FH MÜNSTER
University of Applied Sciences

FACHHOCHSCHULE MÜNSTER

Fachbereich Elektrotechnik und Informatik

MASTERARBEIT

zur Erlangung des akademischen Grades Master of Science (M.Sc.) im Studiengang Informatik

HYBRIDISATION OF SEQUENTIAL MONTE CARLO SIMULATION WITH NON-LINEAR BOUNDED-ERROR STATE ESTIMATION BASED ON INTERVAL ANALYSIS APPLIED TO GLOBAL LOCALISATION OF MOBILE ROBOTS

Vorgelegt am	6. November 2018
von	Robin Weiß
Matrikelnummer:	657193
Erstprüfer:	Prof. Dr.-Ing. Peter Glösekötter
Zweitprüfer:	Prof. Dr. Mariana Luderitz Kolberg

ABSTRACT

When creating mobile robots that operate autonomously, possibly without any human supervision, accurate self-localisation is one of the fundamental abilities such robots should possess. In conventional Monte Carlo localisation, a probability distribution over a space of possible hypotheses accommodates the inherent uncertainty in the position estimate, whereas bounded-error localisation provides a region that is guaranteed to contain the robot. However, this guarantee is accompanied by a constant probability over the confined region and therefore, depending on its size, the information yield may not be sufficient for certain practical applications. In this thesis, four new hybrid localisation algorithms are proposed, combining probabilistic filtering with non-linear bounded-error state estimation based on interval analysis. A forward-backward contractor and the Set Inverter via Interval Analysis are hybridised with a bootstrap filter and an unscented particle filter, respectively. The four new algorithms are applied to global localisation of an underwater robot, using simulated distance measurements to distinguishable landmarks. As opposed to previous hybrid methods found in the literature, the bounded-error state estimate is not maintained throughout the whole estimation process. Instead, it is only computed once in the beginning, when solving the wake-up robot problem, and after kidnapping of the robot, which drastically reduces the computational cost when compared to existing algorithms. Evaluating the performance of the localisation algorithms with different numbers of landmarks, it is shown that the newly proposed algorithms can solve the wake-up robot problem as well as the kidnapped robot problem more accurately than the two conventional probabilistic filters.

To me, mathematics, computer science and the arts are insanely related. They're all creative expressions.

— Sebastian Thrun

ACKNOWLEDGMENTS

I would like to thank Professor Peter Glösekötter, who has been supportive since the days I was an undergraduate student, and whose drive for innovation I admire. It was with his supervision that this work came into existence. In addition to my stay in Brazil, he enabled me to both work and pursue my studies in England, Spain, and China. Thank you for the excellent cooperation and for all of the opportunities I was given to, besides boosting my technical skills, broaden my personal horizons during this incredibly instructive period.

Also, I must express my sincere gratitude to Mariana Kolberg and Edson Prestes for co-supervising this work. Thank you for the lively discussions and your patient mentoring. Collaborating with you and the whole team of the Φ-Robotics Research Group has enriched my experience in Porto Alegre tremendously. I have been extremely lucky to have been surrounded by people like you, who cared so much about my work and who responded to my questions and queries so promptly.

My wholehearted gratefulness and appreciation goes to Mathias Mantelli, who helped me grow personally beyond my work at the keyboard and who cheered me up during the most difficult times when writing this thesis. Without him, I would have been lost many times. Our stimulating discussions and his attention to detail profoundly contributed in the completion of this thesis. Thank you for all the fun we had in the past six months.

I heartily dedicate this thesis to my mother Karin and to my brother Manuel, whose love and guidance are with me in whatever I pursue. Thank you for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. What I have accomplished would not have been possible without you.

Last but not least, I would like to express my very profound gratitude to my friends Nina, Denise, Sebastian and Tim. I am so blessed to have you by my side. During all the months-long periods that I spent abroad, you were the ones who waited patiently for me to return whilst encouraging me from back home.

Thank you all for your unwavering support.

Rob

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACRONYMS

AUV	Autonomous underwater vehicle
CDF	Cumulative distribution function
CSP	Constraint satisfaction problem
EKF	Extended Kalman filter
KF	Kalman filter
NCSP	Numerical constraint satisfaction problem
PDF	Probability density function
PF	Particle filter
SIR	Sequential importance resampling
SIS	Sequential importance sampling
SMC	Sequential Monte Carlo
UKF	Unscented Kalman filter
UPF	Unscented particle filter

$\lvert[x]\rvert$	Absolute value of an interval $[x]$
α	Scaling parameter that controls the spread of sigma points around the mean
$\mathbf{B}$	Control matrix that relates the control input to the state of a linear dynamical system
$\mathcal{C}\big([\mathbf{x}]\big)$	Contractor applied to the box $[\mathbf{x}]$
$\left.\dfrac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}}\right\rvert_{\mathbf{x}=\mathbf{x}_0}$	Partial derivative of $\mathbf{f}$ with respect to $\mathbf{x}$, evaluated at $\mathbf{x}_0$
$\mathbb{E}[\mathbf{x}]$	Expected value of $\mathbf{x}$
$\mathbb{E}_{p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})}\big[f(\mathbf{x}_k)\big]$	Expectation under $p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})$
$\hat{\mathbb{E}}\big[f(\mathbf{x}_k)\big]$	Estimate of the expectation $\mathbb{E}\big[f(\mathbf{x}_k)\big]$
$[\mathbf{f}]\big([\mathbf{x}]\big)$	Inclusion function of $\mathbf{f}$
$\mathbf{h}$	Non-linear function that relates the measurement to the state of a dynamical system
$\mathbf{H}$	Measurement sensitivity matrix defining the linear relationship between the state of a dynamical system and its observation
$\mathbf{H}^{[1]}$	Jacobian matrix of the function $\mathbf{h}$
$\mathbf{I}_n$	Identity matrix, $\mathbf{I}_n \in \mathbb{R}^{n \times n}$
k	Discrete time, $k \in \mathbb{N}^0$
$\mathbf{K}$	Kalman gain matrix
κ	Scaling parameter in unscented transformation; $\kappa \geqslant 1$ guarantees positive semi-definiteness of the covariance matrix
$\mathrm{mid}\big([x]\big)$	Midpoint of an interval $[x]$

n	Non-dimensional number
$p(\mathbf{x}_0)$	Probability distribution of the initial state
$p(\mathbf{x}_k \mid \mathbf{Z}_{k-1}, \mathbf{U}_{k-1})$	Predictive distribution of the state $\mathbf{x}_k$ at the current time k, given the entire sequence of observations and the entire sequence of control inputs up to and including time $k-1$
$p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})$	Posterior distribution of the current state $\mathbf{x}_k$, given the entire sequence of observations up to and including the current time k and the entire sequence of control inputs up to and including time $k-1$
$p_C(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})$	Constrained posterior distribution of the current state $\mathbf{x}_k$, given the entire sequence of observations up to and including the current time k and the entire sequence of control inputs up to and including time $k-1$
$p_{\mathcal{N}}(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})$	Gaussian approximation of the posterior distribution of the current state $\mathbf{x}_k$, given the entire sequence of observations up to and including the current time k and the entire sequence of control inputs up to and including time $k-1$
$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_{k-1})$	State-transition distribution of the current state $\mathbf{x}_k$, given the immediate past state $\mathbf{x}_{k-1}$ and control input $\mathbf{u}_{k-1}$
$p(z_k \mid \mathbf{x}_k)$	Likelihood function of the current observation z_k, given the current state $\mathbf{x}_k$
$\mathbf{P}_k$	Error covariance matrix representing the uncertainty in the state estimation
$\mathbf{P}_{k\mid k-1}$	A priori error covariance matrix representing the uncertainty of the a priori estimate $\hat{\mathbf{x}}_{k\mid k-1}$
$\mathbf{P}_{\tilde{z}_k \tilde{z}_k}$	Innovation covariance matrix
$\mathbf{P}_{\tilde{x}_k \tilde{z}_k}$	Cross covariance matrix
$\boldsymbol{\phi}$	Non-linear state transition function of a dynamical system
$\boldsymbol{\Phi}$	State transition matrix of a linear dynamical system
$\boldsymbol{\Phi}^{[1]}$	Jacobian matrix of the function $\boldsymbol{\phi}$
$\mathbf{Q}$	Covariance matrix of the process noise

$\mathbf{R}$	Covariance matrix of the measurement noise
$\mathbb{R}^n$	n-dimensional real vector space
$\mathbb{S}$	Solution set of a numerical constraint satisfaction problem or of a set inversion problem
$\left(\sqrt{(n+\lambda)\mathbf{P}_{\mathbf{x}}}\right)_i$	The i-th column of the matrix square root of $(n+\lambda)\mathbf{P}_{\mathbf{x}}$
$\mathbf{u}$	Control input vector
$\mathbf{U}_k$	Sequence of control inputs, denoting $\{\mathbf{u}_i\}_{i=0}^{k}$
$\boldsymbol{v}$	Measurement noise vector
$\boldsymbol{w}$	Process noise vector
$w\big([x]\big)$	Width of an interval $[x]$
$\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{P})$	The random variable $\boldsymbol{w}$ is distributed normally with mean $\boldsymbol{\mu}$ and covariance $\mathbf{P}$
$W_i^{(c)}$	Weight of the i-th sigma point for the computation of the covariance
$W_i^{(m)}$	Weight of the i-th sigma point for the computation of the mean
$[x]$	Real interval, denoting $\big\{x \in \mathbb{R} \mid \underline{x} \leqslant x \leqslant \overline{x}\big\}$
$[\mathbf{x}]$	Real interval vector, denoting the Cartesian product of intervals, $[x]_1 \times [x]_2 \times \cdots \times [x]_n$
$\mathbf{x}$	State vector of a dynamical system
$\mathbf{x}_k$	The kth element of a sequence $\ldots, \mathbf{x}_{k-1}, \mathbf{x}_k, \mathbf{x}_{k+1}, \ldots$ of vectors
$\hat{\mathbf{x}}$	Estimate of the state vector of a dynamical system
$\bar{\mathbf{x}}$	Mean of the random vector $\mathbf{x}$
$\hat{\mathbf{x}}_{k\|k-1}$	A priori estimate of $\hat{\mathbf{x}}_k$, conditioned on all prior measurements except the one at time k
$\mathbf{X}_k$	Sequence of states, denoting $\{\mathbf{x}_i\}_{i=0}^{k}$
$\mathcal{X}_i$	The i-th sigma point
$\boldsymbol{\mathcal{X}}_{k-1}^{a}$	Set of augmented sigma points computed using the previous state estimate
$\boldsymbol{\mathcal{X}}_{k-1}$	Set of sigma points computed using the previous state estimate

