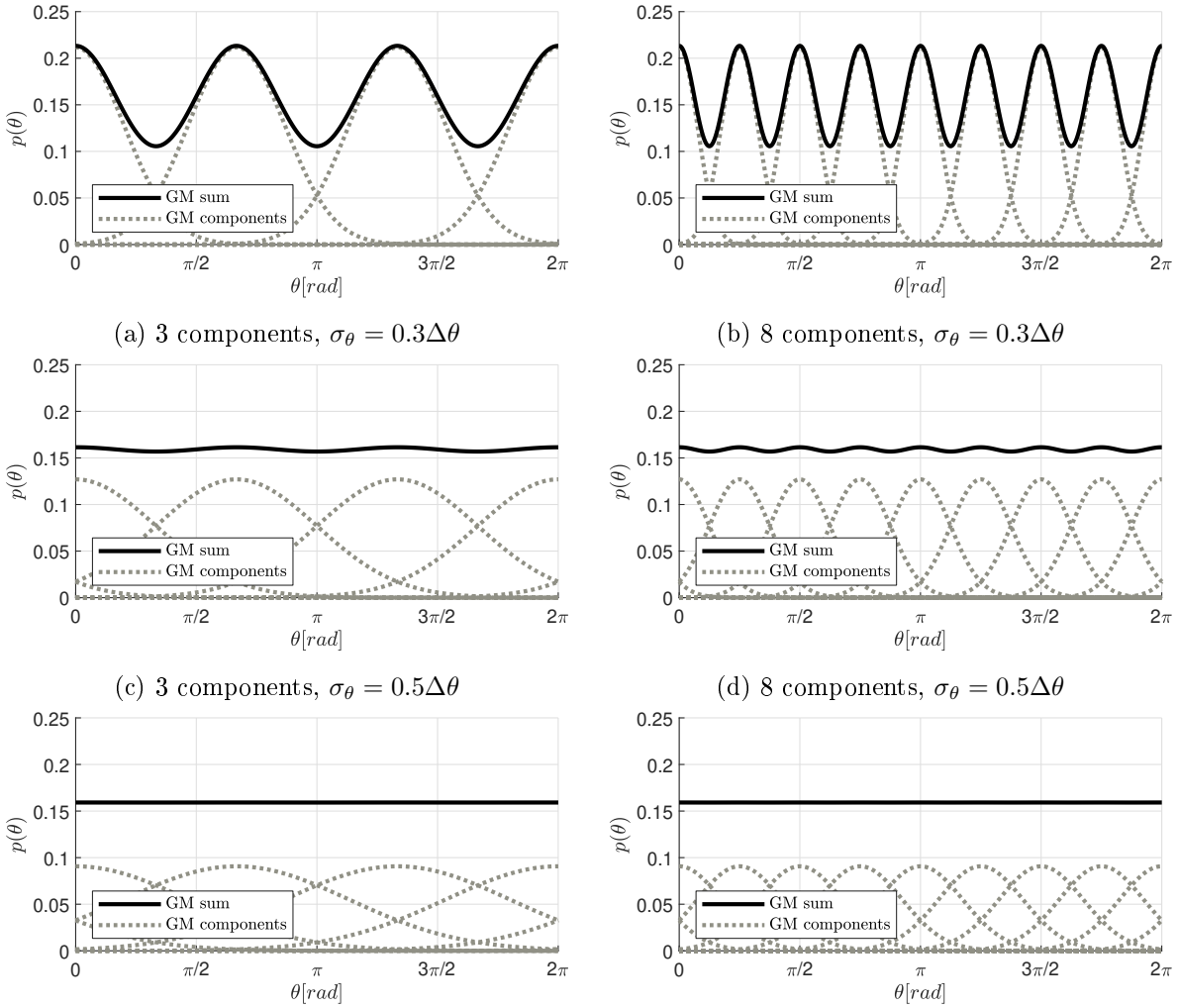
$$\lim_{\kappa \to 0} p_{\mathrm{VM}}(\theta) = \lim_{\kappa \to 0} \frac{1}{n_j} \sum_{j=1}^{n_j} \frac{1}{2\pi I_0(\kappa)} \exp\Big(\kappa \cos(\hat{\theta}_j - \theta) \Big) \tag{8.8}$$

$$= \lim_{\kappa \to 0} \frac{1}{n_j} \sum_{j=1}^{n_j} \frac{1}{2\pi I_0(\kappa)}. \tag{8.9}$$

Since $I_0(0) = 1$ [111, Eq. 5], the expression simplifies to

$$\lim_{\kappa \to 0} p_{\mathrm{VM}}(\theta) = \frac{1}{2\pi}, \tag{8.10}$$

which is equivalent to the pdf of the uniform distribution.
For practical applications, however, it is often more convenient to work with a Gaussian mixture rather than a von Mises mixture. The convergence of the Gaussian mixture to a uniform distribution with increasing uncertainty is illustrated in Figure 61, where the distance between two GM components is denoted as $\Delta_\theta = \theta_j - \theta_{j-1}$. The three figures on the left show the ridgeline densities for three GM components, while the three figures on the right show the results for eight components. From top to bottom, the uncertainty σ_θ increases from $0.3\Delta_\theta$ to $0.7\Delta_\theta$.
For the low uncertainty of $0.3\Delta_\theta$ in Figures 61a and 61b, strong density fluctuations are visible across θ. For a value of $0.5\Delta_\theta$ in Figures 61c and 61d, the density is almost unimodal, with only slight deviations. Since small variance values result in smaller approximation errors when transforming the density into Cartesian coordinates, a value of $0.5\Delta_\theta$ is sufficient for the application. A further increase, such as to $0.7\Delta_\theta$ as shown in Figures 61e and 61f, is not necessary, even though the density here shows almost no fluctuations.



(a) 3 components, $\sigma_\theta = 0.3\Delta\theta$

(b) 8 components, $\sigma_\theta = 0.3\Delta\theta$

(c) 3 components, $\sigma_\theta = 0.5\Delta\theta$

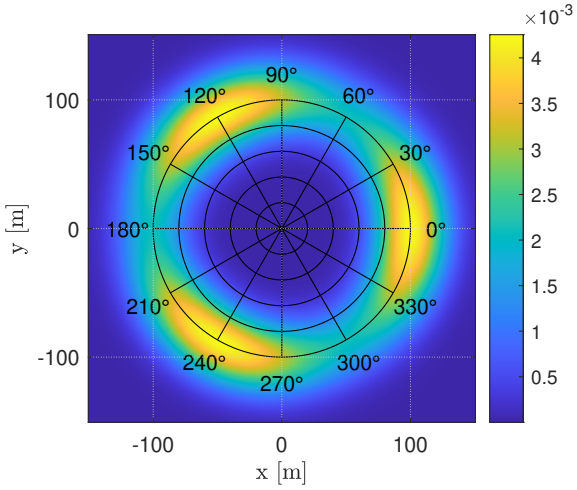
(d) 8 components, $\sigma_\theta = 0.5\Delta\theta$

(e) 3 components, $\sigma_\theta = 0.7\Delta\theta$

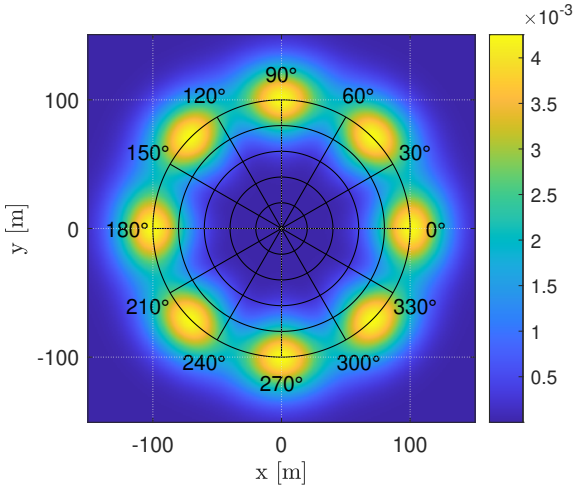
(f) 8 components, $\sigma_\theta = 0.7\Delta\theta$

Figure 61: Unimodal or multimodal Gaussian mixtures for the ridgeline elevation density achieved by marginalizing the birth density over radius r. The random variable is the angle θ which is periodic with a period of $T = 2\pi$ [PH9].

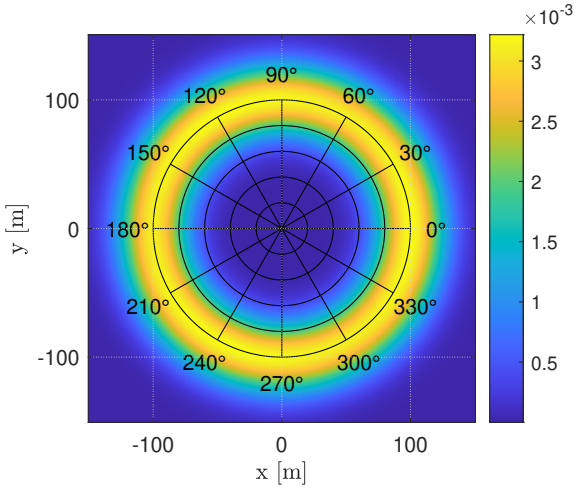
The choice of $\sigma_\theta = 0.5\Delta\theta$ proves beneficial when the GM approximation is visualized in polar coordinates, as shown in Figure 62. With $\sigma_\theta = 0.3\Delta\theta$, as shown in Figures 62a and 62b, clear gaps appear through which objects could potentially pass undetected in tracking scenarios.
In contrast, with $\sigma_\theta = 0.5\Delta\theta$, as shown in Figures 62c and 62d, the birth density resulting from the GM approximation is nearly indistinguishable from the original density from Figure 60. Increasing σ_θ further, for example to $0.7\Delta\theta$, yields no significant improvement and may negatively affect the subsequent Cartesian approximation.
Although the number of GM components may seem irrelevant up to this point, it becomes crucial for the Cartesian approximation discussed in the following section.



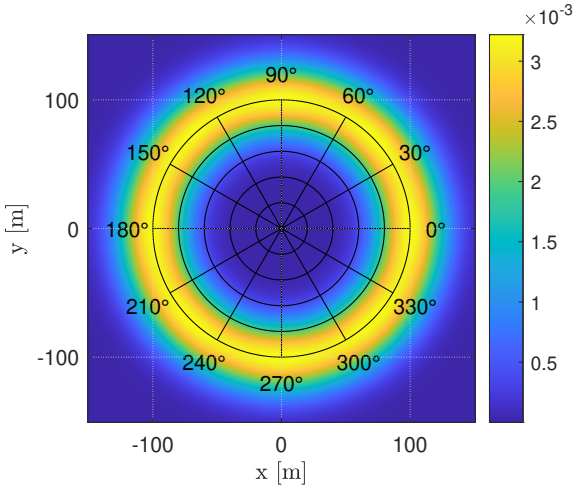
(a) 3 components, $\sigma_\theta = 0.3\Delta\theta$



(b) 8 components, $\sigma_\theta = 0.3\Delta\theta$



(c) 3 components, $\sigma_\theta = 0.5\Delta\theta$



(d) 8 components, $\sigma_\theta = 0.5\Delta\theta$

Figure 62: Gaussian mixtures in polar coordinates [PH9].

8.2. Cartesian transformation of a polar Gaussian distribution

The transformation of a Gaussian distribution from polar coordinates, $p_{\text{polar}}(r, \theta)$, into Cartesian coordinates, $p_{\text{cart}}(x, y)$, can be illustrated graphically, as shown in Figure 63. Mathematically, this transformation can be expressed as follows:

$$p_{j,\text{cart}}(x, y) = \mathcal{N}\left(\begin{bmatrix} x \\ y \end{bmatrix}; \begin{bmatrix} \hat{x}_j \\ \hat{y}_j \end{bmatrix}, \underbrace{\begin{bmatrix} \sigma_{j,x}^2 & 0 \\ 0 & \sigma_{j,y}^2 \end{bmatrix}}_{P_{j,\text{cart}}}\right) \approx p_{j,\text{polar}}(r, \theta) = \mathcal{N}\left(\begin{bmatrix} r \\ \theta \end{bmatrix}; \begin{bmatrix} \hat{r} \\ \hat{\theta}_j \end{bmatrix}, \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\theta^2 \end{bmatrix}\right) \tag{8.11}$$

The transformation of the expected value is straightforward:

$$\hat{x}_j = \hat{r}\cos(\hat{\theta}_j), \tag{8.12}$$

$$\hat{y}_j = \hat{r}\sin(\hat{\theta}_j). \tag{8.13}$$

For the variances, we first consider the density in Figure 63 for $\hat{\theta} = 0°$.

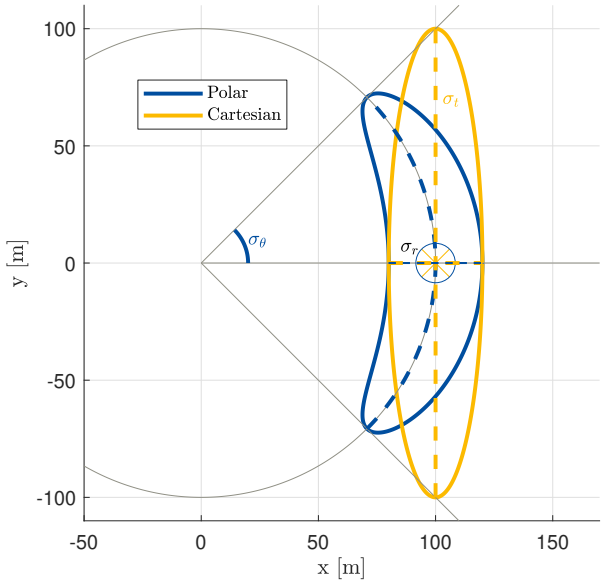


Figure 63: $1\sigma-$ ellipses for the Cartesian approximation of a polar Gaussian distribution[PH9].

In Figure 63, the radial standard deviation σ_r corresponds to σ_x, while the angular standard deviation σ_θ must be converted into a tangential standard deviation σ_t, given by $\sigma_t = \hat{r}\tan(\sigma_\theta)$. Considering the general case for rotated GM components, the Cartesian covariance matrix $P_{j,\text{cart}}$ is obtained through the following transformation:

$$P_{j,\text{cart}} = R_j V_{rt} R_j^\top, \tag{8.14}$$

where V_{rt} is the covariance matrix in the radial-tangential coordinate system:

$$V_{rt} = \begin{pmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_t^2 \end{pmatrix}, \tag{8.15}$$

and R_j is the rotation matrix defined as:

$$R_j = \begin{pmatrix} \cos(\hat{\theta}_j) & -\sin(\hat{\theta}_j) \\ \sin(\hat{\theta}_j) & \cos(\hat{\theta}_j) \end{pmatrix}. \tag{8.16}$$

Since the transformation to Cartesian coordinates is an approximation, it causes an approximation error that can be quantified using the Kullback-Leibler divergence (KLD) [112]. The KLD measures the discrepancy between two probability distributions and is defined as [112]:

$$d(P_{\text{cart}} \| P_{\text{polar}}) = \int_{-\infty}^{\infty} p_{\text{cart}}(x, y) \log \left(\frac{p_{\text{cart}}(x, y)}{p_{\text{polar}}(r, \theta)} \right) dx\, dy. \tag{8.17}$$

However, due to the different random variables involved, the KLD is evaluated in a discrete form in this context. Both continuous densities $p_{\text{cart}}(x, y)$ and $p_{\text{polar}}(r, \theta)$ are discretized, and the divergence is computed using [113, p. 19]:

$$d(P_{\text{cart}} \| P_{\text{polar}}) = \frac{1}{n} \sum_{i=1}^{n} p_{\text{cart}}(x_i, y_i) \log \left(\frac{p_{\text{cart}}(x_i, y_i)}{p_{\text{polar}}(r_i, \theta_i)} \right), \tag{8.18}$$

where n is the number of samples, and $p_{\text{cart}}(x_i, y_i)$ and $p_{\text{polar}}(r_i, \theta_i)$ are the normalized density values at the corresponding sample positions. The normalization ensures that $\sum_{i=1}^{n} p_{\text{cart}}(x_i, y_i) = \sum_{i=1}^{n} p_{\text{polar}}(r_i, \theta_i) = 1$.
In the following, it is investigated how the KLD behaves for different numbers of GM components. For the discretization, 10 000 Cartesian samples are uniformly distributed such that $x_{\min} = \hat{x} - 3\sigma_x$, $x_{\max} = \hat{x} + 3\sigma_x$, $y_{\min} = \hat{y} - 3\sigma_y$ and $y_{\max} = \hat{y} + 3\sigma_y$. For each sample, the corresponding (r, θ) values are computed, the densities are evaluated and normalized, and the KLD is then calculated using Equation (8.18). The results for the current example birth density are shown in Figure 64 for different numbers of Gaussian components.

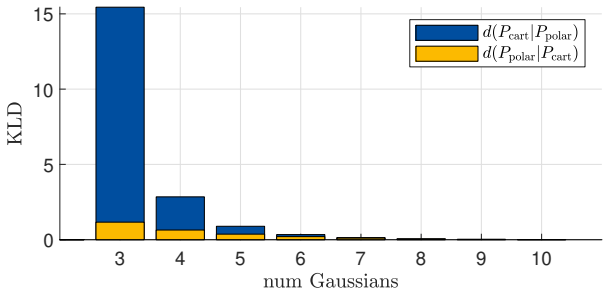
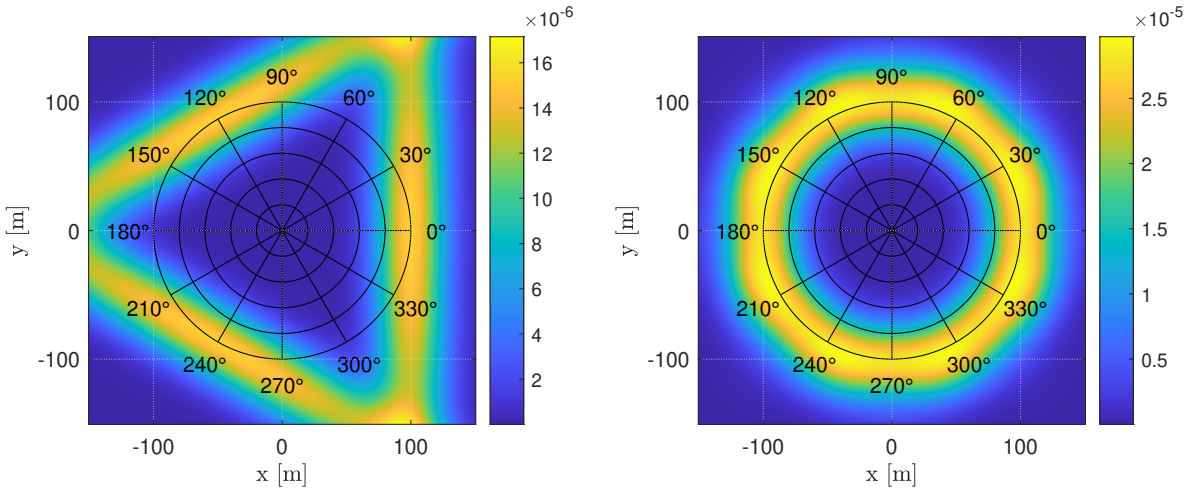


Figure 64: KLD for the approximation of a polar Gaussian density by a Cartesian density [PH9].