$\mathcal{X}^{v}_{k\|k-1}$	Set of sigma points constituted of the components that are associated with the measurement noise
$\mathcal{X}^{w}_{k\|k-1}$	Set of sigma points constituted of the components that are associated with the process noise
$\mathcal{X}^{x}_{k\|k-1}$	Set of sigma points constituted of the components that are associated with the state
$\mathcal{X}$	Set of sigma points, denoting $\{\mathcal{X}_i\}_{i=0}^{2n}$
z^{-1}	Unit-delay
z	Observation or measurement vector of a dynamical system
z_0	Empty measurement
$\hat{z}_{k\|k-1}$	Prediction of the measurement vector z_k before it comes available at time k
Z_k	Sequence of observations, denoting $\{z_i\}_{i=1}^{k}$
$\mathcal{Z}_{k\|k-1}$	Set of sigma points capturing the predicted measurement

1 INTRODUCTION

For a long time, people have dreamt of building intelligent machines to perform tedious, repetitive or dangerous tasks. Today, we call these machines robots, derived from the Slavic word *robota*, meaning servitude or drudgery [1]. For a mobile robot, in order to operate in its environment, it is essential to know its position relative to an external reference frame. The corresponding localisation problem constitutes the most basic perceptual challenge in robotics and is described in more detail below.

1.1 THE GLOBAL LOCALISATION PROBLEM

A scenario in which a robot is given a map of its environment, to estimate its position relative to this map using its sensors, is commonly known as the *global localisation problem*. In contrast, *position tracking* denotes the process of continuously determining the robot's position relative to a known initial position. Since here the uncertainty is confined to the region near the robot's true position, tracking is a *local* problem and easier than global localisation. When a robot has to establish its own global position without prior knowledge, for instance after having been carried to an arbitrary location before being put to operation, this is referred to as the *wake-up robot problem*. The related *kidnapped robot problem*, which forms another subclass of global localisation problems, describes a situation where a well-localised mobile robot is teleported to an arbitrary location without being told. That is, the robot strongly believes itself to be somewhere else at the time of the kidnapping. Solving the above problems under varying circumstances using different sensory input represents a major challenge in robotics and is of paramount importance for a successful practical application of mobile robots.

1.1.1 *Sensor Systems*

In tackling the self-localisation problems described above, the fundamental choices of sensory information may be classified as those obtained by *proprioceptive* sensors, such as wheel encoders or inertial sensors, and *exteroceptive* sensors, such as cameras, laser scanners or supersonic sensors. By means of the latter type of sensors, the robot can sense *landmarks*, which denote any identifiable sensory perception that has a known position with respect to a given global coordinate system. In practice, proprioceptive methods alone fail after a short

time since they are impaired by incorrigible drift errors. For this reason, a fusion of proprioceptive and exteroceptive information has been widely and successfully used in mobile robotics [2].

1.1.2 *Sensor Fusion*

As noise-free sensors for measuring the position do not exist, the position has to be inferred from the evolution of noisy sensor data over time. The combination of information from multiple sensors with the aim to increase the overall precision of the estimation of a certain quantity of interest is termed *sensor fusion*. Raol [3] states the following advantages of sensor fusion:

- Robust functional and operational performance is given, in case of data loss from one sensor, due to redundancy provided by multiple sensors.
- Enhanced confidence in the results inferred from the measurement of one sensor, if they are confirmed by the measurement of another sensor.
- With sensor fusion an arbitrary fine time resolution of measurements is possible, whereas single sensors need a finite time to transmit measurements and so limit the frequency of measurements.
- One sensor might be, to some extent, better in a certain state of the measured process and thus, by fusing multiple sensor signals, a satisfactory accuracy among all states of a process may be attained.

As we shall see in Chapter 2, in the context of self-localisation, digital filters can be used to fuse information from multiple sensors with additional a priori information about the position of a mobile robot.

1.2 MOTIVATION

Robotics has enormous potential to change the world for the better, beckoning humanity to tap into it. Releasing mankind from mundane and repetitive tasks frees time and energy and thus allows us to focus on the more interpersonal and creative aspects of life. Enhanced efficiency enables us to engage with higher order functions and to devote our time to further improving quality of life. Automating dangerous tasks protects human lives and renders operation under inhuman conditions possible in the first place. Robotics' potential to make a positive, lasting impact on the world was the motivation for this work.

1.3 STATE OF THE ART

Mobile robots have successfully been applied to search and rescue [4–8], monitoring and surveillance [9], autonomous locomotion [10, 11], and education [12]. When aiming at creating mobile robots that operate autonomously and safely in these scenarios, possibly without any human supervision, accurate self-localisation is one of the fundamental abilities any mobile robot should possess. In order to perform robust localisation, many novel methods have emerged in the past few years, based both on probability theory and interval analysis. Nevertheless, self-localisation remains a rich and highly active area of research to this day.

1.3.1 *Probabilistic Localisation*

Taking a probabilistic approach to localisation, uncertainty is represented explicitly, using the calculus of probability theory. In *probabilistic localisation* a probability distribution over a space of possible hypotheses accommodates the inherent uncertainty in a position estimate. This uncertainty originates from the above-mentioned measurement noise and other influences discussed in more detail in the following chapter. In the literature, there are numerous probabilistic methods, applied to both global localisation as well as tracking. Besides a manifold of particle filters [13–16], Kalman filters [15, 17] have been used successfully.

1.3.2 *Bounded-Error Localisation*

Naturally, the probabilistic approach does not ensure a correct solution and in fact probabilistic estimators may diverge under certain circumstances. Hence, several methods based on interval computations have been developed to tackle the local and global localisation problem [18–26]. As opposed to the probabilistic approach, which provides a point estimate of the position, the rationale of the so-called *bounded-error localisation* is to determine a region that is guaranteed to contain the robot's position. Here, however, the guarantee is accompanied by a constant probability over the confined region and in practice the information yield may not be sufficient.

1.3.3 *Hybrid Localisation*

In addition to stand-alone probabilistic or bounded-error localisation methods, there have been attempts to combine both in order to mitigate their respective shortcomings and thus improve the localisation accuracy. Neuland et al. [27] used a particle filter in combination with a set-membership method to restrict the spread of particles to

regions of the search space that are associated with a high observation likelihood. In [28] they used a set-inversion method instead and reported an increase in the estimation accuracy in return for higher computational cost in both cases.

Nicola [29] combined a set-membership method with a Kalman filter to obtain a reliable and precise algorithm for simultaneous localisation and mapping of underwater robots. Ashokaraj et al. proposed sensor-based robot localisation using an extended Kalman filter [30] as well as an unscented Kalman filter [31] in combination with interval analysis to bound the estimation error in the presence of landmarks. If the position estimate of the Kalman filter lay outside of the feasible region it was corrected to the geometrically closest point on the boundary. In [32], multiple interval robot positions were processed using a fuzzy logic weighted average algorithm to obtain a single robot interval position. The error of an unscented Kalman filter position estimate was then bounded by the interval robot position as described above. In [33], Ashokaraj et al. used ultrasonic sensors with limited range and corrected the mean and covariance of an unscented Kalman filter by means of the interval estimate. All their methods resulted in a more accurate position estimate.

1.4 CONTRIBUTIONS

As will be shown in the following chapter, in Monte Carlo localisation there is a natural dilemma between accuracy and computational cost, which can be balanced by the number of hypotheses used, also referred to as particles. Both bounded-error state estimation as well as an unscented Kalman filter can be utilised in order to move particles to regions of the state space that are associated with a high observation likelihood. As no particles are wasted in unlikely regions, with a finite number of particles the particle density in likely regions can be increased and consequently the localisation accuracy can be improved. Previous hybrid localisation methods successfully combined bounded-error state estimation with a bootstrap particle filter. However, it is shown experimentally below that when too little information in terms of visible landmarks are available, the hybridisation of both bounded-error state estimation and an unscented particle filter can further improve the estimation accuracy.

A particle filter and an unscented particle filter is combined with two bounded-error estimators, namely a forward-backward contractor and the Set Inverter via Interval Analysis, respectively. The four resulting new hybrid localisation algorithms and the two conventional probabilistic filters are applied in three different simulated landmark-based global localisation scenarios. While the hybrid localisation methods in [27] and [28] maintain both a bootstrap particle filter estimate and a bounded-error state estimate at each time step, in order

to bound the estimation error of the particle filter, it is shown that the localisation accuracy particularly benefits from the bounded-error state estimate in the very first iterations or shortly after kidnapping of the robot, respectively. The bounded-error state estimate is therefore not maintained throughout the whole estimation process. Instead, the satisfaction of given constraints based on geometrical considerations of the environment are tested, in order to bound the estimation error and detect kidnapping. In the latter case, the bounded-error state estimation is triggered again to repeat the global localisation process over the entire map. The rationale behind the newly proposed algorithms, which are explained in detail in Chapter 3, is to drastically reduce computational cost when compared to previous methods while preserving the benefits of the hybrid approach and therefore improve the estimation accuracy when compared to conventional unconstrained probabilistic filtering.

1.5 METHODOLOGY

In the following chapter, stochastic filtering theory is reviewed in detail, beginning with a statement of the problem and its conceptual solution, the Bayesian filter. Under linear-quadratic-Gaussian circumstances, the Kalman filter represents an optimal solution within the Bayesian framework. As the linear scenario is hardly found in practice, suboptimal non-linear filtering techniques such as the extended Kalman filter and the unscented Kalman filter are extensively investigated. Subsequently, we focus our attention on the sequential Monte Carlo method, including importance sampling and resampling. As another representative of the Bayesian framework, the generic particle filter assumes a non-linear, non-Gaussian model. Using the transition prior probability distribution as a proposal distribution yields the bootstrap filter, while the unscented particle filter represents a more elaborate estimator that uses an unscented Kalman filter in order to generate better proposal distributions. At the end of Chapter 2, we introduce basic concepts of interval analysis and two algorithms that can be applied to bounded-error localisation. Chapter 3 elaborates on the four newly proposed localisation algorithms. The system and measurement models of both the probabilistic and bounded-error estimators are described and the incorporation of constraints into the inherently unconstraint bayesian filters is explained. In Chapter 4 the four new hybrid localisation algorithms and the two conventional probabilistic filters are put to the test in three different simulated landmark-based global localisation scenarios. A detailed explanation of the experiments is followed by the presentation of the results. Finally, in Chapter 5 we conclude and present potential future work.

2 THEORETICAL BACKGROUND

Conceived in general terms, a filter is a device for removing unwanted components of a mixture. In accordance with this definition, in the technical sphere, a *filter* refers to a system designed to extract information about a quantity of interest, using noisy observations of a process. That is, a filter delivers an estimate of the variables of principal interest, which is why it is also called an *estimator*.

In the next section, we will present the filtering problem in a formal manner, followed by a comprehensive introduction of a generic framework to cope with it, the Bayesian filter. The Kalman filter presented in Section 2.3.1 is suitable for linear models and additive white Gaussian noise. We deliberately chose to start from this special-case scenario, generalising step by step, as the Kalman filter is an easy to comprehend, straight forward solution to the filtering problem and its mechanism and underlying terminology is essential for the remainder of this thesis. In Section 2.3.2, we will introduce the extended Kalman filter, which uses linearisation around the latest state estimate in order to cope with non-linear models. A higher-order approximation of non-linear models is obtained by the unscented Kalman filter described in Section 2.3.3. After elaborating on sequential Monte Carlo simulation, including sequential importance sampling in Section 2.4.1 and resampling in Section 2.4.2, we focus on the importance of the proposal distribution in Section 2.4.3. In order to drop the Gaussian assumption and allow for arbitrary, multi-modal distributions, we introduce the generic particle filter in Section 2.5.1, followed by two concrete implementations, namely the bootstrap filter in Section 2.5.2 and the unscented particle filter in Section 2.5.3. Finally, we will close this chapter with basic concepts of interval analysis in Section 2.6.1 including the notion of constraint satisfaction problems and set inversion problems in Sections 2.6.2 and 2.6.4, respectively. Both types of problems will be encountered in the following Chapter 3 and will be tackled with either the `HC4` contractor described in Section 2.6.3 or the Set Inverter via Interval Analysis presented in Section 2.6.5.

2.1 THE FILTERING PROBLEM

Consider, as an example involving filter theory, the discrete-time dynamical system depicted in Figure 1. The desired state vector of the system, x_k, at the discrete time step k, is usually hidden and can only be observed by indirect measurements z_k that are a function of x_k and subject to noise. Equally, the equation describing the evolution of the

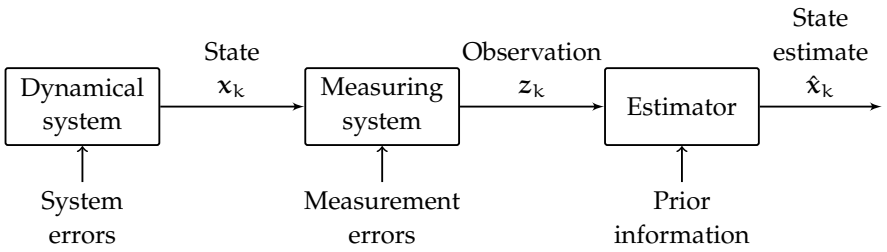


Figure 1: Block diagram depicting the components involved in state estimation of a discrete-time dynamical system [34].

state $\mathbf{x}_k$ is usually subject to errors, caused by effects not accounted for in the model. The dynamical system may be an underwater robot, in which case the elements of the state vector are constituted by its position and velocity, while the measuring system may be an inertial measurement unit producing the observation vector $\mathbf{z}_k$. The requirement of the filter is to deliver a reliable estimate $\hat{\mathbf{x}}_k$ of the actual state, by taking the measurement as well as prior information into account.

Assuming a stationary stochastic process with known statistical parameters as the mean and correlation function of the useful signal and the unwanted additive noise, the solution to the filtering problem is commonly known as the *Wiener filter*. Yet, since the Wiener filter requires a priori information about the statistics of the data to be processed, it may not be optimal for non-stationary processes. For such an environment, in which the statistics are time-varying, it needs a filter that constantly adapts its parameters to optimise its output. A so-called *adaptive filter* is a self-designing system that relies, in contrast to the non-recursive Wiener filter, on a recursive algorithm, allowing the filter to perform satisfactorily, even if there is no complete knowledge of the relevant signal characteristics. Provided the variations in the statistics of the input data are sufficiently slow, the algorithm can track time variations and is thus suitable for non-stationary environments. In a stationary environment it converges to the optimum Wiener solution in some statistical sense after successive iterations.

Generic State-space Model

Now, let us consider the generic stochastic filtering problem in a dynamic state-space form:

$$\mathbf{x}_k = \boldsymbol{\phi}_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \boldsymbol{w}_{k-1}), \quad k > 0\,. \tag{1}$$

Here, $\mathbf{x}_k \in \mathbb{R}^{n_x}$ is the state vector to be estimated, k denotes the time step, and $\boldsymbol{\phi}_{k-1} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_x}$ is the known, possibly non-linear state transition function at time $k-1$. The control vector $\mathbf{u}_{k-1} \in \mathbb{R}^{n_u}$ represents an exogenous input to the system and $\boldsymbol{w}_{k-1} \in \mathbb{R}^{n_w}$ represents a white noise sequence, usually referred to

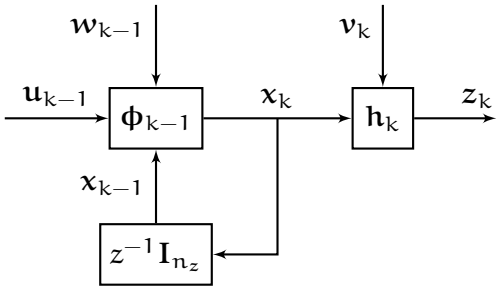


Figure 2: Block diagram depicting a non-linear discrete-time dynamical system, its internal state x_k, and its observation z_k.

as the process noise. The state vector is related to the observation or measurement of the process, $z_k \in \mathbb{R}^{n_z}$, by

$$z_k = h_k(x_k, v_k), \quad k > 0\,, \tag{2}$$

where $h_k : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \to \mathbb{R}^{n_z}$ is a known, possibly non-linear transformation from state variables to measurement variables and $v_k \in \mathbb{R}^{n_v}$ represents a white noise sequence, usually referred to as the measurement noise. Note that here we do not assume additivity of the noise sources. A block diagram of the non-linear discrete-time dynamic system and its observation is depicted in Figure 2, where z^{-1} denotes the unit-delay and $\mathbf{I}_{n_x}$ the $n_x \times n_x$ identity matrix.

2.2 RECURSIVE BAYESIAN ESTIMATION

Provided the system dynamics model and the measurement model can be expressed in a probabilistic form, a Bayesian approach to solving the filtering problem may be adopted. The formal *Bayesian filter* constitutes a general unifying framework for sequential state estimation, at least in a conceptual sense [35]. We will only go into detail as far as necessary for the treatment of the subject matter of this thesis. A complete mathematical derivation of the Bayesian filter can be found in [36].

The system model given by Equation 1 implicitly assumes that the state x_k depends only on the immediate past state x_{k-1} and the control input u_{k-1}. Thus, it defines a first-order discrete Markov process and has an equivalent probabilistic description given by the conditional probability distribution $p(x_k \,|\, x_{k-1}, u_{k-1})$, with the initial state distributed according to

$$p(x_0 \,|\, z_0) = p(x_0)\,, \tag{3}$$

where z_0 denotes the empty measurement.

Likewise, the measurement model given by Equation 2 has an equivalent probabilistic description given by the conditional probability density $p(z_k \,|\, x_k)$. Here, we implicitly assume that if we knew the

state $\mathbf{x}_k$ and were to predict the measurement $\mathbf{y}_k$, no past measurement or control input would provide us additional information. This conditional independence given the state, the Markov property, and the fact that we cannot observe the state directly let us describe our system using the hidden Markov model depicted in Figure 3. In the remainder of this chapter, we will use the notation below:

$\mathbf{X}_k$	*sequence of states*, denoting $\{\mathbf{x}_i\}_{i=0}^{k}$.
$\mathbf{U}_k$	*sequence of control inputs*, denoting $\{\mathbf{u}_i\}_{i=0}^{k}$.
$\mathbf{Z}_k$	*sequence of observations*, denoting $\{\mathbf{z}_i\}_{i=1}^{k}$.
$p(\mathbf{x}_k \mid \mathbf{Z}_{k-1}, \mathbf{U}_{k-1})$	*predictive distribution* of the state $\mathbf{x}_k$ at the current time k, given the entire sequence of observations and the entire sequence of control inputs up to and including time $k-1$.
$p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})$	*posterior distribution* of the current state $\mathbf{x}_k$, given the entire sequence of observations up to and including the current time k and the entire sequence of control inputs up to and including time $k-1$; this distribution is commonly referred to as simply the posterior.
$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_{k-1})$	*state-transition distribution* of the current state $\mathbf{x}_k$, given the immediate past state $\mathbf{x}_{k-1}$ and control input $\mathbf{u}_{k-1}$; this distribution is defined in terms of the system model and is commonly referred to as the transition prior or simply the prior.
$p(\mathbf{z}_k \mid \mathbf{x}_k)$	*likelihood function* of the current observation $\mathbf{z}_k$, given the current state $\mathbf{x}_k$. This function is defined in terms of the observation model.