The KLD is an asymmetric measure, meaning that $d(P_{\text{cart}} \| P_{\text{polar}}) \neq d(P_{\text{polar}} \| P_{\text{cart}})$. However, the behavior of the two directions shown in Figure 64 is qualitatively similar. As the number of GM components increases, the KLD decreases exponentially and converges to zero.
Due to the observed symmetry and the low resulting KLD, a total of eight Gaussian components is used in the following sections.
Previously, it was noted that a larger value for σ_θ (e.g., $0.7\Delta_\theta$) negatively affects the Cartesian approximation. With eight GM components and $\sigma_\theta = 0.5\Delta_\theta$, the 1-$\sigma$ angular range of a component covers approximately 45°. In contrast, for $\sigma_\theta = 0.7\Delta_\theta$, this range increases to roughly 63°, which results in a larger KLD, as the curvature of the circular structure is increasingly neglected.
The final result of the GM approximation and the Cartesian transformation is shown in Figure 65. The birth density depicted in Figure 65b, based on eight components and $\sigma_\theta = 0.5\Delta_\theta$, closely matches the original density in Figure 60 and is therefore recommended.



(a) 3 components, $\sigma_\theta = 0.5\Delta\theta$
KLD $d(P_{\text{cart}}|P_{\text{polar}}) = 15.4$

(b) 8 components, $\sigma_\theta = 0.5\Delta\theta$
KLD $d(P_{\text{cart}}|P_{\text{polar}}) = 0.06$

Figure 65: Birth densities after GM and Cartesian approximation [PH9].

8.3. Birth velocities

Newly detected objects are typically expected to move toward the sensor, which corresponds to a negative radial velocity v_r. The expected tangential velocity v_t is assumed to be zero. As a result, the expected Cartesian velocity components v_x and v_y depend on the angle θ and therefore differ across the individual GM components.

Figure 66 illustrates the relationship between velocity components in polar and Cartesian coordinates.

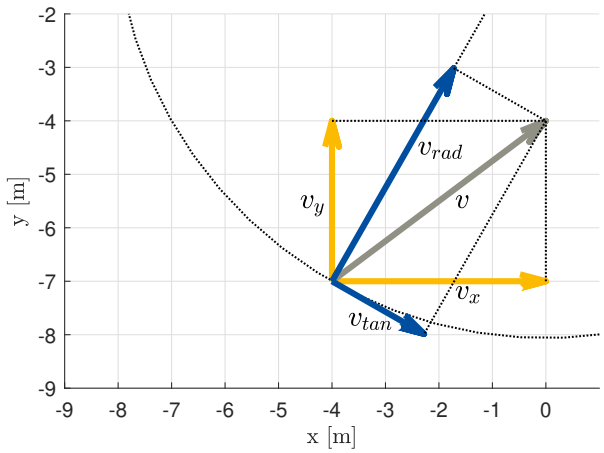


Figure 66: Velocities in polar and Cartesian coordinates [PH8, PH9, M2].

From the radial and tangential velocity components, the expected Cartesian velocities $\hat{v}_x$ and $\hat{v}_y$ can be computed as follows:

$$\begin{pmatrix} \hat{v}_x \\ \hat{v}_y \end{pmatrix} = R_j \begin{pmatrix} \hat{v}_r \\ \hat{v}_t \end{pmatrix}, \tag{8.19}$$

where R_j denotes the rotation matrix associated with the corresponding GM component.

The covariance matrix of the velocity in Cartesian coordinates is given by:

$$P_{j,\text{cart,vel}} = R_j \begin{pmatrix} \sigma_{v_r}^2 & 0 \\ 0 & \sigma_{v_t}^2 \end{pmatrix} R_j^\top. \tag{8.20}$$

Here, σ_{v_r} and σ_{v_t} denote the standard deviations of the radial and tangential velocities, respectively, as used in Equation (8.1).

Summary and birth intensities for PHD filters The resulting birth intensity for Cartesian PHD filters is given by:

$$D_{k|k-1}^{(b)}(\underline{x}) = \frac{p_{b,0}}{n_j} \sum_{j=1}^{n_j} \mathcal{N}\left(\underline{x}; \underline{m}_j, P_j\right), \tag{8.21}$$

where $p_{b,0}$ denotes the birth probability and n_j the number of GM components. Based on prior investigations, the use of eight components is recommended in practical applications. The expected value $\underline{m}_j$ of component j is defined as:

$$\underline{m}_j = \left(x_j, y_j, v_{x_j}, v_{y_j}\right)^\top, \tag{8.22}$$

and the associated covariance matrix P_j is given by:

$$P_j = \begin{pmatrix} P_{j,\text{cart}} & 0 \\ 0 & P_{j,\text{cart,vel}} \end{pmatrix}. \tag{8.23}$$

Instead of using a constant velocity (CV) model, an alternative approach could involve a constant turn model, where the state vector is defined as $\underline{x}_{ct} = \left(x, y, v, \phi, \omega\right)^\top$, with v denoting the total velocity and ϕ the orientation. These quantities can be derived from the radial and tangential velocities of the circular birth density as follows:

$$v_j = \sqrt{\hat{v}_{r_j}^2 + \hat{v}_{t_j}^2} = \sqrt{\hat{v}_{x_j}^2 + \hat{v}_{y_j}^2}, \tag{8.24}$$

$$\phi_j = \text{atan2}(v_{y_j}, v_{x_j}). \tag{8.25}$$

8.4. Detection-driven adaptation for a circular birth density

In the previous sections, it was assumed that the parameters of the circular birth density in polar coordinates, namely the expected value $\hat{\underline{r}}$ and the covariance matrix P_b, were known and remained constant. Analogous to the approach in Section 7, where detection-driven adaptation was introduced for a single Gaussian birth density, a corresponding mechanism is now proposed for the circular birth density. While the radius may still be available from sensor specification sheets, initial values for radial and tangential velocities are often difficult to estimate, especially with regard to their associated uncertainties. Figure 67 illustrates the proposed framework for detection-driven adaptation of the circular birth density.

The starting point is the birth density in polar coordinates, given by Equation (8.1), which is shown on the far left in Figure 67. Its Gaussian mixture approximation and subsequent transformation to Cartesian coordinates have already been discussed.

The MOT filter, which operates in Cartesian coordinates, is primarily responsible for detecting and tracking objects. In this context, however, it also takes on the secondary task of estimating their initial states. This can be achieved, for instance, by applying a backward recursion once an object has been successfully observed over multiple time steps, for a more accurate estimate of its initial state. This approach was described in Section 7 for the LMB filter using the RTS recursion.

If MOT filters based on sets of trajectories are used, a separate backward recursion is no longer necessary. In that case, the initial state is continuously updated as new measurements are

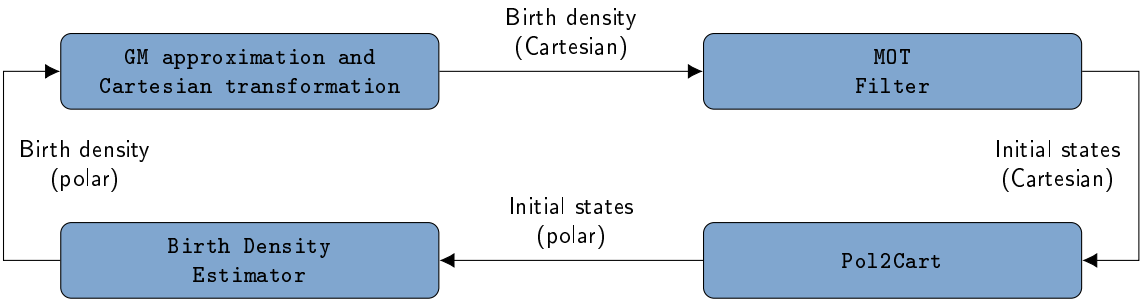


Figure 67: Detection-driven birth density adaptation for a circular birth density [PH8, PH9].

received. The resulting estimate can then be treated as a sample from the birth density for the purpose of estimating its parameters.
Since the birth density is described in polar coordinates, a coordinate transformation from Cartesian back to polar is required. The transformation from Cartesian to polar coordinates is given by:

$$r = \sqrt{x^2 + y^2}, \tag{8.26}$$
$$v_r = \cos(\alpha)v_x + \sin(\alpha)v_y, \tag{8.27}$$
$$v_t = -\sin(\alpha)v_x + \cos(\alpha)v_y \tag{8.28}$$

with angle $\alpha = \operatorname{atan2}(y, x)$. For the update of the birth density, the initial state of object n in polar coordinates is defined by $\underline{r}_{0,n} = \left(r_{0,n}, v_{r_{0,n}}, v_{t_{0,n}}\right)^\top$.
To estimate the expected value and the covariance matrix, the same recursive estimators with weighting parameter τ as introduced in Section 7.4 for the single Gaussian case can be applied here:

$$\hat{\underline{r}}_n = \hat{\underline{r}}_{n-1} + \frac{\underline{r}_{0,n} - \hat{\underline{r}}_{n-1}}{\tau}, \tag{8.29}$$
$$P_{b_n} = \left(1 - \frac{1}{\tau - 1}\right) P_{b_{n-1}} + \tau \left(\hat{\underline{r}}_n - \hat{\underline{r}}_{n-1}\right)\left(\hat{\underline{r}}_n - \hat{\underline{r}}_{n-1}\right)^\top. \tag{8.30}$$

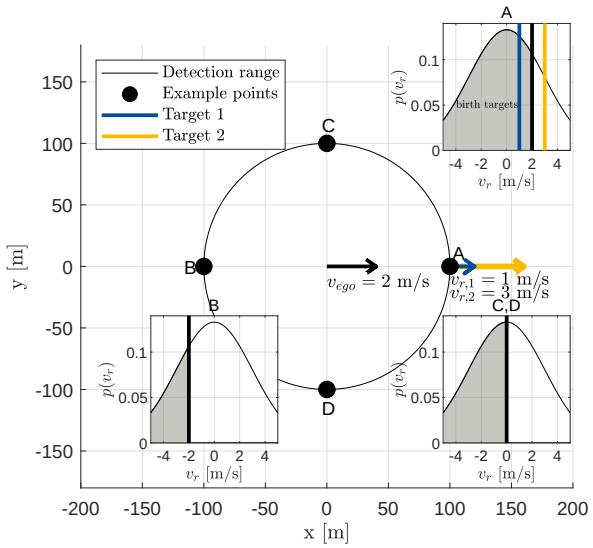
8.5. Effects of ego motion

In the previous sections, it was assumed that the sensor remained stationary throughout the scenario. However, in cases where the sensor itself is in motion, the influence of ego motion on the birth density must be taken into account. For the remainder of this section, the ego motion is described by the velocity v_{ego} and direction θ_{ego}. In addition, the random variable $v_{r,0} \sim p(v_{r,0})$ is introduced to represent the prior distribution of radial velocities of the targets.

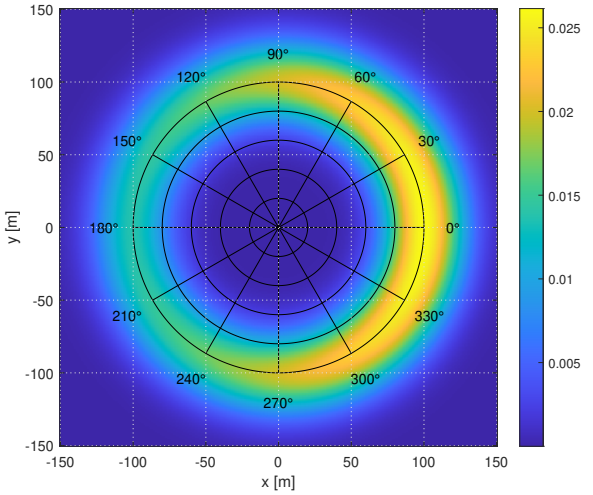
Probability of birth The relevance of ego motion becomes evident when considering two extreme cases: negligible ego motion ($v_{\text{ego}} = 0$) and dominant ego motion ($v_{\text{ego}} \gg v_{r,0}$ or $v_{r,0} \approx 0$). In the first case, where the sensor remains stationary, new targets are expected to appear uniformly from all directions, as illustrated in Figure 60. In contrast, if v_{ego} significantly exceeds the velocity of the targets, new objects are expected primarily in the direction of motion, specifically within the angular direction $\theta_{\text{ego}} \pm 90°$. In general, the birth probability p_b becomes a function of the angle θ. This angular dependence is incorporated via an additional factor $p_{b,\text{ego}}(\theta)$.
In the presence of ego motion, the birth intensity is therefore expressed as:

$$D^{(b)}_{k|k-1}(\underline{r}, \theta) = p_{b,0}\, p_{b,\text{ego}}(\theta)\, \mathcal{N}(\underline{r}; \hat{\underline{r}}, P_b)\, \mathcal{U}(\theta; 0, 2\pi). \tag{8.31}$$

To derive the function $p_{b,\text{ego}}(\theta)$, we consider the example shown in Figure 68.



(a) Birth probability and radial velocity at different points



(b) Circular birth density for ego motion with $v_{\text{ego}} = 2\,\text{m/s}$, $\theta_{\text{ego}} = 0°$, and $p(v_{r,0}) = \mathcal{N}\left(v_{r,0}; 0, 3^2\right)$

Figure 68: Introductory example for ego motion consideration within circular birth densities [PH9].

In this example, the ego motion is defined by a velocity of $v_{\text{ego}} = 2\text{m/s}$ and a direction of $\theta_{\text{ego}} = 0°$. The expected detection range is $\hat{r} = 100\text{m}$. At point A, two hypothetical targets are considered: Target 1 with a radial velocity of $v_{r,1} = 1\text{m/s}$ and Target 2 with $v_{r,2} = 3\text{m/s}$.
Intuitively, Target 1 is likely to be detected, as its radial velocity is lower than the ego motion velocity. In contrast, Target 2 is moving away too quickly and is therefore not expected to be detected at that location. As a prior distribution for the radial velocity, $p(v_{r,0}) = \mathcal{N}(v_{r,0}; 0,\ 3^2)$ is assumed, from which the two example velocities are drawn.
At point A, all targets with a radial velocity below 2m/s are considered plausible birth targets (see Figure 68a, upper right subfigure). The contribution of ego motion to the birth probability corresponds to the shaded area under the distribution curve and is computed as:

$$p_{b,\text{ego}}(\theta) = 2 \int_{-\infty}^{v_{\text{ego}} \cos(\theta + \theta_{\text{ego}})} p(v_{r,0}) dv_{r,0} \tag{8.32}$$

$$= 2\Phi \left(\frac{v_{\text{ego}} \cos\left(\theta + \theta_{\text{ego}}\right) - \hat{v}_{r,0}}{\sigma_{v_{r,0}}} \right) \tag{8.33}$$

where $\Phi(x)$ denotes the cumulative distribution function (CDF) of the standard normal distribution. The factor of 2 serves to normalize the CDF, since without ego motion term, the result would otherwise be 0.5. The computation of the ego-motion-dependent birth probability $p_{b,\text{ego}}$ via Equation (8.33) is not limited to normally distributed radial velocities; the method is compatible with arbitrary prior distributions. The circular birth density for the previously introduced example in Figure 60, now incorporating ego motion and assuming prior radial velocities as shown in Figure 68a, is visualized in Figure 68b. It demonstrates that the birth probability increases in the direction of motion, while it decreases on the opposite side.
In the context of ego motion, it is also necessary to account for the sensor's own position. This requirement can be efficiently addressed by applying a simple coordinate shift. It is advantageous to perform this shift after the Cartesian approximation, as this enables a simplified representation of the birth density in polar coordinates, as shown in Equation (8.31).

Birth density for radial velocity In addition to the birth probability, ego motion has also a significant impact on the expected radial velocities of newly born targets. For example, the

radial velocity of a target born at point A in Figure 68a must be less than 2m/s, whereas for a target at point B it is constrained to be below −2m/s. As a result, the birth density $b(v_r)$ for the radial velocity becomes dependent on the angle θ. If the prior distribution of the radial velocity of newborn targets is Gaussian, then the birth density $b(v_r)$ can be described by a truncated normal distribution. The corresponding pdf is given by (adapted from [114, p. 757]):

$$b(v_r, \theta) = \frac{1}{\sigma_{v_{r,0}}} \frac{\phi\left(\frac{v_r - \hat{v}_{r,0}}{\sigma_{v_r}}\right)}{\Phi\left(\frac{v_{\text{ego}} \cos(\theta + \theta_{\text{ego}}) - \hat{v}_{r,0}}{\sigma_{v_{r,0}}}\right)} \tag{8.34}$$

where $\phi(x)$ represents the pdf of the standard normal distribution. The expected value and variance of this truncated normal distribution (8.34) are given by (see e.g. [114, p. 759]):

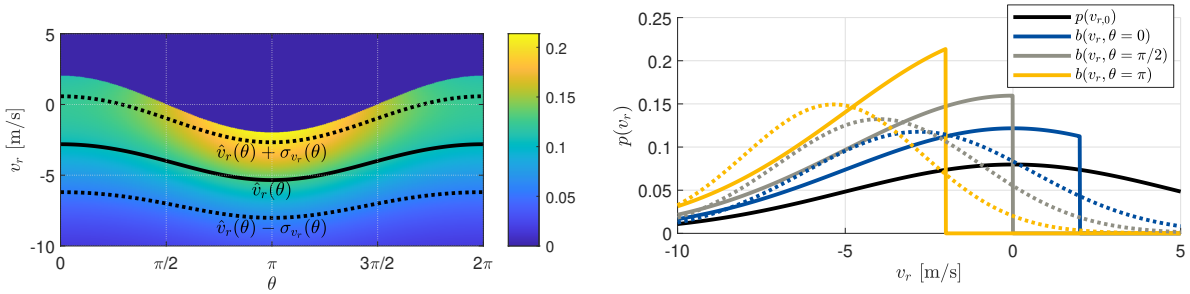
$$\hat{v}_r(\theta) = \hat{v}_{r,0} - \sigma_{v_{r,0}} \frac{\phi(\beta(\theta))}{\Phi(\beta(\theta))}, \tag{8.35}$$

$$\sigma_{v_r}^2(\theta) = \sigma_{v_{r,0}}^2 \left[1 - \beta \frac{\phi(\beta(\theta))}{\Phi(\beta(\theta))} - \left(\frac{\phi(\beta(\theta))}{\Phi(\beta(\theta))}\right)^2\right], \tag{8.36}$$

where

$$\beta(\theta) = \frac{v_{\text{ego}} \cos(\theta - \theta_{\text{ego}}) - \hat{v}_{r,0}}{\sigma_{v_{r,0}}}. \tag{8.37}$$

For the example illustrated in Figure 68a, the pdf (8.34), expected value (8.35), and standard deviation (square root of (8.36)) are shown in Figure 69a for various angles θ.