Now, the aim of the Bayesian filter is to determine the posterior distribution $p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})$, which embodies the entire knowledge that we have about the state $\mathbf{x}_k$ at time k, after being given all the control inputs $\mathbf{U}_{k-1}$ and having received the entire observation sequence $\mathbf{Z}_k$. Given this probability distribution, we can determine an optimal estimator under a specified performance criterion, as for instance the minimum mean-squared error estimator. Then, the optimal state estimate $\hat{\mathbf{x}}_k$ is determined by the mean of the posterior probability distribution,

$$\hat{\mathbf{x}}_k = \mathbb{E}\left[\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1}\right] = \int \mathbf{x}_k \, p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1}) \, d\mathbf{x}_k \,. \tag{4}$$

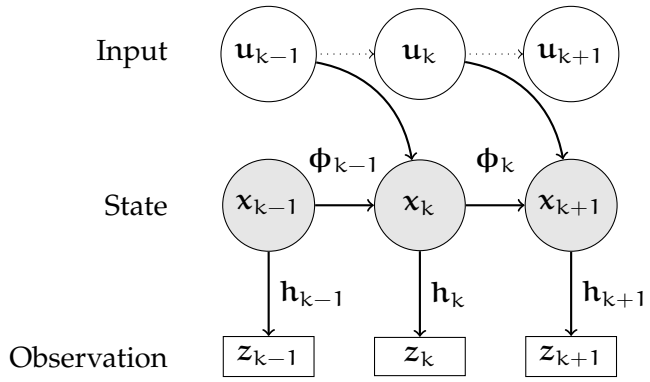


Figure 3: Hidden Markov model of a non-linear discrete-time dynamical system, characterising the evolution of the control inputs, states, and observations.

In order to assess the confidence in the estimated state, we compute the covariance matrix as

$$\begin{aligned}\mathbf{P}_k &= \mathbb{E}\Big[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^T\Big] \\ &= \int (\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^T p(\mathbf{x}_k \,|\, \mathbf{Z}_k, \mathbf{U}_{k-1})\, d\mathbf{x}_k\,.\end{aligned} \tag{5}$$

The posterior distribution can be constructed recursively in two steps, namely by a prediction and an update operation:

1. *Time update*: given the observation sequence $\mathbf{Z}_{k-1}$ and the control sequence $\mathbf{U}_{k-2}$, the predictive distribution is computed according to the Chapman-Kolmogorov identity:

$$\begin{aligned}\underbrace{p(\mathbf{x}_k \,|\, \mathbf{Z}_{k-1}, \mathbf{U}_{k-1})}_{\substack{\text{Predictive}\\ \text{distribution}}} &= \int p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \mathbf{Z}_{k-1}, \mathbf{U}_{k-1}) \\ &\qquad \cdot p(\mathbf{x}_{k-1} \,|\, \mathbf{Z}_{k-1}, \mathbf{U}_{k-1})\, d\mathbf{x}_{k-1} \\ &= \int \underbrace{p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \mathbf{u}_{k-1})}_{\text{Prior}} \\ &\qquad \cdot \underbrace{p(\mathbf{x}_{k-1} \,|\, \mathbf{Z}_{k-1}, \mathbf{U}_{k-2})}_{\text{Old posterior}}\, d\mathbf{x}_{k-1}\,.\end{aligned} \tag{6}$$

   Here, we used the Markov property:

$$p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \mathbf{Z}_{k-1}, \mathbf{U}_{k-1}) = p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \mathbf{u}_{k-1})\,, \tag{7}$$

   and the fact that the old posterior is conditionally independent of future control inputs, so that the following holds:

$$p(\mathbf{x}_{k-1} \,|\, \mathbf{Z}_{k-1}, \mathbf{U}_{k-1}) = p(\mathbf{x}_{k-1} \,|\, \mathbf{Z}_{k-1}, \mathbf{U}_{k-2})\,. \tag{8}$$

2. *Measurement update*: exploiting the current observation z_k and applying Bayes' theorem, we can compute the updated posterior as follows:

$$\begin{aligned}\underbrace{p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})}_{\substack{\text{Updated}\\ \text{posterior}}} &= p(\mathbf{x}_k \mid z_k, \mathbf{Z}_{k-1}, \mathbf{U}_{k-1}) \\ &= \frac{p(z_k \mid \mathbf{x}_k, \mathbf{Z}_{k-1}, \mathbf{U}_{k-1}) p(\mathbf{x}_k \mid \mathbf{Z}_{k-1}, \mathbf{U}_{k-1})}{p(z_k \mid \mathbf{Z}_{k-1}, \mathbf{U}_{k-1})} \\ &= \underbrace{p(z_k \mid \mathbf{x}_k)}_{\substack{\text{Likelihood}\\ \text{function}}} \underbrace{p(\mathbf{x}_k \mid \mathbf{Z}_{k-1}, \mathbf{U}_{k-1})}_{\substack{\text{Predictive}\\ \text{distribution}}} \eta^{-1} ,\end{aligned} \tag{9}$$

where the normalising constant η is given by

$$\begin{aligned}\eta &= p(z_k \mid \mathbf{Z}_{k-1}, \mathbf{U}_{k-1}) \\ &= \int p(z_k \mid \mathbf{x}_k) p(\mathbf{x}_k \mid \mathbf{Z}_{k-1}, \mathbf{U}_{k-1}) d\mathbf{x}_k .\end{aligned} \tag{10}$$

Note how in the last transformation step of Equation 9 we used conditional independence given the state, that is

$$p(z_k \mid \mathbf{x}_k, \mathbf{Z}_{k-1}, \mathbf{U}_{k-1}) = p(z_k \mid \mathbf{x}_k) . \tag{11}$$

The Bayesian filter is the optimal conceptual solution to the recursive estimation problem. However, due to the multi-dimensional integration, a closed-form algorithm can only be obtained in a few special cases. If the probability distributions $p(\mathbf{x}_0)$, $p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_{k-1})$, and $p(z_k \mid \mathbf{x}_k)$ are Gaussian and the dynamic system is described by a linear model, the posterior distribution remains Gaussian and the Equations 6 and 9 reduce to the celebrated Kalman filter, which is described in the following section. Figure 4 compares different Bayesian filters that are discussed in further detail below.

2.3 KALMAN FILTERS

The *Kalman filter* provides an efficient means to analytically compute the evolving sequence of posterior distributions of a linear dynamic system that is perturbed by additive white Gaussian noise [37]. Named after Rudolf E. Kalman, who 1960 published his famous paper describing a recursive solution to the discrete-data linear filtering problem [38], the Kalman filter has been the subject of extensive research, which is due, to a large extent, to the advances in digital computing [39]. The Kalman filter and its many variations find applications in radar tracking, navigation, and orientation estimation, among others. Zarchan

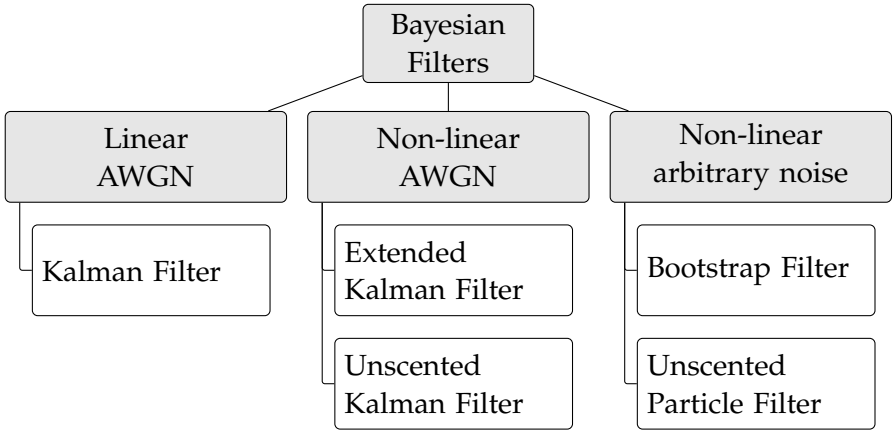


Figure 4: Comparison of different Bayesian filters used in self-localisation of mobile robots. AWGN denotes additive white Gaussian noise.

and Musoff stated in [40]: "With the possible exception of the fast Fourier transform, Kalman filtering is probably the most important algorithmic technique ever devised." We will now proceed to formally introduce the Kalman filter, followed by two of its variations for non-linear models in the subsequent sections.