(a) pdf, expected value and standard deviation

(b) Gaussian approximations (dotted lines) for different angles resp. truncation points.

Figure 69: Truncated normal distribution for the radial velocity [PH9].

At point A in Figure 68a, newborn targets are expected to have a radial velocity of $v_r(\theta = 0) = -2.8 \pm 3.4$m/s, whereas at point B, the expected radial velocity is $v_r(\theta = \pi) = -5.3 \pm$ m/s as shown in Figure 69a. Since it is impractical to employ a combination of normal and truncated normal distributions in the MOT filter, the truncated normal distribution is approximated by a normal distribution using its mean and variance, i.e., a second-order moment approximation. Figure 69b illustrates this Gaussian approximation for the example scenario from Figure 68a and for the angular positions corresponding to points A through D.

9. Results

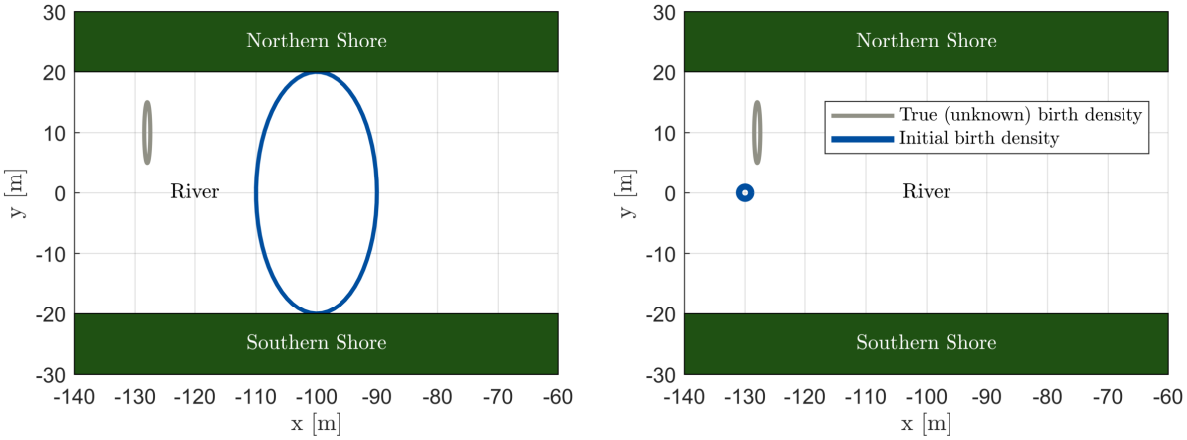
This section presents both simulation studies and real-world experiments concerning the detection-driven adaptive birth densities. The following scenarios are covered:

- Single Gaussian birth density adaptation using an LMB filter and RTS, in line with the framework described in section 7. The results of this scenario are published in the conference paper [PH7].
- Adaptation of the circular birth density with 8 GM components, following the approach from section 8. This scenario was initially explored in the master's thesis [M2, p. 68-69] and subsequently extended for the publications [PH8, p. 6-8] and [PH9, p. 12-14]. The extended version, based on the publications [PH8, PH9], is presented in the further course of this section.
- Simulation studies with ego motion consideration within a circular birth density, as outlined in section 8.5, and published in [PH9, p. 14-16].
- Full-scale experiments with lidar data from a maritime scenario recorded on Lake Constance, presented in [PH9, p. 16-18].

In all scenarios involving the circular birth density, a trajectory probability hypothesis density (TPHD) filter is utilized for MOT.

9.1. Single Gaussian birth density

The first simulation study focuses on tracking ships on a river. In the observation area, where there are no piers, new objects can enter the scene from either the right or left side. For simplicity, only the left side is considered, enabling the use of a single Gaussian birth density. This birth density spans the entire width of the river, as illustrated in Figure 70a.



(a) large initial birth density

(b) small initial birth density

Figure 70: River scenario [PH7].

As it turns out, the initial Gaussian birth density is oversized. In reality, the ships navigate slightly to the left of the middle of the river due to the left-hand traffic rules. The low speed of the ships relative to the sensor sampling rate results in a relatively constant x-position for newly detected targets. Additionally, the narrowness of the river at this point imposes a speed limit, causing the ships to travel at roughly the same speed. The sensor is positioned at the origin and is assumed to have a range of 100m, leading to the placement of the birth density at $x = -100$m. However, in practice, ships begin to generate measurements at a distance of approximately 128 m.

In the second simulation, it is incorrectly assumed that the ships navigate precisely in the middle of the river, as illustrated in Figure 70b. Furthermore, the range of the sensor is slightly overestimated. In both simulations, the expectation value of the true initial state $(x, y, v_x, v_y, a_x, a_y)$ is given as:

$$m_{b,\text{ref}} = (-128, 10, 5, 0, 0, 0) \tag{9.1}$$

with the uncertainties:

$$\sigma_{b,\text{ref}} = (0.5, 5, 0.2, 0.01, 0.01, 0.01). \tag{9.2}$$

Additive white noise affects the acceleration with $\sigma_{a_x} = 0.05\,\text{m/s}^2$ in the x-direction and $\sigma_{a_y} = 0.01\,\text{m/s}^2$ in the y-direction. The sensor is characterized by a measurement noise of $\sigma_x = \sigma_y = 0.1\,\text{m}$, a detection probability of $p_d = 0.98$, and an average clutter detection rate of $\lambda_c = 10$. Clutter is uniformly distributed within the area $-130\,\text{m} < x < -90\,\text{m}$ and $-20\,\text{m} < y < 20\,\text{m}$. The scenario spans $k = 10000$ time steps, with a sampling period of 0.1 s. New targets appear at time steps $k = 0, 100, 200, \ldots$, and leave the surveillance area at $k = 85, 185, 285, \ldots$. Cardinality estimation, optimal subpattern assignment (OSPA), localization OSPA, and cardinality OSPA metrics are used to evaluate the results across 100 Monte Carlo runs.

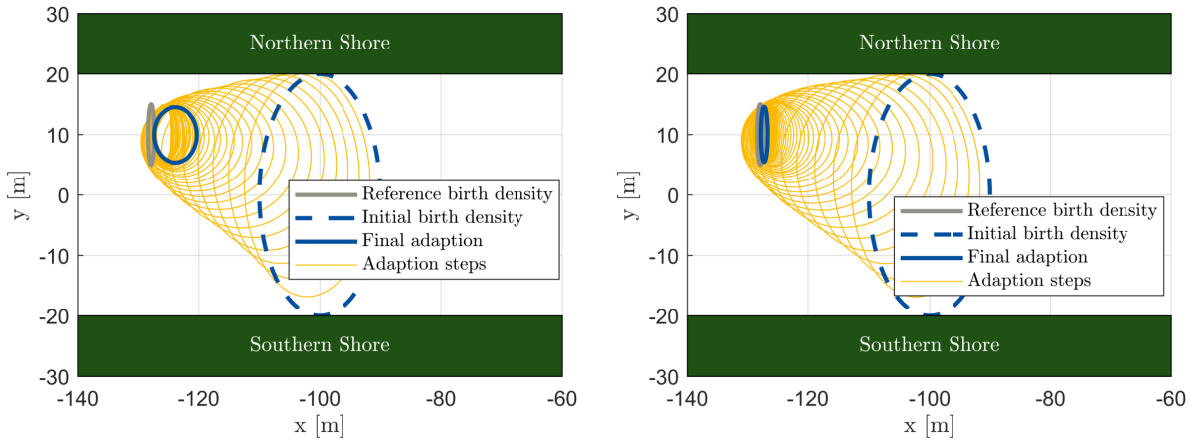
Initial birth density with large uncertainty In the first scenario, the initial birth density of the LMB filter is defined as:

$$m_{b,\text{filter}} = (-100, 0, 5, 0, 0, 0), \tag{9.3}$$

with large uncertainties (Figure 70a):

$$\sigma_{b,\text{filter,large}} = (10, 20, 1, 1, 1, 1). \tag{9.4}$$

Since the targets move in the positive x-direction, each target will cross the birth density after a few seconds. Even without any adaptation, all targets will eventually be detected, although with a delay. The scenario is evaluated twice: once with RTS recursion and once without it. Figure 71 illustrates the adaptation of the birth density.



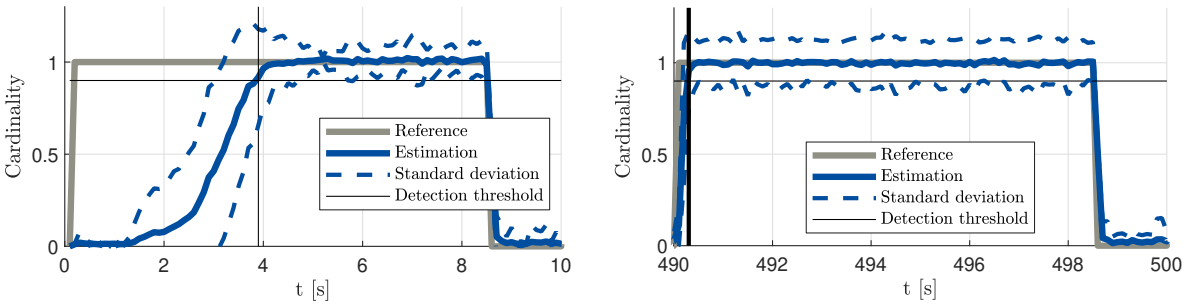
(a) without RTS recursion

(b) with RTS recursion

Figure 71: Adaption of the birth density (Adaptation steps at $k = 100, 200, \ldots$) averaged over 100 Monte Carlo (MC) runs [PH7].

Figure 71b demonstrates that the RTS recursion improves the estimation of the birth density, as its final state is closer to the true initial birth density compared to the result shown in Figure 71a. Since the objects move in the positive x-direction, the detection point typically exhibits a positive x-bias. Theoretically, this bias should approach zero when using the RTS recursion. However, one issue arises: a target born at $k = k_0$ is not always assigned to the birth track at k_0 but may instead be assigned to the birth track at $k_1, k_2, \ldots$. Consequently, the RTS recursion for such a track can only go backward to the time step at which the track was created. This

limitation causes a small offset. Nevertheless, the computational cost of the RTS recursion is very low since it is performed only once for each target. If a track born at $k = k_0$ remains active at $k = k_0 + \Delta k$, the RTS recursion is applied to estimate the kinematic state at k_0. In this simulation, Δk is 20.
To investigate the effects of a well-known birth density on tracking performance, cardinality estimation and the OSPA are considered. Figure 72 shows the cardinality estimation averaged over the Monte Carlo runs for the first target (born at $k = 0$) and the 50th target (born at $k = 4900$). A threshold value of 0.9 is used for track extraction, meaning the average cardinality estimation for the first target reaches this threshold in 3.9 seconds. By adapting the birth density, this detection delay can be reduced to 0.3 seconds for the last target.



(a) first target (b) last target

Figure 72: Cardinality estimation averaged over 100 MC runs [PH7].

The OSPA metric [115] with parameters $p = 1$ and $c = 1$ is used as the second performance indicator. The average OSPA, OSPA Loc, and OSPA Card are calculated for each 10-second interval (corresponding to one target) and are shown in Figure 73.

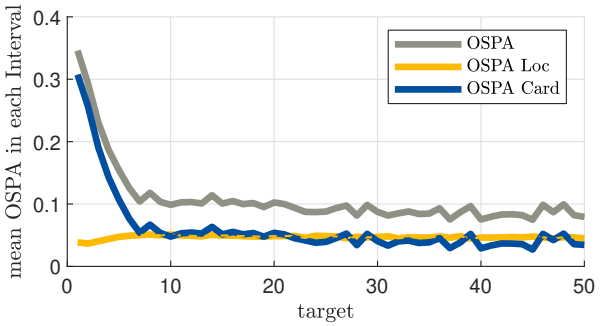


Figure 73: OSPA averaged over 100 MC runs, averaged over each target [PH7].

The Cardinality OSPA is significantly reduced due to the adaptation of the birth density. Interestingly, there is no further improvement after 10 targets (100 seconds), even though the estimation of the birth density takes more time to converge to the true value, as shown in Figure 74. To investigate whether the advantages of the RTS recursion affect the OSPA, the total mean values are calculated and presented in Table 10 and demonstrate that the OSPA could be reduced by 17%.

Table 10: Mean OSPA averaged over 100 MC runs [PH7].

	without RTS	with RTS	Difference
OSPA	0.130	0.108	-17%
OSPA Loc	0.054	0.047	-13%
OSPA Card	0.075	0.061	-19%

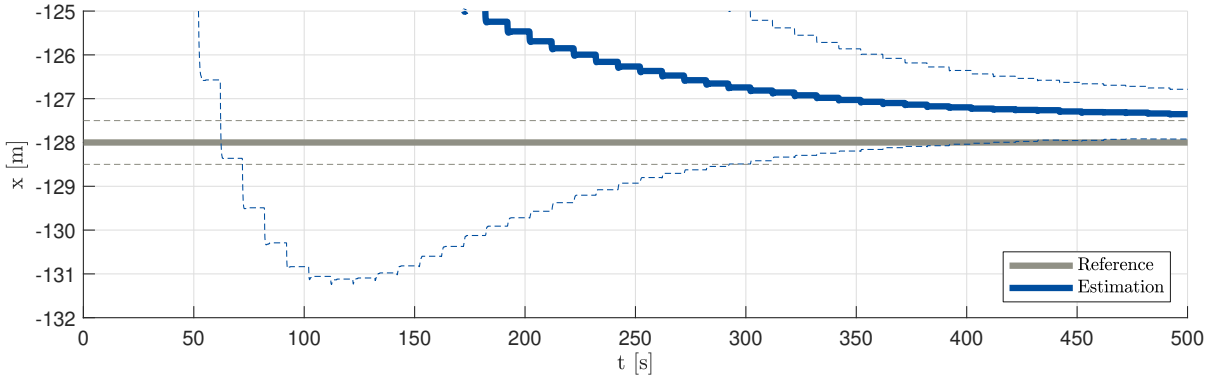


Figure 74: Estimation of $m_{b,x}$ and $\sigma_{b,x}$ averaged over 100 MC runs [PH7].

Initial birth density with small uncertainty Whereas the uncertainty of the initial birth density was chosen too large in the first scenario, it is chosen too small in this scenario. The expected value of the initial birth density is given by:

$$m_{b,\text{filter}} = (-130, 0, 5, 0, 0, 0) \tag{9.5}$$

with the small uncertainties (Figure 70b):

$$\sigma_{b,\text{filter,small}} = (1, 1, 1, 1, 1, 1). \tag{9.6}$$

In this scenario, the targets move away from the birth density and never cross it. The issue is not when they are detected but whether they are detected at all. Without adaptation, it is expected that only targets appearing in the southern part ($y \approx 5\,\text{m}$) of the true birth density will be detected. However, with every detected target, the probability of future targets being detected increases. Figure 75 shows the cardinality estimation averaged over the Monte Carlo runs.

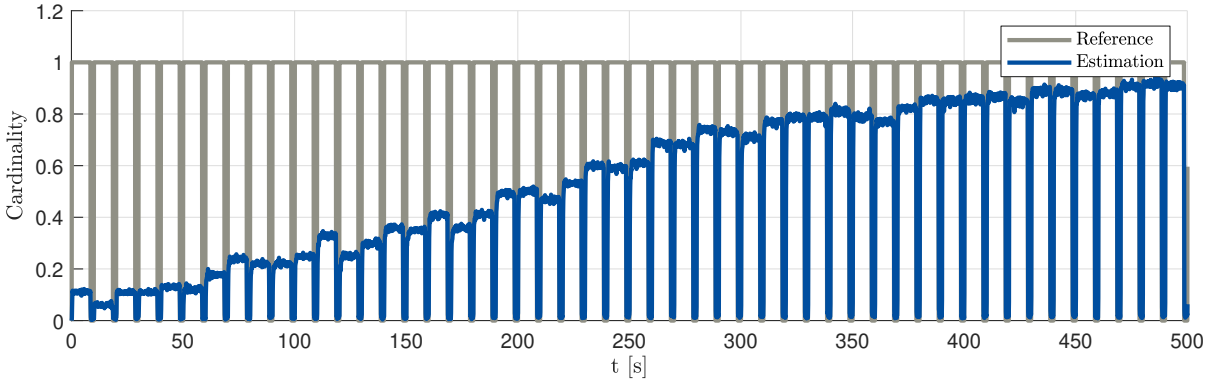


Figure 75: Cardinality estimation averaged over 100 MC runs [PH7].

If a target is successfully detected, its estimated probability of existence is approximately 1 and only decreases briefly for single time steps in the case of a missed detection. The underestimation of the cardinality in Figure 75 is therefore caused by missed targets born too far away from the birth density. The first target is detected in approximately 10% of the Monte Carlo runs, while the last target is detected in approximately 90% of the runs. After 500 seconds, a bias in the y-value of the estimated birth density remains, as shown in Figure 76. It is expected that this bias disappears in longer scenarios. There is also a bias in the x-direction, as shown in Figure 77.