2.3.1 *The Kalman Filter*

Let $\mathbf{x}_k \in \mathbb{R}^{n_x}$ be the state vector of a discrete-time controlled process, governed by the *linear* stochastic difference equation

$$\mathbf{x}_k = \mathbf{\Phi}_{k-1}\mathbf{x}_{k-1} + \mathbf{B}_{k-1}\mathbf{u}_{k-1} + \boldsymbol{w}_{k-1}\,, \tag{12}$$

where the index k again denotes discrete time. The $n_x \times n_x$ state transition matrix $\mathbf{\Phi}_{k-1}$ relates the state at the previous time step $k-1$ to the state at the current step k and the $n_x \times n_u$ matrix $\mathbf{B}_{k-1}$ relates the known, optional control input $\mathbf{u}_{k-1} \in \mathbb{R}^{n_u}$ to the state $\mathbf{x}_k$. Let $\boldsymbol{z}_k \in \mathbb{R}^{n_z}$ denote the measurement vector of this process, which is related to the state by the linear measurement model

$$\boldsymbol{z}_k = \mathbf{H}_k\mathbf{x}_k + \boldsymbol{v}_k\,, \tag{13}$$

where the $n_z \times n_x$ measurement matrix $\mathbf{H}_k$ relates the state $\mathbf{x}_k$ to the measurement $\boldsymbol{z}_k$. The $n_x \times 1$ vector $\boldsymbol{w}_k$ and the $n_z \times 1$ vector $\boldsymbol{v}_k$ in Equation 12 and 13 represent the additive process and measurement noise, respectively, modelled as zero-mean, Gaussian white noise,

$$\boldsymbol{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k)\,, \tag{14}$$

$$\boldsymbol{v}_k \sim \mathcal{N}(0, \mathbf{R}_k)\,, \tag{15}$$

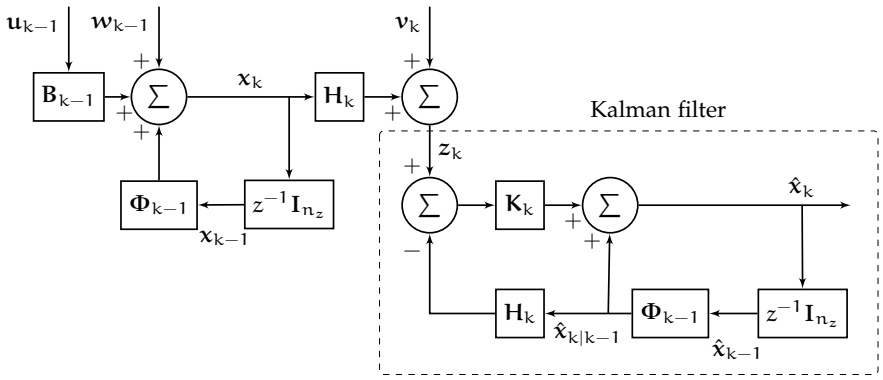


Figure 5: Block diagram depicting the relation between a linear, discrete-time dynamical system, its observation z_k, and the Kalman filter.

with the *process noise covariance matrix* $\mathbf{Q}_k$ and the *measurement noise covariance matrix* $\mathbf{R}_k$.

We define the vector $\hat{\mathbf{x}}_{k|k-1} \in \mathbb{R}^{n_x}$ as the *a priori* state estimate, representing knowledge of the process prior to step k, given by

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{\Phi}_{k-1}\hat{\mathbf{x}}_{k-1} + \mathbf{B}_{k-1}\mathbf{u}_{k-1}\,, \tag{16}$$

and $\hat{\mathbf{x}}_k \in \mathbb{R}^{n_x}$ as the *a posteriori* state estimate at step k, after having received the measurement z_k, given by

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\big(z_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}\big)\,. \tag{17}$$

The term $[z_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}]$ is called the measurement *innovation* or *residual*. It reflects the discordance between the predicted measurement $\mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}$ and the actual measurement z_k. The $n_x \times n_z$ matrix $\mathbf{K}_k$ is termed the *Kalman gain* and is given by

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T\big(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k\big)^{-1}\,, \tag{18}$$

with the *a priori error covariance matrix*

$$\mathbf{P}_{k|k-1} = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1}\mathbf{\Phi}_{k-1}^T + \mathbf{Q}_{k-1} \tag{19}$$

and the *a posteriori error covariance matrix*

$$\mathbf{P}_k = \big(\mathbf{I}_n - \mathbf{K}_k\mathbf{H}_k\big)\mathbf{P}_{k|k-1}\,. \tag{20}$$

Note that the Gaussian posterior distribution is fully determined by its mean $\hat{\mathbf{x}}_k$ and covariance $\mathbf{P}_k$.

Figure 5 illustrates the relation of the Kalman filter to the linear discrete-time dynamical system, where z^{-1} denotes the unit-delay and $\mathbf{I}_{n_x}$ the $n_x \times n_x$ identity matrix. In accordance with the formal Bayesian filter, the Kalman filter equations can be divided into two groups: *time update* Equations 16, 19 and *measurement update* Equations 17 , 18, and 20, as shown in Figure 6, which depicts the 'predict and correct' behaviour of the filter algorithm. After an initialisation

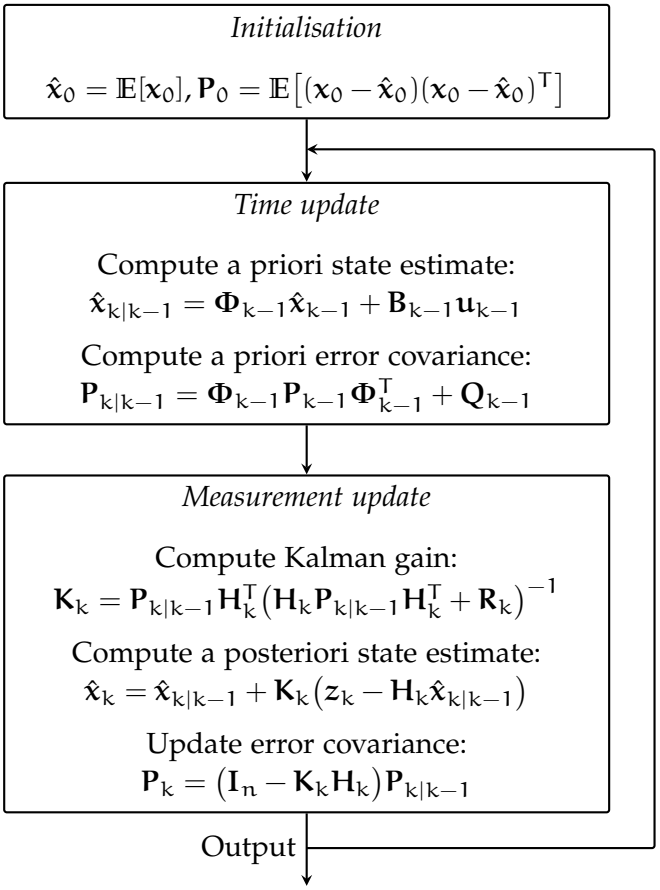


Figure 6: Operation cycle of the Kalman filter, illustrating its 'predict and correct' behaviour.

step, the time update and measurement update steps are carried out recursively at every time step.

The Kalman filter represents an optimal solution to the recursive Bayesian state estimation problem but assumes a linear model and Gaussian noise. Since for many practical problems these assumptions do not hold, a variety of state estimators have been proposed to approximate solutions to the non-linear, possibly non-Gaussian state estimation problem, which has shown to be difficult to solve analytically. An important representative of this class of filters is the extended Kalman filter.

2.3.2 *The Extended Kalman Filter*

The Kalman filter may be modified so as to make it applicable to the state estimation of systems with *non-linear* state dynamics, that is systems that can be described by a model of the following form:

$$\mathbf{x}_k = \boldsymbol{\phi}_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) + \boldsymbol{w}_{k-1}, \quad \boldsymbol{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k)\,. \tag{21}$$

The state transition function $\boldsymbol{\phi}_{k-1} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ relates the state at the previous time step $k-1$ to the current time step k, taking into account the exogenous control input $\mathbf{u}_{k-1}$. The possibly *non-linear* transformation from state variables to measurement variables, $\mathbf{h}_k : \mathbb{R}^{n_x} \to \mathbb{R}^{n_z}$, is given by

$$\mathbf{z}_k = \mathbf{h}_k(\mathbf{x}_k) + \boldsymbol{\nu}_k, \quad \boldsymbol{\nu}_k \sim \mathcal{N}(0, \mathbf{R}_k)\,. \tag{22}$$

Note that, as opposed to the generic state-space model given by Equation 1 and 2, again we assume additive zero-mean white Gaussian noise in both Equations 21 and 22.

Linearisation

Some non-linear problems can be deemed *quasi-linear*, which means that a variation of the respective non-linear functions $\boldsymbol{\phi}_k$ and $\mathbf{h}_k$ are predominantly linear about a value $\mathbf{x}_0$. Assuming that $\boldsymbol{\phi}_k$ and $\mathbf{h}_k$ are differentiable at $\mathbf{x}_0$, they can be approximated as follows:

$$\boldsymbol{\phi}_k(\mathbf{x}_0 + d\mathbf{x}, \mathbf{u}) \approx \boldsymbol{\phi}_k(\mathbf{x}_0, \mathbf{u}) + d\mathbf{x}\, \frac{\partial \boldsymbol{\phi}_k(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}}\bigg|_{\mathbf{x}=\mathbf{x}_0, \mathbf{u}}\,, \tag{23}$$

$$\mathbf{h}_k(\mathbf{x}_0 + d\mathbf{x}) \approx \mathbf{h}_k(\mathbf{x}_0) + d\mathbf{x}\, \frac{\partial \mathbf{h}_k(\mathbf{x})}{\partial \mathbf{x}}\bigg|_{\mathbf{x}=\mathbf{x}_0}\,. \tag{24}$$

A first-order Taylor series expansion of the state-space model at each time instant around the most recent state estimate allows us to use the standard Kalman filter equations stated in Section 2.3.1. The resulting filter is referred to as the *extended Kalman filter* (EKF). It linearises the functions $\boldsymbol{\phi}_{k-1}$ and $\mathbf{h}_k$ using their respective Jacobian matrices

$$\boldsymbol{\Phi}_{k-1}^{[1]} = \frac{\partial \boldsymbol{\phi}_{k-1}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}}\bigg|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1}, \mathbf{u}=\mathbf{u}_{k-1}} \tag{25}$$

and

$$\mathbf{H}_k^{[1]} = \frac{\partial \mathbf{h}_k(\mathbf{x})}{\partial \mathbf{x}}\bigg|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}}\,, \tag{26}$$

where the superscript $[1]$ denotes the *first-order* approximation. The ij-th entry of $\boldsymbol{\Phi}_{k-1}^{[1]}$ is equal to the partial derivative of the i-th component of $\boldsymbol{\phi}_{k-1}(\mathbf{x})$ with respect to the j-th component of $\mathbf{x}$. The derivatives are evaluated at $\mathbf{x} = \hat{\mathbf{x}}_{k-1}$ and $\mathbf{u} = \mathbf{u}_{k-1}$. Likewise, the ij-th entry of $\mathbf{H}_k^{[1]}$ is equal to the partial derivative of the i-th component of $\mathbf{h}_k(\mathbf{x})$ with respect to the j-th component of $\mathbf{x}$. The derivatives are evaluated at $\mathbf{x} = \hat{\mathbf{x}}_{k|k-1}$.