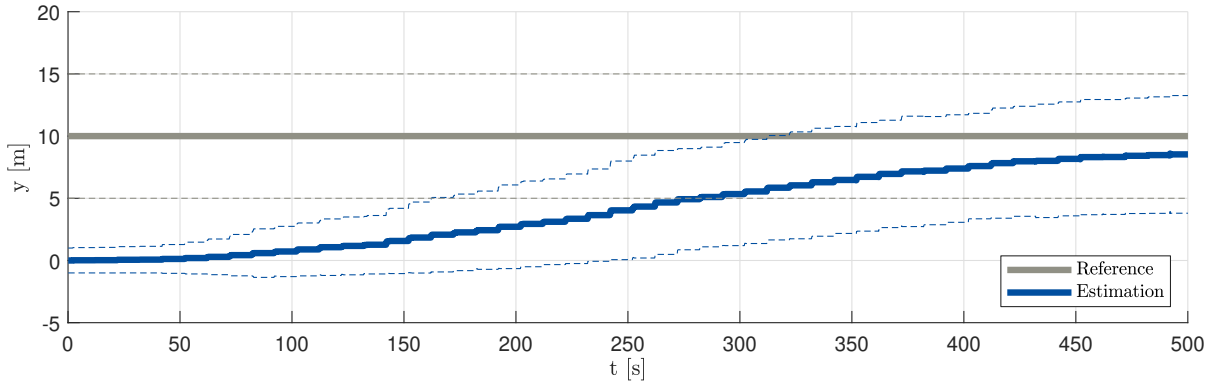


Figure 76: Estimation of $m_{b,y}$ and $\sigma_{b,y}$ averaged over 100 MC runs [PH7].

This bias is not problematic because it is positive, and objects are moving in the x-direction; thus, they will eventually cross the birth density. As explained in the first scenario, this bias will persist and will not disappear.

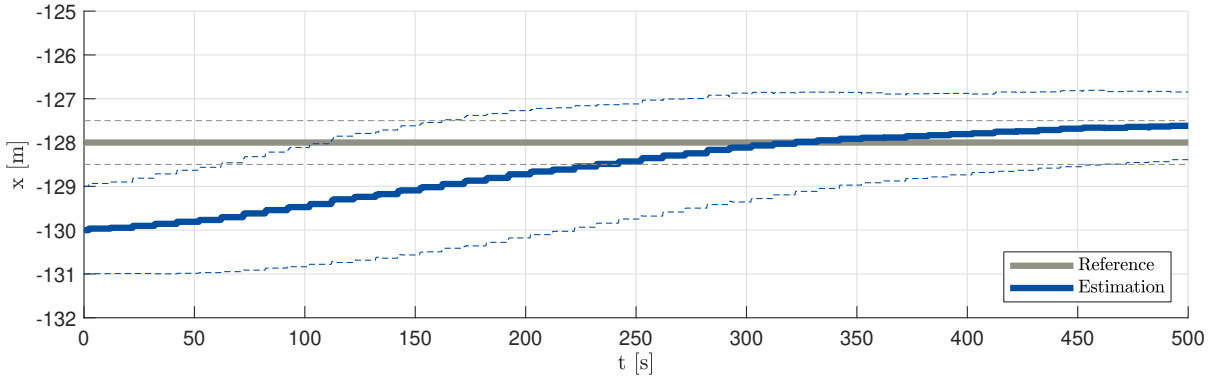


Figure 77: Estimation of $m_{b,x}$ and $\sigma_{b,x}$ averaged over 100 MC runs [PH7].

Limitation A significant limitation of the detection-driven adaptation lies in its reliance on the presence of detections. Adaptation is only possible if detections occur. If the birth density is so poorly placed that no detections are made, the filter has no means to adapt. However, even if the probability of detecting a target is very low, the birth density in the filter will gradually adapt until it aligns with the true birth density of newly born targets.

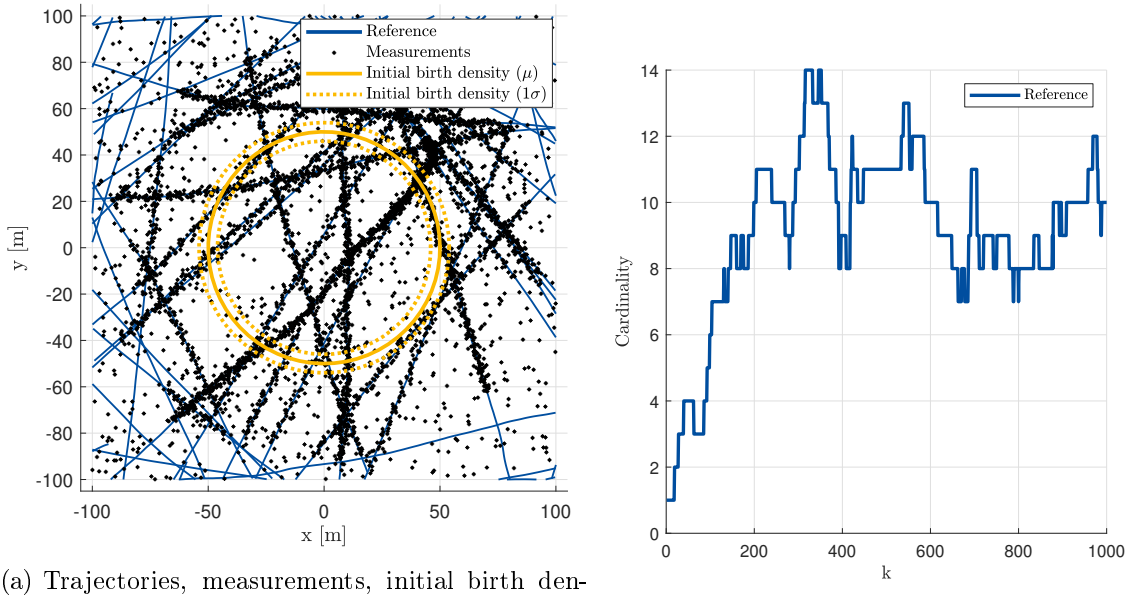
9.2. Circular birth density adaptation

This section marks the first detailed investigation of the circular birth density. The focus lies on a detection-driven adaptation, assuming a static sensor setup and the absence of ego motion. An initial version of this scenario was presented in the master's thesis [M2, p. 68–69], albeit only briefly and without in-depth analysis. For the subsequent conference publication [PH8], the journal article [PH9], and this dissertation, the scenario was revisited with greater detail and with additional insight.

To evaluate the circular detection-driven adaptive birth density using a TPHD filter, a complex scenario involving multiple targets is considered. The sampling period is set to $T = 0.1\,\mathrm{s}$, and over the course of $k = 1000$ time steps, a total of 50 targets are born at uniformly distributed time indices. The surveillance area is defined as a square of size $200\,\mathrm{m} \times 200\,\mathrm{m}$, bounded by $-100\,\mathrm{m} < x, y < 100\,\mathrm{m}$. Newborn targets originate along the border of the surveillance area, with initial positions uniformly distributed across the 800 m perimeter. Targets born on the southern edge have initial velocities sampled from $v_x \sim \mathcal{U}(-10, 10)\,\mathrm{m/s}$ and $v_y \sim \mathcal{U}(0, 20)\,\mathrm{m/s}$, ensuring movement toward the center. Equivalent velocity distributions are applied for targets originating from the remaining edges [M2].

The sensor is positioned at the origin of the coordinate system, and each target is assigned an individual detection range p_r, which is sampled from a uniform distribution as $p_r \sim \mathcal{U}(80, 100)\,\mathrm{m}$. This concept of a variable detection range accounts for differences in object sizes and geometries, which influence the maximum distance at which targets can generate measurements. In the simulation, a target only generates measurements if it is closer to the sensor than its assigned value of p_r. Each target's p_r value is drawn once at the time of birth and remains constant throughout its lifetime. The clutter rate is set to $\lambda = 1$, and all measurements are affected by white Gaussian noise with standard deviations $\sigma_x = \sigma_y = 1\,\mathrm{m}$.

Figure 78a illustrates the reference trajectories, the generated measurements, and the initial circular birth density. The corresponding reference cardinality of the scenario is shown in Figure 78b [M2].



(a) Trajectories, measurements, initial birth density

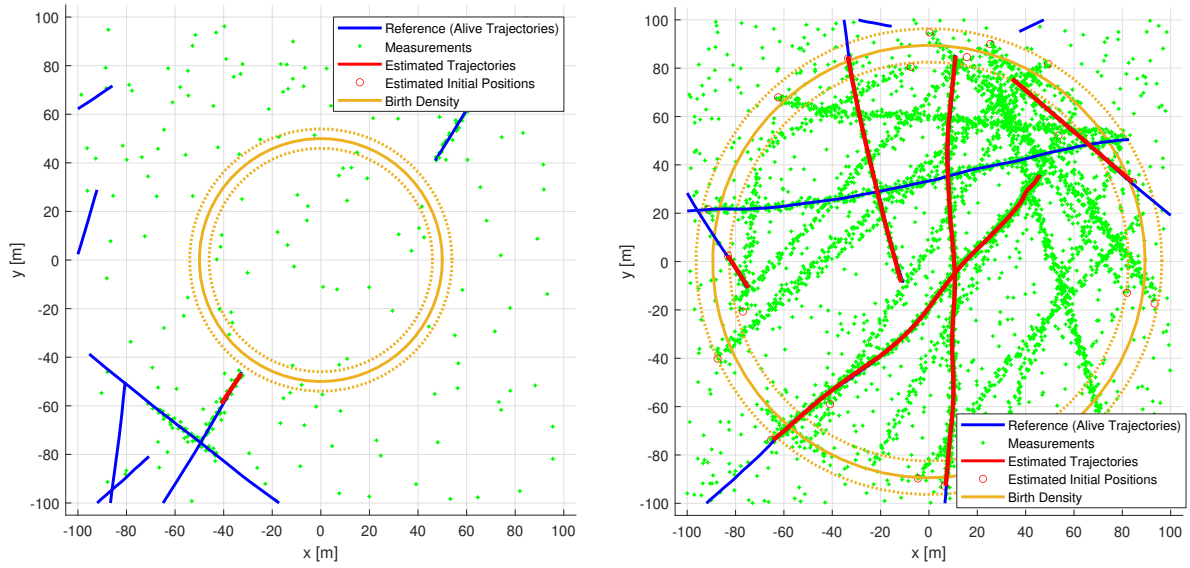
(b) Cardinality

Figure 78: Scenario to demonstrate the adaptation of the circular birth density [PH8, PH9, M2].

For multi-object tracking, a TPHD filter is employed in its Gaussian mixture implementation. Components with weights smaller than 0.01 are pruned, and trajectories that lie within one Mahalanobis distance of a longer trajectory are absorbed.

At the beginning of the scenario, the circular birth density is positioned too close to the sensor,

with an initial radius of $\hat{r} = 50\,\mathrm{m}$ and a standard deviation of $\sigma_r = 4\,\mathrm{m}$. As a result, the first targets are detected only with delays or not at all.
The radial velocity is expected to be negative, as objects move toward the sensor. It is initialized with a value of $v_r = -10\,\mathrm{m/s}$ and a standard deviation of $\sigma_{v_r} = 4\,\mathrm{m/s}$. Due to symmetry, the tangential velocity is assumed to have an expected value of $v_t = 0\,\mathrm{m/s}$, with an initial uncertainty of $\sigma_{v_t} = 2\,\mathrm{m/s}$. This circular birth density is approximated by a mixture of eight Gaussian components for use within the TPHD filter.
Once the TPHD filter has tracked an object for 20 consecutive time steps, the resulting trajectory, of length $\iota = 20$, is used to adapt the birth density based on its initial position. A tuning parameter of $\tau = 3$ is applied for this adaptation. If the sole purpose of estimating the full trajectory is to update the birth density, an L-scan implementation with $L = \iota$ can be used to reduce memory and computation. However, for visualization purposes, the complete trajectory is retained.
In the following, the results of a single realization of the scenario are examined. The first adaptation of the birth density occurs at time step $k = 168$. Figure 79a shows the tracking results at this point.



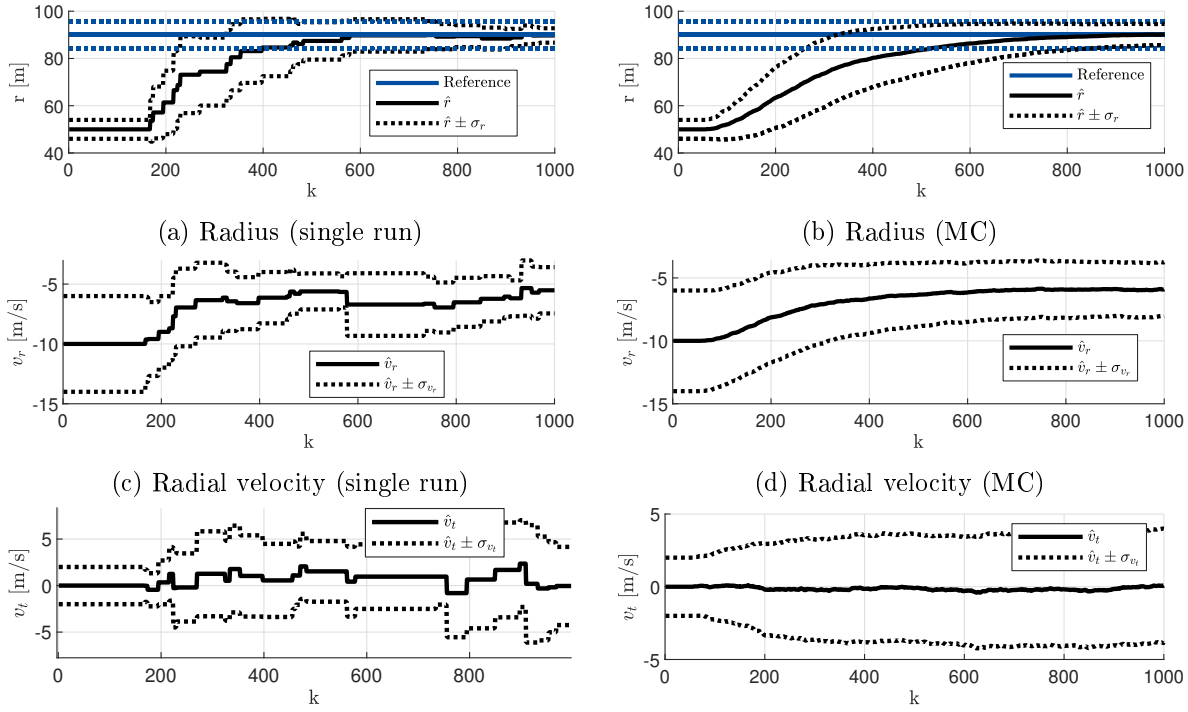
(a) Tracking results at $k = 178$

(b) Tracking results at $k = 850$

Figure 79: Tracking results of the TPHD filter with the circular detection-driven adaptive birth density [PH8, PH9, M2].

The object that was now tracked by the filter for 20 time steps was detected with a delay, as expected. By this point, two other objects had already passed through the surveillance area without being detected by the filter, as they had entered outside the region where new objects were likely to appear according to the initial birth density. The measurements of these two undetected objects can be clearly seen in the top right and bottom left regions of the surveillance area.
For the detected trajectory, the initial state is transformed into polar coordinates, resulting in $r = 71.8\,\mathrm{m}$, $v_r = -7.7\,\mathrm{m/s}$, and $v_t = -0.2\,\mathrm{m/s}$. Based on these values, the birth density is adapted for the first time.
After numerous subsequent adaptations, the birth density converges toward the actual region in which objects are first observed. This result is illustrated in Figure 79b. New objects are now detected immediately. One reference trajectory is not detected anymore, but this target is no longer generating measurements and is therefore no longer considered *alive* by the filter.
The evolution of the birth density's parameters over time is shown in the left column of Figure 80. The final parameter estimates at the end of the scenario are summarized in Table 11. A reference

value is only available for the radius: since the detection range of new targets is uniformly distributed between 80 m and 100 m, the expected value is 90 m. The corresponding standard deviation of the true birth density is given by $\sigma_r = \sqrt{1/12} \cdot 20\,\text{m} = 5.78\,\text{m}$.
So far, only a single run of the scenario has been considered. In the following, a MC simulation with 100 runs is performed. In each run, both the reference trajectories and the measurements are newly generated.
The averaged results of this Monte Carlo evaluation are shown in the right column of Figure 80, and the final values for each parameter are summarized in Table 11.



(a) Radius (single run)

(b) Radius (MC)

(c) Radial velocity (single run)

(d) Radial velocity (MC)

(e) Tangential velocity (single run)

(f) Tangential velocity (MC)

Figure 80: Estimated birth density parameters of a single run in the left column. Mean and standard deviation from 100 MC runs in the right column [PH8, PH9].

Table 11: Results of the birth density estimation [PH8, PH9].

	Reference	Single run	MC runs
r [m]	90.00	89.72	90.14
σ_r [m]	5.78	3.04	4.49
v_r [m/s]	n.a.	−5.52	−5.91
σ_{v_r} [m/s]	n.a.	1.94	2.16
v_t [m/s]	n.a.	−0.04	0.08
σ_{v_t} [m/s]	n.a.	4.20	3.91

The radius is estimated at $r = 90.14$ m, which is very close to the reference value of 90 m. However, the standard deviation is estimated at 4.49 m and is therefore too low compared to the reference value of 5.78 m.
For the radial velocity, a final value of $v_r = -5.91 \pm 2.16$ m/s is determined. This result is consistent with the expectation that the initial radial velocity must be negative.
The final estimate for the tangential velocity is $v_t = -0.08 \pm 3.91$ m/s. This matches the expectation that the mean of the tangential velocity should be close to zero due to symmetry. The large standard deviation reflects the fact that objects enter the surveillance area at a wide range of angles.