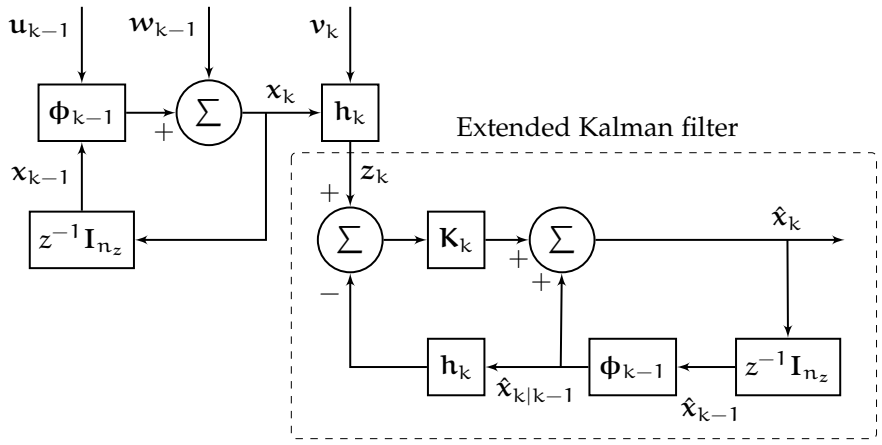


Figure 7: Block diagram depicting the relation between a non-linear discrete-time dynamical system, its observation $\boldsymbol{z}_k$, and the extended Kalman filter.

Extended Kalman Filter Equations

In structure similar to the Kalman filter Equations 16 and 17, the a priori state estimate is given by

$$\hat{\mathbf{x}}_{k|k-1} = \boldsymbol{\phi}_{k-1}(\boldsymbol{x}_{k-1}, \mathbf{u}_{k-1}) \tag{27}$$

and the a posteriori estimate, conditioned on the current measurement, is given by

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\big(\boldsymbol{z}_k - \mathbf{h}_k(\hat{\mathbf{x}}_{k|k-1})\big)\,. \tag{28}$$

The corresponding a priori error covariance matrix $\mathbf{P}_{k|k-1}$, the Kalman gain $\mathbf{K}_k$, and the a posteriori covariance matrix $\mathbf{P}_k$ are computed as follows:

$$\mathbf{P}_{k|k-1} = \boldsymbol{\Phi}^{[1]}_{k-1}\mathbf{P}_{k-1}\boldsymbol{\Phi}^{[1]\mathsf{T}}_{k-1} + \mathbf{Q}_{k-1}\,, \tag{29}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}^{[1]\mathsf{T}}_k\big(\mathbf{H}^{[1]}_k\mathbf{P}_{k|k-1}\mathbf{H}^{[1]\mathsf{T}}_k + \mathbf{R}_k\big)^{-1}\,, \tag{30}$$

$$\mathbf{P}_k = \big(\mathbf{I}_n - \mathbf{K}_k\mathbf{H}^{[1]}_k\big)\mathbf{P}_{k|k-1}\,. \tag{31}$$

Figure 7 illustrates the relation of the extended Kalman filter to the non-linear discrete-time dynamical system, where z^{-1} denotes the unit-delay and $\mathbf{I}_{n_x}$ the $n_x \times n_x$ identity matrix. Figure 8 illustrates the 'predict and correct' behaviour of the extended Kalman filter algorithm. After an initialisation step, the *time update* and *measurement update* steps are carried out recursively at every time step. In order to linearise the state-space model at each time instant around the most recent state estimate, additionally, the Jacobian matrices have to be computed, which can prove to be computationally demanding for high dimensional systems.

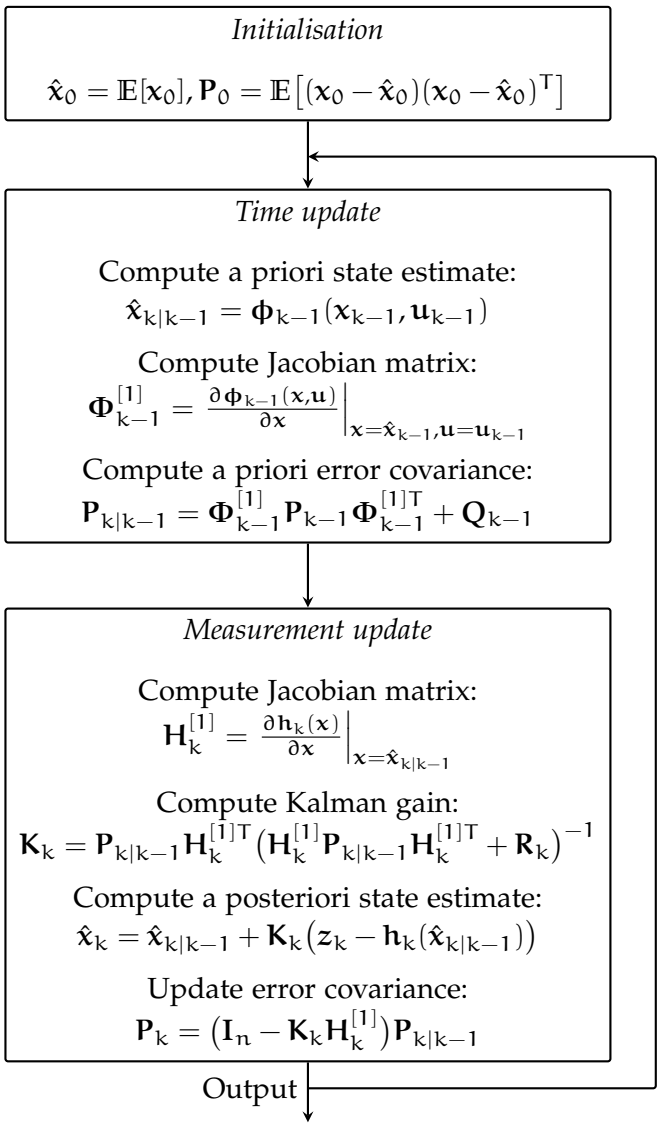


Figure 8: Operation cycle of the extended Kalman filter, illustrating its 'predict and correct' behaviour.

Extended Kalman filtering is commonly used and was, in fact, the first successful application of the Kalman filter [41]. Unlike its linear counterpart, the extended Kalman filter may not necessarily be an optimal estimator. Owing to its local linearisation the EKF may quickly diverge if the model is highly non-linear. This limitation may be overcome using a higher-order approximation, which characterises the unscented Kalman filter described in the following section.

2.3.3 *The Unscented Kalman Filter*

The *unscented Kalman filter* (UKF) is a derivative-free state estimation approach first proposed by Julier and Uhlmann in [42] and further developed by Wan and Van Der Merwe in [43]. The Gaussian state

distribution is represented using a minimal set of carefully chosen sample points around the mean, called *sigma points*, which capture the true mean and covariance of the process. When propagated through the non-linear system, the sigma points capture the posterior mean and covariance accurate to the second-order Taylor polynomial for any non-linearity [44]. Whilst providing superior performance, the computational complexity of the UKF is the same order as that of the EKF. In the following section, we will introduce the underlying unscented transformation. A comprehensive description including an extension of the unscented Kalman filter to a broader class of estimation problems can be found in [45].

Unscented Transformation

The *unscented transformation* [46] and *scaled unscented transformation* [47] was developed by Julier and Uhlmann and is motivated by their following intuition: "With a fixed number of parameters it should be easier to approximate a Gaussian distribution than it is to approximate an arbitrary nonlinear function." Thus, the continuous Gaussian distribution is approximated using a discrete distribution having the same first and second-order moments.

Given an n-dimensional Gaussian random variable $\mathbf{x}$, with mean $\bar{\mathbf{x}}$ and covariance $\mathbf{P}_{\mathbf{x}}$, we represent its distribution using an ensemble $\mathcal{X}$ of $2n+1$ sigma points $\mathcal{X}_i$, with

$$\mathcal{X}_0 = \bar{\mathbf{x}}\,, \tag{32}$$

$$\mathcal{X}_i = \bar{\mathbf{x}} + \left(\sqrt{(n+\lambda)\mathbf{P}_{\mathbf{x}}}\right)_i, \qquad i \in \{1,\dots,n\}, \tag{33}$$

$$\mathcal{X}_i = \bar{\mathbf{x}} - \left(\sqrt{(n+\lambda)\mathbf{P}_{\mathbf{x}}}\right)_{i-n}, \quad i \in \{n+1,\dots,2n\}, \tag{34}$$

where $\left(\sqrt{(n+\lambda)\mathbf{P}_{\mathbf{x}}}\right)_i$ is the i-th column of the matrix square root of $(n+\lambda)\mathbf{P}_{\mathbf{x}}$, obtained, for instance, by a Cholesky decomposition. The scaling parameter λ is given by

$$\lambda = \alpha^2(n+\kappa) - n, \quad \alpha,\kappa \in \mathbb{R}, \quad 0 \leqslant \alpha \leqslant 1, \quad \kappa \geqslant 1\,, \tag{35}$$

in which α determines the spread of the sigma points around the mean [48] and $\kappa \geqslant 1$ guarantees positive semi-definiteness of the covariance matrix [49]. According to [45], typical recommendations are $\kappa = 0$ and $10^{-4} \leqslant \alpha \leqslant 1$.