9.3. Comparison with other birth models

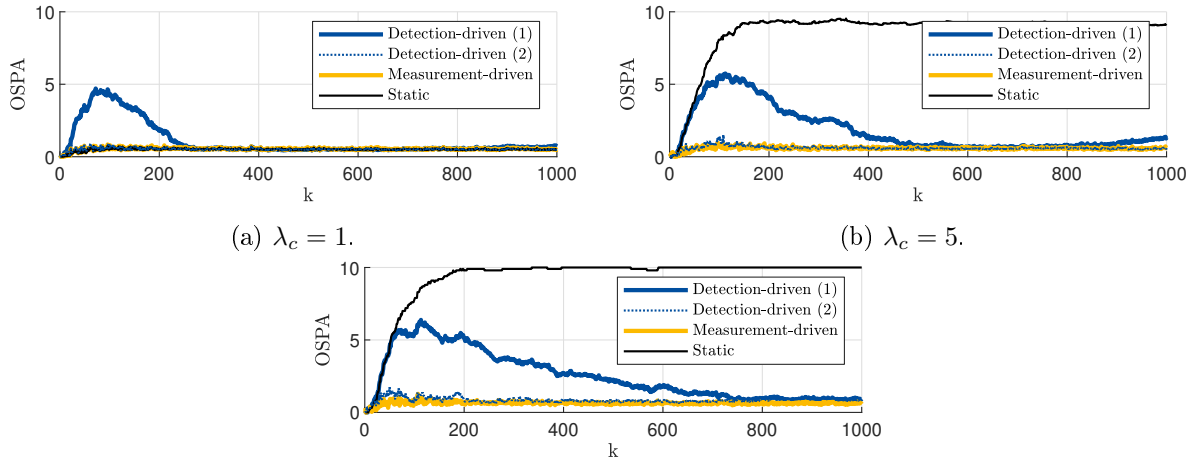
In this section, the detection-driven adaptive circular birth density is compared under two different parameter settings with two alternative birth models. The comparison is based on the same scenario as described in the previous section. The methods being compared are:

- **Detection-driven (1):** The birth density is configured as in the previous section, with an initial expected radius of $\hat{r} = 50\,\mathrm{m}$ and an expected radial velocity of $\hat{v}_r = -10\,\mathrm{m/s}$. For fast adaptation, the parameter $\tau = 3$ is used, consistent with the earlier setup.
- **Detection-driven (2):** In this configuration, the sensor range is assumed to be better known. The initial expected radius is set to $\hat{r} = 80\,\mathrm{m}$, and the expected radial velocity, derived from previous runs, is set to $\hat{v}_r = -5\,\mathrm{m/s}$. Since fast adaptation is no longer required, τ is increased to 20, ensuring more precise stationary behavior.
- **Measurement-driven:** Gaussian birth components with $P_{\mathrm{md}} = \mathrm{diag}(10^2, 10^2, 10^2, 10^2)$ are placed at every location where measurements were received during the previous time step.
- **Static:** A static Gaussian birth density with large uncertainty covers the entire surveillance area. It is given by

$$p_{b,\mathrm{single}}(\underline{x}) = \mathcal{N}\left(\underline{x}; \underline{0}, \mathrm{diag}(100^2, 10^2, 100^2, 10^2)\right),$$

  where the Cartesian state is defined as $\underline{x} = (x, v_x, y, v_y)$.

The OSPA metric is used as the evaluation criterion, and the results are presented in Figure 81 for different clutter intensities.



(a) $\lambda_c = 1$. (b) $\lambda_c = 5$.

(c) $\lambda_c = 10$.

Figure 81: Comparison of different birth models using OSPA. Different clutter intensities are evaluated. Averaged over 100 MC runs.

In the previous chapter, the adaptation behavior of the detection-driven circular birth density was demonstrated by deliberately initializing the radius with a poor estimate. This configuration is labeled as (1) in Figure 81 and is represented by a solid line.
In contrast, if the birth density is initialized with improved parameters ($\hat{r} = 80\,\mathrm{m}$, $\hat{v}_r = -5\,\mathrm{m/s}$) and the adaptation rate is reduced by setting $\tau = 20$, more accurate results are obtained. This configuration is labeled as (2) in Figure 81 and is shown with a dashed line.
At low clutter rates of $\lambda_c = 1$, all approaches perform similarly well, including the single Gaussian birth density with large uncertainty. However, the poorly initialized circular birth density requires a few targets to be observed before it can adapt effectively and detect subsequent targets.

At higher clutter rates ($\lambda_c = 5$, $\lambda_c = 10$), the single Gaussian density fails completely and is unable to detect targets, highlighting the necessity of adaptive birth models.
Interestingly, the results of the measurement-driven birth densities remain very good and even slightly outperform the circular birth densities at higher clutter rates. A significantly increased false-alarm rate was not observed in this scenario, likely due to the low uncertainties of the measurement-driven components and the relatively slow movement of the targets. The slow motion ensures that targets remain close to the expected positions of the birth components.
However, higher target speeds would negatively affect the performance of the circular birth density, as the model assumes targets to appear near the sensor and to approach slowly. The computational effort across all compared approaches was nearly identical, with the static single Gaussian density requiring slightly less computation.
Even though the detection-driven birth density does not quite match the measurement-driven approach in terms of the OSPA metric, it can still be advantageous to use this method. It provides additional contextual information that can be valuable for scene interpretation. For instance, even when no target is currently present, one might act more cautiously if it is known that new targets could appear very close and with high velocity. Conversely, less attention might be required if new targets are expected to appear far away and move slowly. This type of contextual awareness is provided by the birth density estimator.

9.4. Simulation studies with ego motion consideration

In the previous scenario, using the TPHD filter and the detection-driven circular birth density, the sensor remained stationary, and thus no effects of ego motion occurred. In this section, a scenario is examined in which the sensor is mounted on a moving platform following a trajectory as shown in Figure 82. In contrast to the previous setup, the parameters of the circular birth density are assumed to be known, with $\hat{r} = 200$ m and $\sigma_r = 10$ m. They are only adjusted by explicitly accounting for ego motion.
For the radial velocity of other targets, a prior distribution is assumed, given by

$$p\left(v_{r,0}\right) = \mathcal{N}\left(v_{r,0}; 0, 5^2\right). \tag{9.7}$$

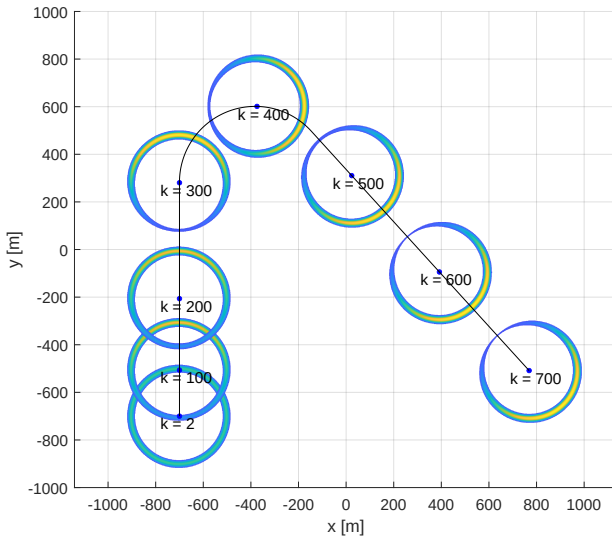


Figure 82: Scenario with ego motion consideration. The birth density is shown for every 100 time steps [PH9].

At the first time step after initialization ($k = 2$), the sensor is stationary ($v_{\text{ego}} = 0\,\text{m/s}$). Consequently, objects are expected to arrive from all directions with equal probability. This is

reflected in Figure 82, where the circular birth density at $k = 2$ exhibits uniform intensity across all directions.
This situation changes as soon as ego motion is present. In addition to the birth probability, it is also illustrative to examine the initial expected velocities of newly appearing targets. These are summarized in Table 12.

Table 12: Parameters of the birth density with ego motion consideration [PH9].

time step k	2	100	200	300	400	500	600	700
Ego motion velocity v_{ego} [m/s]	0.0	3.0	3.0	5.1	5.1	5.1	5.6	5.6
Ego motion orientation θ_{ego} [°]	90	90	90	90	−2	−48	−48	−48
expected velocity v_x for targets in front [1][m/s]	0.0	0.0	0.0	0.0	1.4	−0.9	−0.8	−0.8
expected velocity v_y for targets in front [1][m/s]	−4.0	−2.3	−2.3	−1.4	0.0	1.0	0.9	0.9
expected velocity v_x for targets from behind [1][m/s]	0.0	0.0	0.0	0.0	7.7	5.2	5.5	5.5
expected velocity v_y for targets from behind [1][m/s]	4.0	6.1	6.1	7.7	−0.2	−5.7	−6.0	−6.0

[1] Expected value of the Gaussian component representing the section of the circular birth density located directly ahead of or behind the ego vehicle. In global coordinates, not relative to ego motion.

Without ego motion, the expected radial velocity from the truncated normal distribution (8.35) simplifies to an expression that is independent of the angle θ:

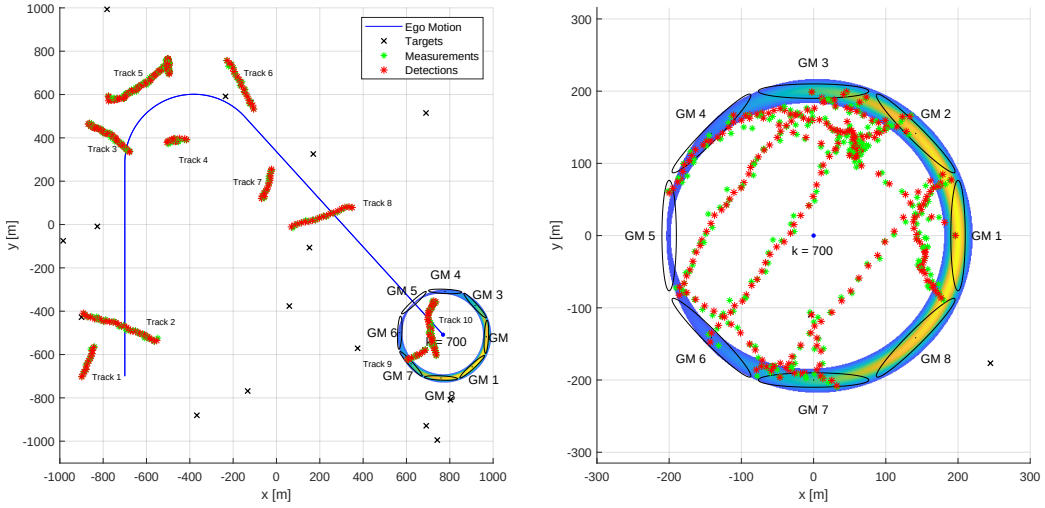
$$\hat{v}_r = \hat{v}_{r,0} - \sigma_{v_{r,0}} \frac{\phi\left(\frac{-\hat{v}_{r,0}}{\sigma_{v_{r,0}}}\right)}{\Phi\left(\frac{-\hat{v}_{r,0}}{\sigma_{v_{r,0}}}\right)}, \tag{9.8}$$

and inserting the values $\hat{v}_{r,0} = 0\text{m/s}$ and $\sigma_{v_{r,0}} = 5\text{m/s}$ from the prior distribution leads to:

$$\hat{v}_r = 0 - 5 \cdot \frac{0.4}{0.5}\text{m/s} = -4 \text{ m/s}. \tag{9.9}$$

Considering an ego orientation of $\theta_{\text{ego}} = 90°$, the Cartesian velocities of the Gaussian mixture components can be calculated. For 2 out of the 8 components, the resulting velocities are listed in Table 12.
Targets that appear in front of the sensor exhibit a negative y-velocity, while those appearing behind the sensor exhibit a positive y-velocity. In both cases, the targets move toward the sensor, which is a prerequisite for target birth.
At higher ego velocities v_{ego}, the required velocity of targets appearing behind the sensor increases, as they must always move faster than the sensor itself. In contrast, in front of the sensor, not only targets moving toward the sensor but also slower targets moving in the same direction as the ego motion can be detected. As a result, the expected velocities in front decrease.

Ego motion consideration within multi-object tracking To investigate the effects of the circular birth density with ego motion consideration in the context of MOT, the following tracking scenario is defined based on the ego motion trajectory from Figure 82:
A total of 100 targets are generated, each with a birth time step sampled uniformly over $k_{\max} = 700$. The initial positions and velocities are generated similarly to Section 9.2, but now from the edges of a square surveillance area of size 2000 m×2000 m, bounded by $-1000\ \text{m} < x, y < 1000\ \text{m}$. The initial velocities follow the same distributions as in Section 9.2; for instance, targets born on the southern edge are sampled with $v_x \sim \mathcal{U}(-10, 10)$ m/s and $v_y \sim \mathcal{U}(0, 20)$ m/s, ensuring motion toward the center of the area.
The transition model, process noise, and death condition are also equivalent to Section 9.2. Each target has a detection range sampled as $p_r \sim \mathcal{U}(200, 210)$ m, and measurements are affected by white Gaussian noise with standard deviations $\sigma_x = \sigma_y = 1$ m. To isolate the effects on track initialization, no clutter is introduced in this initial investigation, as each target is assumed to be detected upon its first measurement.
A PHD filter in its Gaussian mixture implementation is used for this scenario. Since the birth density is not adapted during the simulation, a trajectory filter is not required. The circular birth density is approximated by a Gaussian mixture with 8 components. Figure 83 shows the tracking results at the final time step $k_{\max} = 700$, both in local and sensor coordinate systems.

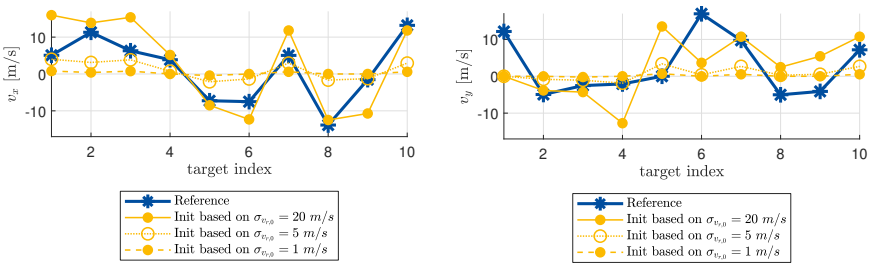


(a) Local coordinates (b) Sensor coordinates

Figure 83: Targets and birth density at the last time step $k_{\max} = 700$, measurements and filter's detection of all time steps [PH9].

Out of the 100 generated targets, only 10 produced measurements, as most targets remained outside the sensor's detection range throughout the scenario. As expected for this relatively simple setup, all targets that generated measurements were successfully tracked.
However, the advantages of ego motion consideration become evident when analyzing the initialization of the tracks in more detail. Figure 84 illustrates the actual velocities (v_x, v_y) of the targets at the time of track initialization, alongside the velocities used for the initialization based on the birth model.

Track 1 (see Figure 83a) approaches the sensor from the left. Consequently, its x-velocity must be positive, and the expected value of the resulting truncated distribution depends on the assumed prior velocity distribution. With the chosen value of $\sigma_{v_{r,0}} = 5$ m/s, the initialized x-velocity of the first target matches the reference velocity well.
Since the object approaches from the side, there is no directional constraint on the y-velocity. Due to symmetry, the expected value remains zero. This means that for the first target, only



(a) Velocities in x-direction (b) Velocities in y-direction

Figure 84: Reference velocities of targets and initialization velocities [PH9].

the initialization of the x-velocity benefits from ego motion consideration.
Across the entire scenario, the average root mean square error (RMSE) in both x- and y-velocity is lowest—5.5m/s—when $\sigma_{v_{r,0}} = 5$m/s is assumed. Assuming narrower priors, such as $\sigma_{v_{r,0}} = 2$ m/s or $\sigma_{v_{r,0}} = 1$ m/s, yields slightly higher RMSE values of 6.1 m/s and 6.7 m/s, respectively.
In all cases, the performance is worse when using a naïve initialization with zero velocity, which results in an RMSE of 7.0 m/s.

Monte Carlo runs For more conclusive results, a Monte Carlo simulation is conducted for the same scenario. The ego motion trajectory remains constant across all runs, while the target trajectories and their corresponding measurements are resampled in each run.
For different assumed prior densities of the initial radial velocities, each with expected value 0, but varying standard deviations $\sigma_{v_{r,0}}$, a total of 100 runs is simulated per configuration. The resulting RMSE values are summarized in Table 13 and compared against a baseline where velocity is initialized with zero.
The best result was obtained using the prior $\mathcal{N}(0, 10^2)$, yielding an average RMSE of 4.6m/s. Compared to zero initialization, this corresponds to a reduction of 31 %. However, if the prior uncertainty is set too high—implying unrealistically fast targets, the RMSE increases substantially, as seen in the entry for $\sigma_{v_{r,0}} = 50$m/s.

Table 13: RMSE for different assumed prior densities for the initial velocity

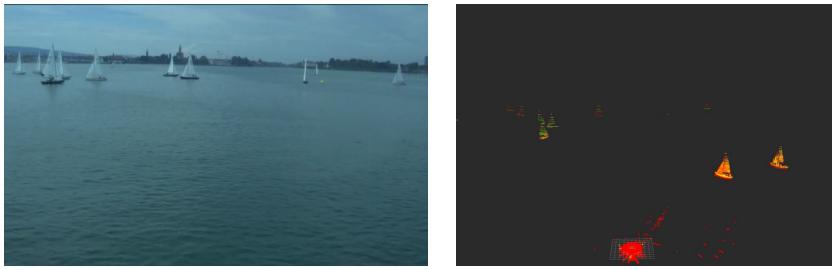
$\sigma_{v_{r,0}}$ [m/s]	0	1	2	5	10	15	20	50
RMSE	6.7	6.4	6.2	5.4	4.6	5.0	6.4	20.0
[%]		−4	−8	−19	−31	−25	−4	+201

9.5. Full-scale experiments

In the final results section, the proposed approach is evaluated on real lidar data recorded in a maritime scenario on Lake Constance. Figure 85 shows the sensor equipment used during the experiment. For the determination of the ego motion, the global positioning system (GPS) and inertial measurement unit (IMU) were used.

Figure 85: Sensor equipment: Velodyne VLS-128 lidar Alpha Prime, stereo cameras, GPS and IMU mounted on the research vessel Solgenia.

Figure 86a shows an image from one of the onboard cameras, while Figure 86b displays the measurements generated by the lidar sensor at the beginning of the scenario.



(a) Camera output (b) Lidar output

Figure 86: Sensor output at the beginning of the scenario.