In order to compute an estimate of the mean and covariance of an m-dimensional random variable $\mathbf{y}$, which is related to $\mathbf{x}$ by a non-linear transformation $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^m$, so that

$$\mathbf{y} = \mathbf{g}(\mathbf{x})\,, \tag{36}$$

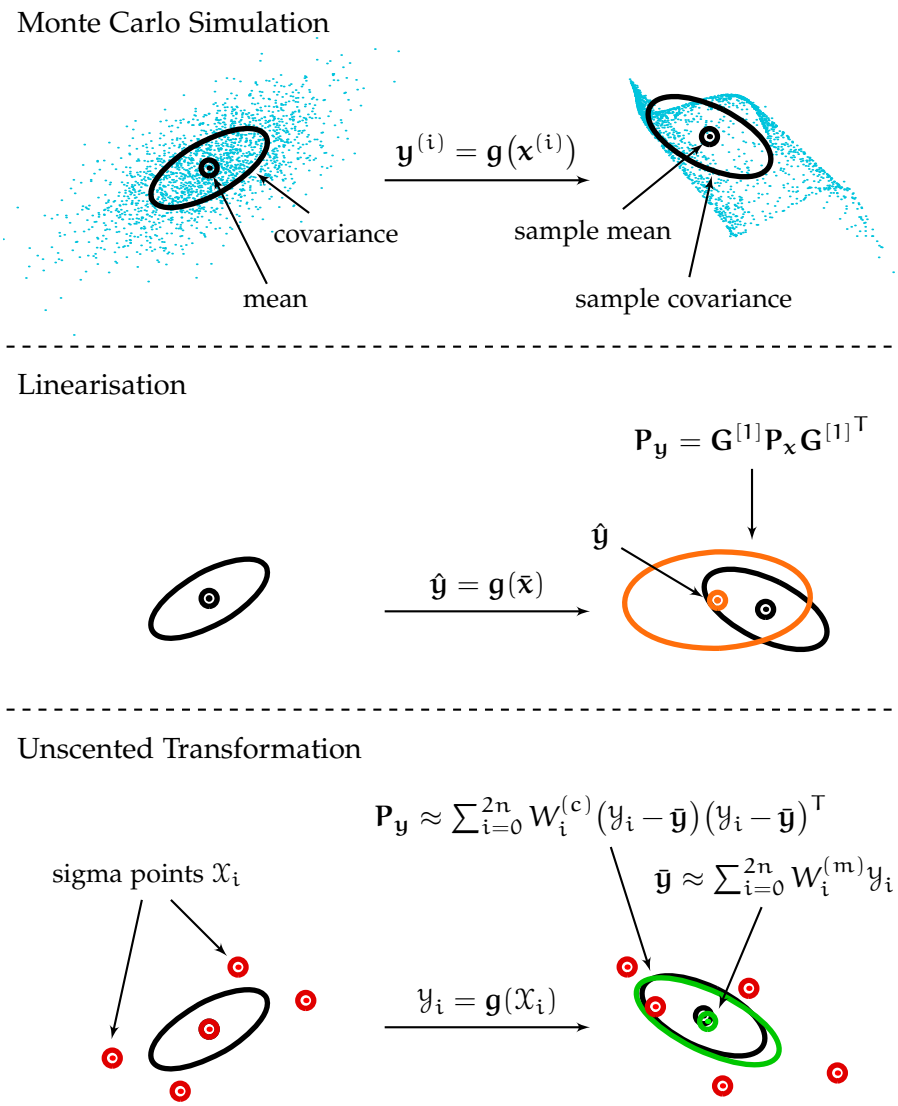


Figure 9: The unscented transformation in comparison with Monte Carlo sampling and linearisation about the mean [45]. Above, a cloud $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ of $N = 200$ samples drawn from a two-dimensional Gaussian distribution is propagated individually through a highly nonlinear function $\mathbf{g}$. After the transformation the empirical sample mean and covariance is computed, respectively. In the middle, the mean $\bar{\mathbf{x}}$ is propagated through $\mathbf{g}$ to obtain an estimate $\hat{\mathbf{y}}$ of the true mean $\bar{\mathbf{y}}$, and the covariance is approximated through a linearisation of $\mathbf{g}$ about $\bar{\mathbf{x}}$, using its Jacobian evaluated at $\bar{\mathbf{x}}$, which is denoted by $\mathbf{G}^{[1]}$. Below, the set of sigma points $\{\mathcal{X}_i\}_{i=1}^{5}$ capturing the true mean and covariance of the distribution of $\mathbf{x}$ are transformed independently and the mean and covariance is estimated, respectively. As can be seen, the result of the unscented transformation approximates the true mean and covariance better than the linearisation approach.

the sigma points $\mathcal{X}_i \in \boldsymbol{\mathcal{X}}$ are independently propagated through $\mathbf{g}$,

$$\mathcal{Y}_i = \mathbf{g}(\mathcal{X}_i), \quad i \in \{0, \dots, 2n\}, \tag{37}$$

to obtain a set $\boldsymbol{\mathcal{Y}}$ of transformed sigma points $\mathcal{Y}_i$.

The mean $\bar{\mathbf{y}}$ and the covariance $\mathbf{P}_\mathbf{y}$ of the random variable $\mathbf{y}$ are approximated as the weighted sample mean and covariance of the posterior sigma points, respectively,

$$\bar{\mathbf{y}} \approx \sum_{i=0}^{2n} W_i^{(m)} \mathcal{Y}_i \,, \tag{38}$$

$$\mathbf{P}_\mathbf{y} \approx \sum_{i=0}^{2n} W_i^{(c)} (\mathcal{Y}_i - \bar{\mathbf{y}})(\mathcal{Y}_i - \bar{\mathbf{y}})^\mathsf{T} \,. \tag{39}$$

The weights for the mean, $W_i^{(m)}$, and for the covariance, $W_i^{(c)}$, are given by

$$W_0^{(m)} = \frac{\lambda}{n+\lambda} \,, \tag{40}$$

$$W_0^{(c)} = \frac{\lambda}{n+\lambda} + (1 - \alpha^2 + \beta) \,, \tag{41}$$

$$W_i^{(m)} = W_i^{(c)} = \frac{1}{2(n+\lambda)}, \quad i \in \{1, \dots, 2n\}. \tag{42}$$

The real parameter β is used to incorporate information about the probability distribution of $\mathbf{x}$, where $\beta = 2$ is optimal if the distribution is Gaussian [50].

Figure 9 shows the unscented transformation for a two-dimensional random vector in comparison with Monto-Carlo sampling, which will be introduced in detail in Section 2.4, and the linearisation approach used by the extended Kalman filter. Note that only five sigma points are required to capture the mean and covariance of the distribution of $\mathbf{x}$. The mean and covariance of the independently transformed sigma points approximate the true mean and covariance up to the second order. Julier and Uhlmann mention the following significant advantages of the unscented transformation compared to the linearisation used in the EKF:

- It is not necessary to compute Jacobians.
- In the prediction stage only standard linear algebra operations like matrix square roots, outer products, and matrix and vector summations are required.
- The number of computations, including an efficient computation of the matrix square root, scale with dimensions at the same rate as linearisation.

- Constraints can be seamlessly incorporated by applying them to each of the projected sigma points $\mathcal{Y}_i$.

Unscented Kalman Filter Equations

Given a discrete-time dynamic process and its ovservation governed by Equation 21 and 22, respectively, at each time step k we compute the set of $2n_x + 1$ sigma points using the old posterior mean $\hat{\mathbf{x}}_{k-1}$ and covariance $\mathbf{P}_{k-1}$,

$$\mathcal{X}_{0,k-1} = \hat{\mathbf{x}}_{k-1}\,, \tag{43}$$

$$\mathcal{X}_{i,k-1} = \hat{\mathbf{x}}_{k-1} + \gamma\left(\sqrt{\mathbf{P}_{k-1}}\right)_i, \qquad i \in \{1,\dots,n_x\}, \tag{44}$$

$$\mathcal{X}_{i,k-1} = \hat{\mathbf{x}}_{k-1} - \gamma\left(\sqrt{\mathbf{P}_{k-1}}\right)_{i-n}, \quad i \in \{n_x+1,\dots,2n_x\}, \tag{45}$$

with

$$\gamma = \sqrt{(n_x + \lambda)}\,. \tag{46}$$

For the sake of brevity, in the following, the entire set of sigma points is denoted as

$$\begin{aligned}\boldsymbol{\mathcal{X}}_{k-1} &= \left\{\mathcal{X}_{0,k-1}, \mathcal{X}_{1,k-1}, \dots, \mathcal{X}_{2n_x,k-1}\right\} \\ &= \left\{\mathcal{X}_{0,k-1}\right\} \cup \left\{\mathcal{X}_{1,k-1}, \dots, \mathcal{X}_{n_x,k-1}\right\} \\ &\qquad\qquad \cup \left\{\mathcal{X}_{n_x+1,k-1}, \dots, \mathcal{X}_{2n_x,k-1}\right\} \\ &= \left\{\hat{\mathbf{x}}_{k-1}, \quad \hat{\mathbf{x}}_{k-1} + \gamma\sqrt{\mathbf{P}_{k-1}}, \quad \hat{\mathbf{x}}_{k-1} - \gamma\sqrt{\mathbf{P}_{k-1}}\right\}.\end{aligned} \tag{47}$$