In this experiment, the sensor equipment was mounted on a catamaran operating as public transport between Friedrichshafen and Constance. On its way to Constance, the catamaran encountered and crossed a large group of sailing boats participating in a regatta.
The lidar measurements were clustered using density-based spatial clustering of applications with noise (DBSCAN) [87], with a *distance* parameter of $\epsilon = 10$m and a *minimum number of measurements* set to 1. All measurements within 20m of the sensor were discarded, as they originated from the ship itself. Figure 87 shows the resulting clustered measurements in sensor coordinates.
The tracks of the objects are clearly recognizable in the measurements. They typically begin at a distance of approximately 250 m, which corresponds to the sensor's maximum range. Clutter measurements are almost absent at larger distances, but become more frequent at close range, particularly within 50 m in front of the sensor, where they are primarily caused by wave reflections. Using different DBSCAN parameter settings can reduce the number of clutter measurements. However, the tracking filter is also capable of suppressing these spurious detections effectively.
The total duration of the scenario is 1200 time steps, with a sampling period of $T = 0.1$ s.
For the multi-object tracking, a PHD filter is employed, assuming a CV motion model and an object dynamic model (ODM) for all targets. The process noise is set to $\sigma_a = 0.2$ m/s^2, and the measurement noise to $\sigma_x = \sigma_y = 2$ m. The reason for this high noise value is not the noise in individual lidar measurements, but rather that the centers of the clusters behave like noisy measurements of the object's center of gravity. The detection probability is $p_d = 0.9$, and the clutter rate is $\lambda_c = 1$. The circular birth density is approximated using 8 GM components. For the prior velocity distribution of newly appearing targets, a truncated Gaussian with $v_{r,0} \sim \mathcal{N}(0, \sigma^2_{v_{r,0}})$ and $\sigma_{v_{r,0}} = 5$ m/s is assumed. Figure 88a shows the tracking results at time step $k = 370$.
Three tracks have been successfully maintained over the last few time steps, and a fourth track is currently being initialized between birth components 3 and 4. Clutter measurements near the sensor have not caused any false alarms.
However, two major issues must be addressed. First, the filter failed to track two targets. One of the missed targets, located in the lower-right corner, was initially too close to the sensor and thus too far from all birth components. The second missed target was located near GM 4. For this target, no measurements were generated at a distance of 250 m, and they only appeared later in the scenario. The second issue concerns false alarms: in addition to the missed targets, four false tracks were initialized in regions without any measurements.
Figure 88b shows the tracking results at time step $k = 1000$, where the previously observed issues repeat. While the majority of targets are successfully tracked, a number of tracks are missed and false alarms are generated in measurement-free regions.
Most of the missed tracks are caused by lidar shadowing on the left-hand side, which is already clearly visible in the measurements shown in Figure 87. If targets emerge from the shaded area, the circular birth density filter is no longer able to track them. The missed detection hypothesis

cannot handle this either, as the period of invisibility due to shading is too long and leads to eventual track deletion.
To address this, the shadowing would need to be explicitly modeled, e.g. by placing additional birth components in shaded regions, or the approach would need to be extended using a combination with a measurement-driven adaptive birth density.

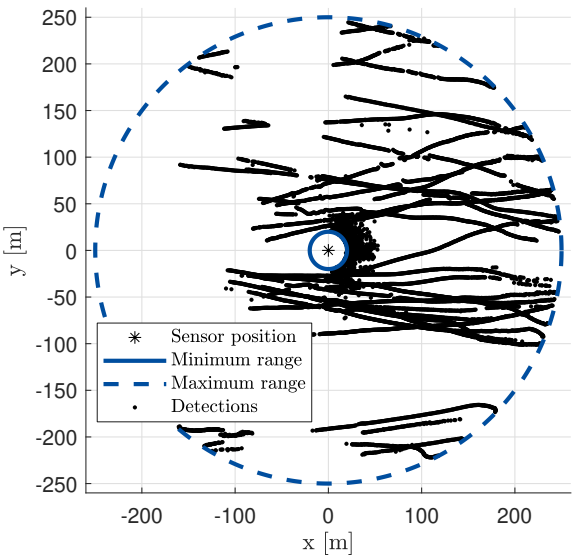
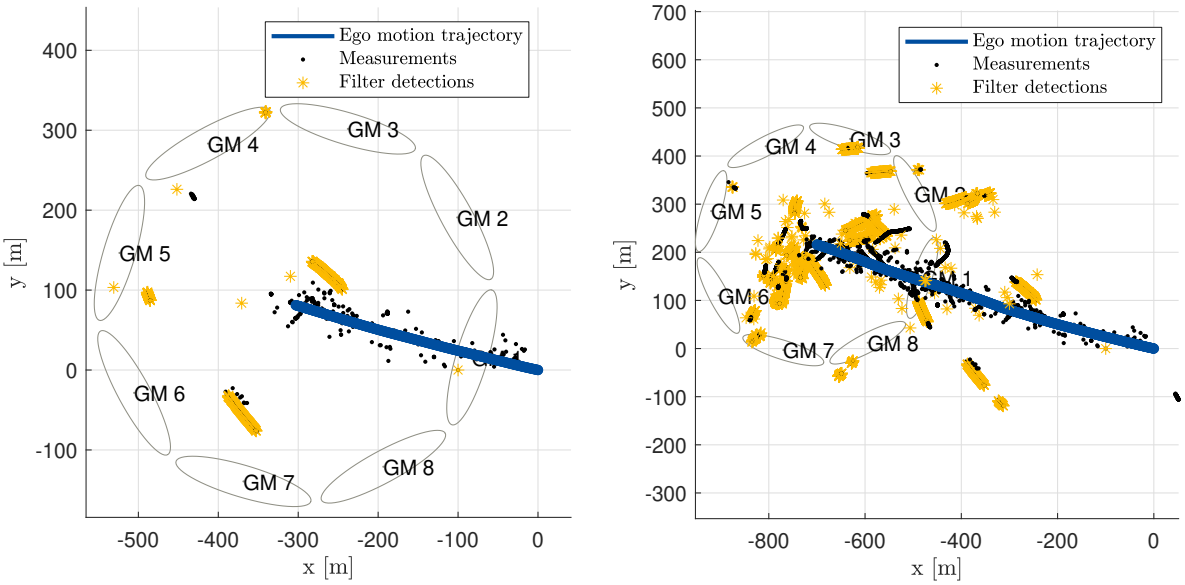


Figure 87: Measurements in the sensor coordinate system [PH9].



(a) $k = 370$ (b) $k = 1000$

Figure 88: Tracking results [PH9].

9.6. Critical interpretation of the results

The simulation results in Section 9.2 demonstrated that, particularly in the presence of unknown parameters, the adaptation of the birth density is successful. It leads to improved filter performance, as tracks can be detected more reliably and at earlier stages.
By considering ego motion in Section 9.4, the initial velocity of newly appearing objects could be initialized more accurately, resulting in up to 31 % lower RMSE, as shown in Table 13. However, this improvement had no significant effect on evaluation metrics such as the OSPA score or position error, since even with zero velocity initialization (0m/s), all tracks were reliably established. The influence of initialization vanished after a few time steps.
In contrast, the consideration of the sensor's own position proved to be indispensable, as it enabled consistent detection of tracks throughout the entire scenario.
The real-data experiment presented in Section 9.5, however, revealed the most critical limitation of the approach: If a track is lost during the scenario, e.g. due to lidar shadowing, the circular birth density is too inflexible to support reinitialization of that track.
Moreover, in scenarios with extremely low clutter rates, the circular birth density offers no advantage over a measurement-driven birth model. Since nearly every measurement originates from an actual object, a measurement-driven approach does not significantly increase the false alarm rate or the computational burden.
Figure 89 illustrates the results of the same tracking scenario with a few modifications: DBSCAN now requires at least 10 measurements per cluster. Instead of a PHD filter, a simple nearest-neighbor measurement association followed by a Bernoulli update is employed. Additionally, the circular birth density is replaced by a measurement-driven model. Under these conditions, no tracks are missed, and only a few false alarms remain, mostly caused by clutter measurements.

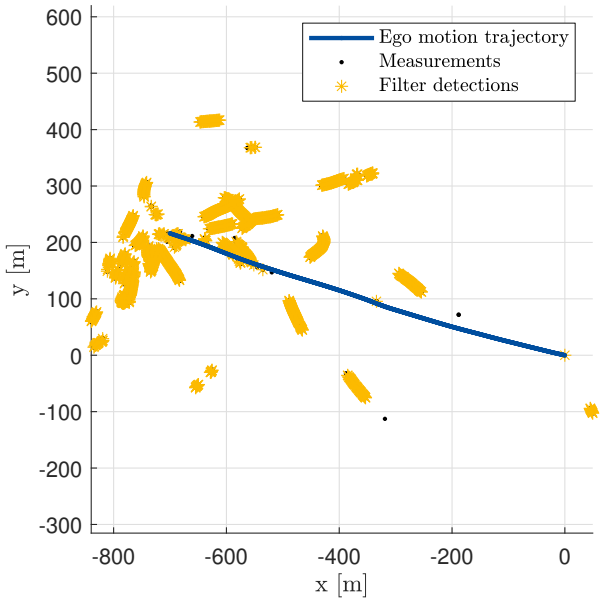


Figure 89: Tracking results at $k = 1000$ with a measurement-driven adaptive birth model and nearest neighbor measurement assignment [PH9].

9.7. Conclusion for adaptive birth densities

It was shown that a detection-driven adaptation of a Gaussian birth density can improve tracking performance in scenarios with limited knowledge of the initial positions of targets. However, the restriction to a single Gaussian component makes this approach applicable only in specific cases, such as tracking vessels on a river. For broader applications, particularly with lidar sensors in open areas, a circular birth density was introduced in polar coordinates, incorporating all point-symmetric quantities such as radial and tangential velocity. For Cartesian filter implementations, a Gaussian mixture approximation and a corresponding coordinate transformation were derived. When employing a multi-object tracking filter based on sets of trajectories, such as the TPHD filter, the estimated initial positions of newly detected trajectories can be used to adapt the parameters of the birth density. Simulation results demonstrated that this adaptation enables faster and more reliable detection of future targets. In direct comparison to a measurement-driven birth model, the detection-driven approach achieved comparable overall performance while additionally providing spatial information on likely target appearances, which may support scene interpretation.
It was further shown that ego motion alters both the birth probability and the birth density. In such cases, new objects must exhibit an initial velocity component towards the sensor to be considered valid births, which effectively truncates the velocity distribution. Incorporating this effect reduced the RMSE of the initial velocity estimate by up to 31% in simulation, although no significant improvement was observed in global performance measures such as the average OSPA, even under challenging clutter and velocity conditions.
Experiments with real data revealed specific limitations of the circular birth density, for instance in scenarios involving occlusions, which require additional modeling. For practical large-scale applications, further refinements are necessary, including the consideration of blind-spot regions and concealment effects. In low-clutter, low-noise environments with simple clustering, however, a purely measurement-driven adaptation may remain advantageous.

A. Derivation of the constant acceleration model

As an example for physics-based modeling for the kinematic state evolution, the discretized constant acceleration ODM is derived in this section. Additional to the differential equations $\dot{x} = v$ and $\dot{v} = a$, it is assumed that an input disturbance to the constant acceleration model is a change in acceleration, leading $\dot{a} = w$. This can be written in the form $\dot{\underline{x}}(t) = A\underline{x}(t) + B\underline{u}(t)$:

$$\begin{pmatrix} \dot{x} \\ \dot{v} \\ \dot{a} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ v \\ a \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} w. \tag{A.1}$$

The discrete time system $\underline{x}_k = F\underline{x}_{k-1} + Gu_k$ is derived by using

$$F = e^{AT}, \tag{A.2}$$

which is defined by a Taylor expansion:

$$e^{AT} = I + AT + A^2\frac{T^2}{2!} + A^3\frac{T^3}{3!} + ... \tag{A.3}$$

For the CA model, this leads to:

$$F = e^{AT} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & T & 0 \\ 0 & 0 & T \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & \frac{T^2}{2} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & T & \frac{T^2}{2} \\ 0 & 1 & T \\ 0 & 0 & 1 \end{pmatrix}. \tag{A.4}$$

Expressions with A^3 and higher exponents, turn to zero. Additionally, we have to calculate a discretization G of the input matrix B using:

$$G = \int_0^T e^{A\tau} d\tau B = \left(IT + A\frac{T^2}{2!} + A^2\frac{T^3}{3!} + ... \right) B. \tag{A.5}$$

For the CA model, this leads to:

$$G = \left(\begin{pmatrix} T & 0 & 0 \\ 0 & T & 0 \\ 0 & 0 & T \end{pmatrix} + \begin{pmatrix} 0 & \frac{T^2}{2} & 0 \\ 0 & 0 & \frac{T^2}{2} \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & \frac{T^3}{6} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \right) \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{T^3}{6} \\ \frac{T^2}{2} \\ T \end{pmatrix}. \tag{A.6}$$

To calculate the covariance matrix Q, there are several approaches. In this work, all simulations follow the approach given by Vo

$$Q = \sigma_w^2 BB^\top = \sigma_w^2 \begin{pmatrix} \frac{T^6}{36} & \frac{T^5}{12} & \frac{T^4}{6} \\ \frac{T^5}{12} & \frac{T^4}{4} & \frac{T^3}{2} \\ \frac{T^4}{6} & \frac{T^3}{2} & T^2 \end{pmatrix} = \sigma_w^2 T^2 \begin{pmatrix} \frac{T^4}{36} & \frac{T^3}{12} & \frac{T^2}{6} \\ \frac{T^2}{12} & \frac{T^2}{4} & \frac{T}{2} \\ \frac{T^2}{6} & \frac{T}{2} & 1 \end{pmatrix}. \tag{A.7}$$

However, the most common result for Q in the literature is

$$Q = \sigma_w^2 T \begin{pmatrix} \frac{T^5}{20} & \frac{T^4}{8} & \frac{T^3}{6} \\ \frac{T^4}{8} & \frac{T^3}{3} & \frac{T^2}{2} \\ \frac{T^3}{6} & \frac{T^2}{2} & T \end{pmatrix} = \sigma_w^2 T^2 \begin{pmatrix} \frac{T^4}{20} & \frac{T^3}{8} & \frac{T^2}{6} \\ \frac{T^3}{8} & \frac{T^2}{3} & \frac{T}{2} \\ \frac{T^2}{6} & \frac{T}{2} & 1 \end{pmatrix}, \tag{A.8}$$

which is therefore stated in the background section. However, especially for small values of T, differences can be neglected.

Index

3D tracking, 24, 36–38, 42, 75–78

adaptation
  algorithm, 40–46, 51, 52, 57
  law, 40, 41, 45
adaptive birth density, 12, 83–87, 97, 101–110
  detection-driven, *see* detection-driven adaptive birth density
artificial measurements, *see* virtual measurements
automotive, 24, 67, 68

Bayes filter, 7
Bernoulli
  filter, 11, 83, 116
  random finite set (RFS), 11
Bernoulli filter, 12
Bienaymé identity, 88
birth density, 12, 83–116
  adaptation, 85–87, 97, 98, 102, 103, 107, 116
  adaptive, *see* adaptive birth density
  circular, *see* circular birth density
  detection-driven adaptive, *see* detection-driven adaptive birth density
  measurement-driven, 109, 110, 116
  static, 109, 110
birth intensity, 15, 83, 84, 97, 98
boat, 23, 24, 35, 60, 61, 76–78
  motor, *see* motor boat
  sailing, *see* sailing boat
BODY frame, 30, 31
  for 2D case, *see* target coordinate system
bus, 24

car, 24
cardinalized probability hypothesis density (CPHD) filter, 84
Cartesian approximation, 93–96
center of gravity, 6, 7, 37, 38, 40, 41, 55, 65
Chamfer distance, 51, 52, 69, 74–78
circular birth density, 89–100, 106–116
classification, 24, 51–53, 69, 70, 73–78
closed loop stability, 40–42
cone, 23, 24, 32, 33, 74–77
  Cartesian representation, 32
  lateral surface, 32
constant acceleration (CA), 1, 5, 6, 62, 69, 74, 119
constant velocity (CV), 5, 46, 65, 97
contour
  measurements, 6, 23–29, 31–34, 39, 55, 59–63, 65, 66, 69, 70, 74–78
coordinate transformation, 26, 27, 30–32
cuboid, 23, 24, 33, 74–77
cumulative distribution function (cdf), 99
cylinder, 23

data association, 25, 44–46
detection
  probability, 15, 24, 26, 84
  range, 99, 106, 108, 112
detection-driven adaptive birth density, 83, 85–87, 97, 98, 101–110
dot product kernel, 19, 49, 50

ego motion, 26, 37, 76, 98, 99, 110–113, 116
eigenvalue, 8, 36–38, 40–43, 45, 48, 51, 57
eigenvector, 37, 38
eigenvector-based matching, 36, 42
ellipse, 23–28, 34, 39, 55, 58, 61–63, 65, 66, 69, 73
ellipsoid, 23, 24, 30, 32, 74, 75, 77, 78
ENU (x-East, y-North, z-Up), 30–33
error
  norm, 38, 43, 58
  vector, 38, 40

ferry, 24
finite set statistics (FISST), 10

Gaussian
  distributed measurements, 27
  distribution, 15, 18, 19, 27, 44, 83, 91, 94
  mixture, 86, 89–94, 97, 106, 111, 112
    approximation, 90–93
  processes, 19, 20, 47–50, 58, 59, 61–63
    regression model (GPRM), 47–50, 62, 63

inverse Wishart density, 8

Kalman filter, 7–9, 36, 37, 40, 42–47, 49, 51, 55, 57, 58, 62–66, 70, 74, 77, 85, 86
Kronecker product, 5, 6, 18
Kullback-Leibler divergence (KLD), 15, 95

L-scan, 107
labeled multi-Bernoulli (LMB) filter, 11, 84, 85, 101

lidar, 6, 7, 23–35, 39, 55, 59–63, 65, 66, 69, 70, 73, 75–78, 106, 113, 116
  measurements, *see* contour measurements
local coordinate system, 26, 112

maritime, 24, 35, 60, 61, 63, 67, 68, 76–78, 101–103, 105, 113–116
Markov process, 5
measurement
  -driven adaptive birth density, 83, 84
  likelihood, 6, 7, 15, 23–25, 34–36, 40, 71
  model, 6, 8, 9, 23–35, 39
  source, 24, 26, 27, 31, 33, 34
Monte Carlo simulation, 56–59, 63, 70, 73–75, 102–105, 108, 109, 113
motor boat, 24, 77, 78
Murty's algorithm, 25

Nelder-Mead, 38

Object dynamic model (ODM), 5
occlusion, 34, 65, 66
optimization problem, 38, 40
orientation angle, 37, 38
OSPA, 110, 116

parametric representation, 32
physics-based modeling, 5, 24
pitch, 38
pitch angle, 30, 31, 36, 37
point-object, 24
Poisson distribution, 25
Poisson multi-Bernoulli mixture (PMBM), 84
Poisson point process, 84
polar coordinates, 34, 107
probability density function, 100
probability hypothesis density (PHD) filter, 15, 17, 19, 83, 84, 112, 114

radial velocity, 99
random finite set, 10
random matrix, 8, 25, 35, 36, 39, 40
Rauch-Tung-Striebel (RTS) recursion, 85, 86, 102
ray tracing, 23, 25–32, 34
rectangle, 23, 24, 29, 30, 69, 70, 73
reference model concept, 36–38
reflection centers, 24
regression, 19, 20
rejection sampling, 27
roll angle, 30, 31, 36–38
root mean square error (RMSE), 58, 59, 62, 71, 73, 113, 116
rotation matrix, 25–27, 31, 32, 94, 96

sailing boat, 24, 76, 77
sensor coordinate system, 26–30
set of points on a rigid body (SPRB), 24
Singer's model, 6, 37
spatial distribution, 24
spatial model, 24
spherical coordinates, 30
squared exponential kernel, 19
stability, 40
star-convex, 23
state transition, 5

target
  birth, 111
  coordinate system, 26, 28–30, 32, 33, 52, 58, 71, 72, 78
track initiation, 84
trajectory, 86
  probability hypothesis density (TPHD) filter, 17–19, 106, 107
  pruning and absorption, 106
triangle, 24, 28, 69, 70, 73

uniformly distributed measurements, 27, 29, 30

vessel, *see* boat
virtual
  measurement model (VMM), 35–53, 55–61, 65–70, 73, 75–78
  measurements, 25, 35–39, 45–52, 73, 78
  plant, 36, 38, 39

yaw angle, 26, 27, 30, 31, 37, 38, 76

References

Own publications

[PH1] Patrick Hoher, Stefan Wirtensohn, Tim Baur, Johannes Reuter, Felix Govaers, and Wolfgang Koch. "Extended target tracking with a Lidar sensor using random matrices and a virtual measurement model". In: *IEEE Transactions on Signal Processing* 70 (2022), pp. 228–239. DOI: 10.1109/TSP.2021.3138006.

[PH2] Patrick Hoher, Johannes Reuter, Felix Govaers, and Wolfgang Koch. "Joint parameter estimation and trajectory tracking of bounding boxes". In: *2021 IEEE 24th International Conference on Information Fusion (FUSION)*. 2021, pp. 1–8. DOI: 10.23919/FUSION49465.2021.9626855.

[PH3] Patrick Hoher, Johannes Reuter, Felix Govaers, and Wolfgang Koch. "Tracking of partially visible elliptical objects with a Lidar sensor using random matrices and a virtual measurement model". In: *2022 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. 2022, pp. 1–6. DOI: 10.1109/SDF55338.2022.9931696.

[PH4] Patrick Hoher, Johannes Reuter, Daniel Dold, Dennis Griesser, Felix Govaers, and Wolfgang Koch. "Extended target tracking with a Lidar sensor using random matrices and a Gaussian processes regression model". In: *2023 26th International Conference on Information Fusion (FUSION)*. 2023, pp. 1–8. DOI: 10.23919/FUSION52260.2023.10224159.

[PH5] Patrick Hoher, Johannes Reuter, Felix Govaers, and Wolfgang Koch. "Extended object tracking and shape classification using random matrices and virtual measurement models". In: *2023 IEEE Symposium Sensor Data Fusion and International Conference on Multisensor Fusion and Integration (SDF-MFI)*. 2023, pp. 1–8. DOI: 10.1109/SDF-MFI59545.2023.10361348.

[PH6] Patrick Hoher, Tim Baur, Johannes Reuter, Dennis Griesser, Felix Govaers, and Wolfgang Koch. "3D-Extended object tracking and shape classification with a Lidar sensor using random matrices and virtual measurement models". In: *2024 27th International Conference on Information Fusion (FUSION)*. 2024, pp. 1–8. DOI: 10.23919/FUSION59988.2024.10706411.

[PH7] Patrick Hoher, Tim Baur, Stefan Wirtensohn, and Johannes Reuter. "A detection driven adaptive birth density for the labeled multi-Bernoulli filter". In: *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*. 2020, pp. 1–8. DOI: 10.23919/FUSION45008.2020.9190532.

[PH8] Patrick Hoher, Tim Baur, Johannes Reuter, Felix Govaers, and Wolfgang Koch. "A circular detection driven adaptive birth density for multi-object tracking with sets of trajectories". In: *2022 25th International Conference on Information Fusion (FUSION)*. 2022, pp. 1–8. DOI: 10.23919/FUSION49751.2022.9841312.

[PH9] Patrick Hoher, Tim Baur, Johannes Reuter, Felix Govaers, and Wolfgang Koch. "Circular detection-driven adaptive birth densities". In: *Journal of Advances in Information Fusion* 20.1 (June 2025), pp. 46–67. ISSN: 1557-6418.

Own master theses

[M1] Patrick Hoher. "Tracking von Punkt-Objekten mit Bernoulli-Filtern". Elektrische Systeme. Master's Thesis. Konstanz, Germany: HTWG Konstanz, 2019.

[M2] Patrick Hoher. "Tracking of pedestrians using a probability hypothesis density filter". International Project Engineering. Master's Thesis. Konstanz, Germany: HTWG Konstanz, 2022.

Publications as contributing author

[C1] Hannes Homburger, Stefan Wirtensohn, Patrick Hoher, Tim Baur, Dennis Griesser, Moritz Diehl, and Johannes Reuter. "Solgenia - A test vessel toward energy-efficient autonomous water taxi applications". In: *Ocean Engineering* (2024). submitted for publication.

[C2] Andreas Rauh, Stefan Wirtensohn, Patrick Hoher, Johannes Reuter, and Luc Jaulin. "Reliability assessment of an unscented Kalman filter by using ellipsoidal enclosure techniques". In: *Mathematics* 10.16 (2022). ISSN: 2227-7390. DOI: 10.3390/math10163011.

[C3] Andreas Rauh, Yohann Gourret, Katell Lagattu, Bernardo Hummes, Luc Jaulin, Johannes Reuter, Stefan Wirtensohn, and Patrick Hoher. "Experimental validation of ellipsoidal techniques for state estimation in marine applications". In: *Algorithms* 15.5 (2022). ISSN: 1999-4893. DOI: 10.3390/a15050162.

[C4] Tim Baur, Patrick Hoher, Johannes Reuter, and Uwe D. Hanebeck. "Tracking extended objects with basic parametric shapes using deformable superellipses". In: *2024 27th International Conference on Information Fusion (FUSION)*. 2024, pp. 1–8. DOI: 10.23919/FUSION59988.2024.10706433.

[C5] Tim Baur, Patrick Hoher, Johannes Reuter, and Uwe D. Hanebeck. "Extended object tracking using superellipses". In: *Journal of Advances in Information Fusion*. submitted for publication. 2025.

[C6] Pascal Ketterer, Patrick Hoher, and Johannes Reuter. "Runtime optimization in interacting multiple model filtering with down-sampling and out-of-sequence measurements". In: *2024 27th International Conference on Information Fusion (FUSION)*. 2024, pp. 1–8. DOI: 10.23919/FUSION59988.2024.10706441.

Other references

[1] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. "nuScenes: A multimodal dataset for autonomous driving". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11618–11628. DOI: 10.1109/CVPR42600.2020.01164.

[2] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurélien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. "Scalability in perception for autonomous driving: Waymo open dataset". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 2443–2451. DOI: 10.1109/CVPR42600.2020.00252.

[3] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. "A survey of autonomous driving: Common practices and emerging technologies". In: *IEEE Access* 8 (2020), pp. 58443–58469. DOI: 10.1109/ACCESS.2020.2983149.

[4] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. "Vision meets robotics: The KITTI dataset". In: *The International Journal of Robotics Research* 32.11 (2013), pp. 1231–1237. DOI: 10.1177/0278364913491297.

[5] Kemiao Huang and Qi Hao. "Joint multi-object detection and tracking with camera-lidar fusion for autonomous driving". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2021, pp. 6983–6989. DOI: 10.1109/IROS51168.2021.9636311.

[6] Abdalla Gad, Tasnim Basmaji, Maha Yaghi, Huda Alheeh, Mohammad Alkhedher, and Mohammed Ghazal. "Multiple object tracking in robotic applications: Trends and challenges". In: *Applied Sciences* 12.19 (2022). ISSN: 2076-3417. DOI: 10.3390/app12199408.

[7] Ricardo Pereira, Guilherme Carvalho, Luís Garrote, and Urbano J. Nunes. "Sort and deep-SORT based multi-object tracking for mobile robotics: Evaluation with new data association metrics". In: *Applied Sciences* 12.3 (2022). ISSN: 2076-3417. DOI: 10.3390/app12031319.

[8] Liwei Shi, Zhan Chen, Shuxiang Guo, Ping Guo, Yanlin He, Shaowu Pan, Huiming Xing, Shuxiang Su, and Kun Tang. "An underwater pipeline tracking system for amphibious spherical robots". In: *2017 IEEE International Conference on Mechatronics and Automation (ICMA)*. 2017, pp. 1390–1395. DOI: 10.1109/ICMA.2017.8016020.

[9] Wei Wang, David Fernández–Gutiérrez, Rens Doornbusch, Joshua Jordan, Tixiao Shan, Pietro Leoni, Niklas Hagemann, Jonathan Klein Schiphorst, Fabio Duarte, Carlo Ratti, and Daniela Rus. "Roboat III: An autonomous surface vessel for urban transportation". In: *Journal of Field Robotics* 40.8 (2023), pp. 1996–2009. ISSN: 1556-4959. DOI: 10.1002/rob.22237.

[10] Miguel Mujica Mota and Maurice van der Meche. "Towards a greener Europe: Analysis of the SeaBubble waterline in Rotterdam". In: *Proceedings of the 33rd European Modeling & Simulation Symposium (EMSS 2021)*. 2021, pp. 272–280. DOI: 10.46354/i3m.2021.emss.038.

[11] Michael Schuster and Johannes Reuter. "Target tracking in marine environment using automotive radar and laser range sensor". In: *2015 20th International Conference on Methods and Models in Automation and Robotics (MMAR)*. 2015, pp. 965–970. DOI: 10.1109/MMAR.2015.7284009.

[12] Ghassan Al-Falouji, Lukas Haschke, Dirk Nowotka, and Sven Tomforde. "Self-explanation as a basis for self-integration - The autonomous passenger ferry scenario". In: *2023 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*. 2023, pp. 65–70.

[13] Edmund F. Brekke, Egil Eide, Bjørn-Olav H. Eriksen, Erik F. Wilthil, Morten Breivik, Even Skjellaug, Øystein K. Helgesen, Anastasios M. Lekkas, Andreas B. Martinsen, Emil H. Thyri, Tobias Torben, Erik Veitch, Ole A. Alsos, and Tor Arne Johansen. "milliAmpere: An autonomous ferry prototype". In: *Journal of Physics: Conference Series* 2311.1 (2022), p. 012029. ISSN: 1742-6588. DOI: 10.1088/1742-6596/2311/1/012029.

[14] Ole Andreas Alsos, Mina Saghafian, Erik Veitch, Felix-Marcel Petermann, Taufik Akbar Sitompul, Jooyoung Park, Eleftherios Papachristos, Egil Eide, Morten Breivik, and Øyvind Smogeli. "Lessons learned from the trial operation of an autonomous urban passenger ferry". In: *Transportation Research Interdisciplinary Perspectives* 26 (2024), p. 101142. ISSN: 2590-1982. DOI: 10.1016/j.trip.2024.101142.

[15] Sedat Dogru and Lino Marques. "Drone dtection using sparse lidar measurements". In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 3062–3069. DOI: 10.1109/LRA.2022.3145498.

[16] Byron Hernandez and Henry Medeiros. "Multi-object tracking in agricultural applications using a vision transformer for spatial association". In: *Computers and Electronics in Agriculture* 226 (2024), p. 109379. ISSN: 0168-1699. DOI: 10.1016/j.compag.2024.109379.

[17] Rudolf Kalman. "A new approach to linear filtering and prediction problems". In: *Journal of Basic Engineering* (1960), pp. 35–45.

[18] Stefan Wirtensohn, Hanna Wenzl, Thomas Tietz, and Johannes Reuter. "Parameter identification and validation analysis for a small USV". In: *2015 20th International Conference on Methods and Models in Automation and Robotics (MMAR)*. 2015, pp. 701–706. DOI: 10.1109/MMAR.2015.7283960.

[19] Stefan Wirtensohn, Johannes Reuter, Michael Blaich, Michael Schuster, and Oliver Hamburger. "Modelling and identification of a twin hull-based autonomous surface craft". In: *2013 18th International Conference on Methods and Models in Automation and Robotics (MMAR)*. 2013, pp. 121–126. DOI: 10.1109/MMAR.2013.6669892.

[20] Matej Perše, Janez Perš, Matej Kristan, Stanislav Kovacic, and Goran Vučković. "Physics-based modelling of human motion using Kalman filter and collision avoidance algorithm". In: *ISPA 2005. Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, 2005.* 2005, pp. 328–333. DOI: 10.1109/ISPA.2005.195432.

[21] Marcus Brubaker, David Fleet, and Aaron Hertzmann. "Physics-based human pose tracking". In: (Jan. 2006).

[22] Marcus A. Brubaker, David J. Fleet, and Aaron Hertzmann. "Physics-based person tracking using the anthropomorphic walker". In: *International Journal of Computer Vision* 87.1-2 (2010), pp. 140–155. DOI: 10.1007/s11263-009-0274-5.

[23] Subhash Challa. *Fundamentals of object tracking*. Cambridge books online. Cambridge University Press, 2011. ISBN: 9780521876285.

[24] Fredrik Gustafsson. *Statistical sensor fusion*. Professional Publishing House, 2012. ISBN: 9789144077321.

[25] Yaakov Bar-Shalom and Thomas E. Fortmann. *Tracking and Data Association*. Mathematics in science and engineering. Academic Press, 1988. ISBN: 9780120797608.

[26] Samuel S. Blackman and Robert F. Popoli. Artech House radar library. Artech House, 1999. ISBN: 9781580530064.

[27] Robert A. Singer. "Estimating optimal tracking filter performance for manned maneuvering targets". In: *IEEE Transactions on Aerospace and Electronic Systems* AES-6.4 (1970), pp. 473–483. DOI: 10.1109/TAES.1970.310128.

[28] Jeffrey L. Gertz. "Multisensor Surveillance for Improved Aircraft Tracking". In: *Lincoln Laboratory Journal* (1989).

[29] Michael Roth, Gustaf Hendeby, and Fredrik Gustafsson. "EKF/UKF maneuvering target tracking using coordinated turn models with polar/Cartesian velocity". In: *17th International Conference on Information Fusion (FUSION)*. 2014, pp. 1–8.

[30] Johann Wolfgang Koch. "Bayesian approach to extended object and cluster tracking using random matrices". In: *IEEE Transactions on Aerospace and Electronic Systems* 44.3 (2008), pp. 1042–1059. ISSN: 0018-9251. DOI: 10.1109/TAES.2008.4655362.

[31] Michael Feldmann and Dietrich Fränken. "Tracking of extended objects and group targets using random matrices — a new approach". In: *2008 11th International Conference on Information Fusion*. 2008, pp. 1–8.

[32] Michael Feldmann and Dietrich Fränken. "Advances on tracking of extended objects and group targets using random matrices". In: *2009 12th International Conference on Information Fusion*. 2009, pp. 1029–1036.

[33] Nathan James Bartlett, Christopher Renton, and Adrian G. Wills. "A closed-form prediction update for extended target tracking using random matrices". In: *IEEE Transactions on Signal Processing* 68 (2020), pp. 2404–2418. DOI: 10.1109/TSP.2020.2984390.

[34] Michael Feldmann, Dietrich Fränken, and Wolfgang Koch. "Tracking of extended objects and group targets using random matrices". In: *IEEE Transactions on Signal Processing* 59.4 (2011), pp. 1409–1420.

[35] Jian Lan and X. Rong Li. "Tracking of extended object or target group using random matrix: new model and approach". In: *IEEE Transactions on Aerospace and Electronic Systems* 52.6 (2016), pp. 2973–2989. DOI: 10.1109/TAES.2016.130346.

[36] Le Zhang and Jian Lan. "Extended object tracking using random matrix with skewness". In: *IEEE Transactions on Signal Processing* 68 (2020), pp. 5107–5121. DOI: 10.1109/TSP.2020.3019182.

[37] Le Zhang and Jian Lan. "Tracking of extended object using random matrix with non-uniformly distributed measurements". In: *IEEE Transactions on Signal Processing* 69 (2021), pp. 3812–3825. DOI: 10.1109/TSP.2021.3090946.

[38] Karl Granström and Umut Orguner. "A PHD filter for tracking multiple extended targets using random matrices". In: *IEEE Transactions on Signal Processing* 60.11 (2012), pp. 5657–5671.

[39] Karl Granström, Antonio Natale, Paolo Braca, Giovanni Ludeno, and Francesco Serafino. "PHD extended target tracking using an incoherent X-band radar: Preliminary real-world experimental results". In: *17th International Conference on Information Fusion (FUSION)*. 2014, pp. 1–8.

[40] Karl Granström, Antonio Natale, Paolo Braca, Giovanni Ludeno, and Francesco Serafino. "Gamma Gaussian inverse Wishart probability hypothesis density for extended target tracking using X-band marine Radar data". In: *IEEE Transactions on Geoscience and Remote Sensing* 53.12 (2015), pp. 6617–6631.

[41] Michael Schuster, Johannes Reuter, and Gerd Wanielik. "Probabilistic data association for tracking extended targets under clutter using random matrices". In: *2015 18th International Conference on Information Fusion (Fusion)*. 2015, pp. 961–968.

[42] Ronald Mahler. *Statistical multisource-multitarget information fusion*. Norwood, MA, USA: Artech House, 2007. ISBN: 1596930926, 9781596930926.

[43] Ronald Mahler. *Advances in statistical multisource-multitarget information fusion*. Norwood, MA, USA: Artech House, Inc., 2014.

[44] Ronald Mahler. ""Statistics 102" for Multisource-Multitarget Detection and Tracking". In: *IEEE Journal of Selected Topics in Signal Processing* 7.3 (2013), pp. 376–389. ISSN: 1932-4553. DOI: 10.1109/JSTSP.2013.2253084.

[45] Branko Ristić, Ba-Tuong Vo, Ba-Ngu Vo, and Alfonso Farina. "A tutorial on Bernoulli filters: Theory, implementation and applications". In: *IEEE Transactions on Signal Processing* 61.13 (2013), pp. 3406–3430. ISSN: 1053-587X. DOI: 10.1109/TSP.2013.2257765.

[46] Stephan Reuter. "Multi-object tracking using random finite sets". PhD thesis. Universität Ulm, 2014. DOI: 10.18725/oparu-3204.

[47] Stephan Reuter, Ba-Tuong Vo, Ba-Ngu Vo, and Klaus Dietmayer. "The labeled multi-Bernoulli filter". In: *IEEE Transactions on Signal Processing* 62.12 (2014), pp. 3246–3260. ISSN: 1053-587X. DOI: 10.1109/TSP.2014.2323064.

[48] Ba-Tuong Vo and Ba-Ngu Vo. "Labeled Random Finite Sets and Multi-Object Conjugate Priors". In: *IEEE Transactions on Signal Processing* 61.13 (2013), pp. 3460–3475. DOI: 10.1109/TSP.2013.2259822.

[49] Ronald Mahler. "PHD filters for nonstandard targets, I: Extended targets". In: *2009 12th International Conference on Information Fusion*. 2009, pp. 915–921.

[50] Ba-Ngu Vo, Ba-Tuong Vo, and Andrea Cantoni. "The cardinality balanced multi-target multi-Bernoulli filter and its implementations". In: *IEEE Transactions on Signal Processing* 57.2 (2009), pp. 409–423. ISSN: 1053-587X. DOI: 10.1109/TSP.2008.2007924.

[51] David Eppstein. "Finding the k Shortest Paths". In: *SIAM Journal on Computing* 28.2 (1998), pp. 652–673.

[52] Jin Yen. "Finding the K Shortest Loopless Paths in a Network". In: *Management Science* 17.11 (1971), pp. 712–716. ISSN: 00251909, 15265501. URL: http://www.jstor.org/stable/2629312.

[53] Katta G. Murty. "An algorithm for ranking all the assignments in order of increasing cost". In: *Operations Research* 16.3 (1968), pp. 682–687. ISSN: 0030364X, 15265463.

[54] Ronald Mahler. "Multitarget Bayes filtering via first-order multitarget moments". In: *IEEE Transactions on Aerospace and Electronic Systems* 39.4 (2003), pp. 1152–1178. DOI: 10.1109/TAES.2003.1261119.

[55] Ángel F. García-Fernández and Lennart Svensson. "Trajectory PHD and CPHD Filters". In: *IEEE Transactions on Signal Processing* 67.22 (2019), pp. 5702–5714. DOI: 10.1109/TSP.2019.2943234.

[56] Ba-Ngu Vo and Wing-Kin Ma. "The Gaussian mixture probability hypothesis density filter". In: *IEEE Transactions on Signal Processing* 54.11 (Nov. 2006), pp. 4091–4104. ISSN: 1053-587X. DOI: 10.1109/TSP.2006.881190.

[57] Wolfgang Koch and Felix Govaers. "On Accumulated State Densities with Applications to Out-of-Sequence Measurement Processing". In: *IEEE Transactions on Aerospace and Electronic Systems* 47.4 (2011), pp. 2766–2778. DOI: 10.1109/TAES.2011.6034663.

[58] Wolfgang Koch. "On Accumulated State Densities with applications to out-of-sequence measurement processing". In: *2009 12th International Conference on Information Fusion*. 2009, pp. 2201–2208.

[59] Wolfgang Koch, Felix Govaers, and Alexander Charlish. "An exact solution to track-to-track fusion using accumulated state densities". In: *2013 Workshop on Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. 2013, pp. 1–5. DOI: 10.1109/SDF.2013.6698253.

[60] Ángel. F. García-Fernández, Lennart Svensson, and Mark R. Morelande. "Multiple Target Tracking Based on Sets of Trajectories". In: *IEEE Transactions on Aerospace and Electronic Systems* 56.3 (2020), pp. 1685–1707. DOI: 10.1109/TAES.2019.2921210.

[61] Ángel. F. García-Fernández and Lennart Svensson. "Trajectory probability hypothesis density filter". In: *2018 21st International Conference on Information Fusion (FUSION)*. 2018, pp. 1430–1437. DOI: 10.23919/ICIF.2018.8455270.

[62] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, 2006, pp. I–XVIII, 1–248. ISBN: 026218253X.

[63] Simon Steuernagel, Kolja Thormann, and Marcus Baum. "Modeling Contour Measurements of Elliptical Extended Objects via Gaussian Spatial Distributions". In: *2024 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. 2024, pp. 1–8. DOI: 10.1109/SDF63218.2024.10773723.

[64] Karl Granström, Christian Lundquist, and Umut Orguner. "Tracking rectangular and elliptical extended targets using laser measurements". In: *14th International Conference on Information Fusion*. 2011, pp. 1–8.

[65] Barkın Tuncer and Emre Özkan. "Random matrix based extended target tracking with orientation: A new model and inference". In: *IEEE Transactions on Signal Processing* 69 (2021), pp. 1910–1923. DOI: 10.1109/TSP.2021.3065136.

[66] Karl Granström and Jakob Bramstång. "Bayesian smoothing for the extended object random matrix model". In: *IEEE Transactions on Signal Processing* 67.14 (2019), pp. 3732–3742. DOI: 10.1109/TSP.2019.2920471.

[67] Jian Lan and X. Rong Li. "Extended-object or group-target tracking using random matrix with nonlinear measurements". In: *IEEE Transactions on Signal Processing* 67.19 (2019), pp. 5130–5142. DOI: 10.1109/TSP.2019.2935866.

[68] Hosam Alqaderi, Felix Govaers, and Raymond Schulz. "Spacial elliptical model for extended target tracking using laser measurements". In: *2019 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. 2019, pp. 1–6. DOI: 10.1109/SDF.2019.8916634.

[69] Lars Hammarstrand, Lennart Svensson, Fredrik Sandblom, and Joakim Sorstedt. "Extended Object tracking using a Radar resolution model". In: *IEEE Transactions on Aerospace and Electronic Systems* 48.3 (2012), pp. 2371–2386. DOI: 10.1109/TAES.2012.6237597.

[70] Kristian A. Ruud, Edmund F. Brekke, and Jo Eidsvik. "Lidar extended object tracking of a maritime vessel using an ellipsoidal contour model". In: *2018 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. 2018, pp. 1–6. DOI: 10.1109/SDF.2018.8547047.

[71] Marcus Baum, Benjamin Noack, and Uwe D. Hanebeck. "Extended object and group tracking with elliptic random hypersurface models". In: *2010 13th International Conference on Information Fusion*. 2010, pp. 1–8. DOI: 10.1109/ICIF.2010.5711854.

[72] Marcus Baum and Uwe D. Hanebeck. "Extended object tracking with random hypersurface models". In: *IEEE Transactions on Aerospace and Electronic Systems* 50.1 (2014), pp. 149–159. DOI: 10.1109/TAES.2013.120107.

[73] Shishan Yang and Marcus Baum. "Tracking the orientation and axes lengths of an elliptical extended object". In: *IEEE Transactions on Signal Processing* 67.18 (2019), pp. 4720–4729. DOI: 10.1109/TSP.2019.2929462.

[74] Karl Granström, Stephan Reuter, Daniel Meissner, and Alexander Scheel. "A multiple model PHD approach to tracking of cars under an assumed rectangular shape". In: *17th International Conference on Information Fusion (FUSION)*. 2014, pp. 1–8.

[75] Marcus Baum and Uwe D. Hanebeck. "Tracking an extended object modeled as an axis-aligned rectangle". In: *4th German Work- shop on Sensor Data Fusion: Trends, Solutions, Applications (SDF 2009)*. 2009.

[76] Tobias Hirscher, Alexander Scheel, Stephan Reuter, and Klaus Dietmayer. "Multiple extended object tracking using Gaussian processes". In: *2016 19th International Conference on Information Fusion (FUSION)*. 2016, pp. 868–875.

[77] Niklas Wahlström and Emre Özkan. "Extended target tracking using Gaussian processes". In: *IEEE Transactions on Signal Processing* 63.16 (2015), pp. 4165–4178. DOI: 10.1109/TSP.2015.2424194.

[78] Murat Kumru and Emre Özkan. "Three-dimensional extended object tracking and shape learning using Gaussian processes". In: *IEEE Transactions on Aerospace and Electronic Systems* 57.5 (2021), pp. 2795–2814. DOI: 10.1109/TAES.2021.3067668.

[79] Tim Baur, Johannes Reuter, Antonio Zea, and Uwe D. Hanebeck. "Extent estimation of sailing boats applying elliptic cones to 3D Lidar data". In: *2022 25th International Conference on Information Fusion (FUSION)*. 2022, pp. 1–8. DOI: 10.23919/FUSION49751.2022.9841265.

[80] Florian Faion, Marcus Baum, and Uwe D. Hanebeck. "Tracking 3D shapes in noisy point clouds with random hypersurface models". In: *2012 15th International Conference on Information Fusion*. 2012, pp. 2230–2235.

[81] Karl Granström, Marcus Baum, and Stephan Reuter. "Extended object tracking: Introduction, overview, and applications". In: *Journal of Advances in Information Fusion* 12 (Dec. 2017).

[82] Daniel Frisch and Uwe D. Hanebeck. "Rejection sampling from arbitrary multivariate distributions using generalized Fibonacci lattices". In: *2022 25th International Conference on Information Fusion (FUSION)*. 2022, pp. 1–7. DOI: 10.23919/FUSION49751.2022.9841322.

[83] Thor I. Fossen. *Marine control systems: Guidance, navigation and control of ships, rigs and underwater vehicles*. Marine Cybernetics, 2002. ISBN: 9788292356005.

[84] Florian Faion, Antonio Zea, Jannik Steinbring, Marcus Baum, and Uwe D. Hanebeck. "Recursive Bayesian pose and shape estimation of 3D objects using transformed plane curves". In: *2015 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. 2015, pp. 1–6. DOI: 10.1109/SDF.2015.7347698.

[85] Michael Schuster. *Multiple object tracking for extended targets using JIPDA filters*. Aachen: Shaker Verlag, 2017.

[86] John A. Nelder and Roger Mead. "A simplex method for function minimization". In: *Computer Journal* 7 (1965), pp. 308–313.

[87] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. "A density-based algorithm for discovering clusters in large spatial databases with noise". In: KDD'96. Portland, Oregon: AAAI Press, 1996, pp. 226–231.

[88] Pavlina Konstantinova, Alexander Udvarev, and Tzvetan Semerdjiev. "A study of a target tracking algorithm using global nearest neighbor approach". In: CompSysTech '03. Rousse, Bulgaria: Association for Computing Machinery, 2003, pp. 290–295. ISBN: 9549641333. DOI: 10.1145/973620.973668. URL: https://doi.org/10.1145/973620.973668.

[89] Ba-Ngu Vo, Ba-Tuong Vo, and Hung Gia Hoang. "An efficient implementation of the generalized labeled multi-Bernoulli filter". In: *IEEE Transactions on Signal Processing* 65.8 (2017), pp. 1975–1987. DOI: 10.1109/TSP.2016.2641392.

[90] Barkin Tuncer, Murat Kumru, Emre Özkan, and A. Aydin Alatan. "Extended object tracking and shape classification". In: *2018 21st International Conference on Information Fusion (FUSION)*. 2018, pp. 1–5. DOI: 10.23919/ICIF.2018.8455464.

[91] Harry G. Barrow, Jay M. Tenenbaum, Robert C. Bolles, and Helen C. Wolf. "Parametric correspondence and Chamfer matching: two new techniques for image matching". In: IJCAI'77. Cambridge, USA: Morgan Kaufmann Publishers Inc., 1977, pp. 659–663.

[92] Tong Wu, Liang Pan, Junzhe Zhang, Tai Wang, Ziwei Liu, and Dahua Lin. "Balanced Chamfer distance as a comprehensive metric for point cloud completion". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan. Vol. 34. Curran Associates, Inc., 2021, pp. 29088–29100.

[93] Fangzhou Lin, Yun Yue, Ziming Zhang, Songlin Hou, Kazunori Yamada, Vijaya Kolachalama, and Venkatesh Saligrama. "InfoCD: A contrastive Chamfer distance loss for point cloud completion". In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine. Vol. 36. Curran Associates, Inc., 2023, pp. 76960–76973.

[94] Marcus Baum and Uwe D. Hanebeck. "Shape tracking of extended objects and group targets with star-convex RHMs". In: *14th International Conference on Information Fusion*. 2011, pp. 1–8.

[95] Michael Levandowsky and David Winter. "Distance between sets". In: *Nature* 234.5323 (1971), pp. 34–35. ISSN: 1476-4687. DOI: 10.1038/234034a0.

[96] Tim Baur, Johannes Reuter, Antonio Zea, and Uwe D. Hanebeck. "Shape tracking using Fourier-Chebyshev double series for 3D distance measurements". In: *2023 26th International Conference on Information Fusion (FUSION)*. 2023, pp. 1–8. DOI: 10.23919/FUSION52260.2023.10224120.

[97] Angel. F. García-Fernández, Jason L. Williams, Karl Granström, and Lennart Svensson. "Poisson multi-Bernoulli mixture filter: Direct derivation and implementation". In: *IEEE Transactions on Aerospace and Electronic Systems* 54.4 (2018), pp. 1883–1901. DOI: 10.1109/TAES.2018.2805153.

[98] Ronald Mahler. "PHD filters of higher order in target number". In: *IEEE Transactions on Aerospace and Electronic Systems* 43.4 (2007), pp. 1523–1543. DOI: 10.1109/TAES.2007.4441756.

[99] Karl Granström, Christian Lundquist, and Omut Orguner. "Extended target tracking using a Gaussian-mixture PHD filter". In: *IEEE Transactions on Aerospace and Electronic Systems* 48.4 (2012), pp. 3268–3286. DOI: 10.1109/TAES.2012.6324703.

[100] Ba-Tuong Vo, Ba-Ngu Vo, and Antonio Cantoni. "Analytic implementations of the cardinalized probability hypothesis density filter". In: *IEEE Transactions on Signal Processing* 55.7 (2007), pp. 3553–3567. DOI: 10.1109/TSP.2007.894241.

[101] Shoufeng Lin, Ba Tuong Vo, and Sven E. Nordholm. "Measurement driven birth model for the generalized labeled multi-Bernoulli filter". In: *2016 International Conference on Control, Automation and Information Sciences (ICCAIS)*. 2016, pp. 94–99. DOI: 10.1109/ICCAIS.2016.7822442.

[102] Branko Ristić, Daniel E. Clark, Ba-Ngu Vo, and Ba-Tuong Vo. "Adaptive target birth intensity for PHD and CPHD filters". In: *IEEE Transactions on Aerospace and Electronic Systems* 48.2 (Apr. 2012), pp. 1656–1668. ISSN: 0018-9251. DOI: 10.1109/TAES.2012.6178085.

[103] Àngel F. Garcìa-Fernàndez, Yuxuan Xia, and Lennart Svensson. "A comparison between PMBM Bayesian track initiation and labelled RFS adaptive birth". In: *2022 25th International Conference on Information Fusion (FUSION)*. 2022, pp. 1–8. DOI: 10.23919/FUSION49751.2022.9841338.

[104] Qian Zhu, Tao Li, Jiameng Pan, and Qinglong Bao. "The Modified Probability Hypothesis Density Filter With Adaptive Birth Intensity Estimation for Multi-Target Tracking in Low Detection Probability". In: *IEEE Access* 8 (2020), pp. 43690–43710. DOI: 10.1109/ACCESS.2020.2977431.

[105] Yan Cang, Di Chen, and Weijin Sun. "Adaptive target birth intensity for Gaussian Mixture Probability Hypothesis Density (GM-PHD) filter". In: *2014 IEEE International Conference on Control Science and Systems Engineering*. 2014, pp. 36–39. DOI: 10.1109/CCSSE.2014.7224504.

[106] Feng Yang, Cangan Sun, Yumei Hu, Litao Zheng, and Jianchun Lu. "The comparison of PHD algorithms with adaptive target birth". In: *2017 IEEE International Conference on Unmanned Systems (ICUS)*. 2017, pp. 438–443. DOI: 10.1109/ICUS.2017.8278385.

[107] Christopher Berry, Donald J. Bucci, and Samuel Watt Schmidt. "Passive multi-target tracking using the adaptive birth intensity PHD filter". In: *2018 21st International Conference on Information Fusion (FUSION)*. 2018, pp. 353–360. DOI: 10.23919/ICIF.2018.8455308.

[108] Shayle Robert Searle. *The recurrence formulae for means and variances*. Ithaca, New York, USA: Biometrics Unit Cornell University, 1981. URL: https://core.ac.uk/download/pdf/79038021.pdf.

[109] Irénée-Jules Bienaymé. "Considérations à l'appui de la découverte de Laplace sur la loi de probabilité dans la méthode des moindres carrés". In: *Comptes rendus de l'Académie des sciences Paris*. Vol. 37. 1853, pp. 309–317.

[110] Surajit Ray and Bruce G. Lindsay. "The topography of multivariate normal mixtures". In: *The Annals of Statistics* 33.5 (2005), pp. 2042–2065. ISSN: 00905364. URL: http://www.jstor.org/stable/3448634 (visited on 09/21/2022).

[111] Ivan Marković and Ivan Petrović. "Bearing-only tracking with a mixture of von Mises distributions". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2012, pp. 707–712. DOI: 10.1109/IROS.2012.6385600.

[112] Solomon Kullback and Richard A. Leibler. "On information and sufficiency". In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86. DOI: 10.1214/aoms/1177729694. URL: https://doi.org/10.1214/aoms/1177729694.

[113] Thomas M. Cover and Joy A. ThomasB. "Entropy, relative entropy, and mutual information". In: *Elements of Information Theory*. John Wiley & Sons, Ltd, 2005. Chap. 2, pp. 13–55. ISBN: 9780471748823. DOI: https://doi.org/10.1002/047174882X.ch2. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/047174882X.ch2. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/047174882X.ch2.

[114] William H. Greene. *Econometric analysis*. 5th ed. Upper Saddle River, New Jersey, USA: Prentice Hall, 2003. ISBN: 978-0-13-066189-0.

[115] Dominic Schuhmacher, Ba-Tuong Vo, and Ba-Ngu Vo. "A consistent metric for performance evaluation of multi-object filters". In: *IEEE Transactions on Signal Processing* 56.8 (Aug. 2008), pp. 3447–3457. ISSN: 1053-587X. DOI: 10.1109/TSP.2008.920469.