Now we propagate the sigma points individually, taking into account the control input $\mathbf{u}_{k-1}$, to obtain the set of transformed sigma points $\boldsymbol{\mathcal{X}}_{k|k-1}$. Again, more succinctly we write

$$\boldsymbol{\mathcal{X}}_{k|k-1} = \boldsymbol{\phi}_{k-1}\left(\boldsymbol{\mathcal{X}}_{k-1}, \mathbf{u}_{k-1}\right). \tag{48}$$

The a priori state estimate is computed as the weighted sample mean

$$\hat{\mathbf{x}}_{k|k-1} = \sum_{i=0}^{2n} W_i^{(m)} \mathcal{X}_{i,k|k-1} \tag{49}$$

and the a priori error covariance matrix as

$$\mathbf{P}_{k|k-1} = \sum_{i=0}^{2n} W_i^{(c)} \left(\mathcal{X}_{i,k|k-1} - \hat{\mathbf{x}}_{k|k-1}\right)\left(\mathcal{X}_{i,k|k-1} - \hat{\mathbf{x}}_{k|k-1}\right)^T + \mathbf{Q}_k\,. \tag{50}$$

where $\mathbf{Q}_k$ again denotes the process noise covariance matrix. Transforming the a priori sigma points individually using the measurement function $\mathbf{h}_k$ of Equation 22,

$$\mathcal{Z}_{k|k-1} = \mathbf{h}_k(\mathcal{X}_{k|k-1}), \tag{51}$$

lets us compute the predicted measurement as the weighted sample mean of the transformed sigma points,

$$\hat{\boldsymbol{z}}_{k|k-1} = \sum_{i=0}^{2n} W_i^{(m)} \mathcal{Z}_{i,k|k-1}, \quad \mathcal{Z}_{i,k|k-1} \in \mathcal{Z}_{k|k-1}. \tag{52}$$

Now, we can compute the innovation covariance matrix

$$\mathbf{P}_{\tilde{z}_k \tilde{z}_k} = \sum_{i=0}^{2n} W_i^{(c)} \left(\mathcal{Z}_{i,k|k-1} - \hat{\boldsymbol{z}}_{k|k-1}\right)\left(\mathcal{Z}_{i,k|k-1} - \hat{\boldsymbol{z}}_{k|k-1}\right)^T + \mathbf{R}_k \tag{53}$$

and the cross covariance matrix as

$$\mathbf{P}_{\tilde{x}_k \tilde{z}_k} = \sum_{i=0}^{2n} W_i^{(c)} \left(\mathcal{X}_{i,k|k-1} - \hat{\mathbf{x}}_{k|k-1}\right)\left(\mathcal{Z}_{i,k|k-1} - \hat{\boldsymbol{z}}_{k|k-1}\right)^T, \tag{54}$$

where $\mathbf{R}_k$ denotes the measurement noise covariance matrix. All weights W_i in Equations 49–54 are computed according to Equations 40–42. Given the covariance and cross covariance matrix, we can compute the Kalman gain as

$$\mathcal{K}_k = \mathbf{P}_{\tilde{x}_k \tilde{z}_k} \mathbf{P}_{z_k z_k}^{-1}. \tag{55}$$

As with the Kalman filter, the updated state estimate is the predicted state plus the innovation weighted by the Kalman gain,

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k|k-1} + \mathcal{K}_k\left(\boldsymbol{z}_k - \hat{\boldsymbol{z}}_{k|k-1}\right), \tag{56}$$

and the updated error covariance is the a priori error covariance minus the predicted measurement covariance, weighted by the Kalman gain,

$$\mathbf{P}_k = \mathbf{P}_{k|k-1} - \mathcal{K}_k \mathbf{P}_{\tilde{z}_k \tilde{z}_k} \mathcal{K}_k^T. \tag{57}$$

Figure 10 shows the entire unscented Kalman filter cycle, illustrating the computation of sigma points and the 'predict and correct' behaviour. Note that for now we assume additive zero-mean noise sources. In the next section, we will extend the UKF to arbitrary noise sources with Gaussian distributions.

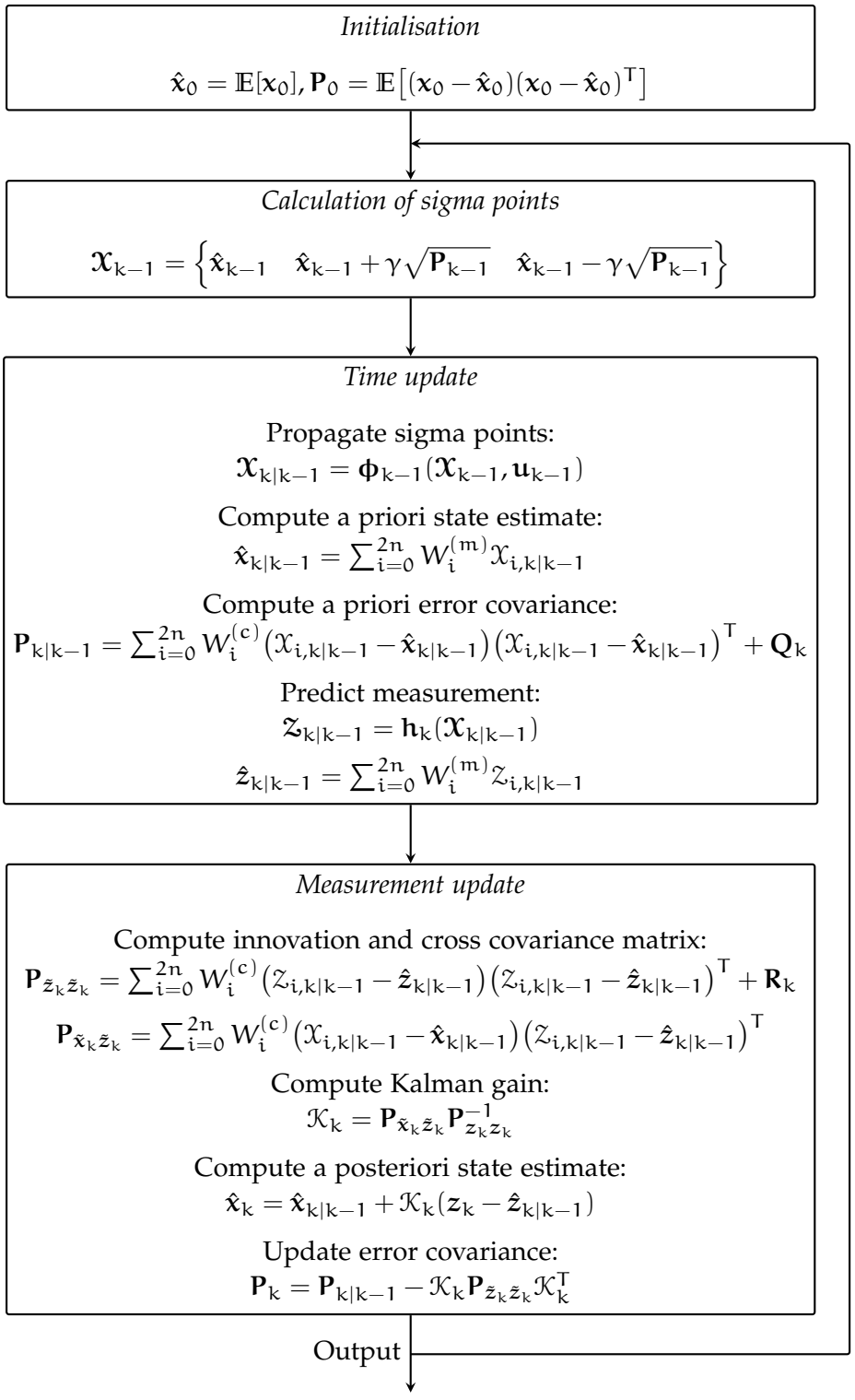


Figure 10: Operation cycle of the unscented Kalman filter for the additive zero-mean noise case, illustrating the computation of sigma points and its 'predict and correct' behaviour.

Non-additive Noise

In the case of non-additive non-zero mean process and measurement noise, the unscented transformation scheme is applied to the *augmented state* $\boldsymbol{x}_k^a \in \mathbb{R}^l$, which is defined as the concatenation of the original state and the noise variables as

$$\boldsymbol{x}_k^a = \left[\boldsymbol{x}_k{}^\mathsf{T}, \boldsymbol{w}_k{}^\mathsf{T}, \boldsymbol{v}_k{}^\mathsf{T}\right]^\mathsf{T}, \tag{58}$$

so that $l = n_x + n_w + n_v$. The corresponding augmented error covariance matrix is given by

$$\mathbf{P}_k^a = \mathbb{E}\left[(\boldsymbol{x}_k^a - \hat{\boldsymbol{x}}_k^a)(\boldsymbol{x}_k^a - \hat{\boldsymbol{x}}_k^a)^\mathsf{T}\right] = \begin{bmatrix} \mathbf{P}_k & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{R}_k \end{bmatrix}, \tag{59}$$

with

$$\hat{\boldsymbol{x}}_k^a = \left[\hat{\boldsymbol{x}}_k^\mathsf{T}, \mathbb{E}\left[\boldsymbol{w}_k\right]^\mathsf{T}, \mathbb{E}\left[\boldsymbol{v}_k\right]^\mathsf{T}\right]^\mathsf{T}. \tag{60}$$

The zeros in bold typeface denote the zero matrix with respective dimension. Equally, every member $\mathcal{X}_{i,k}^a$ of the set $\boldsymbol{\mathcal{X}}_k^a$ of augmented sigma points is constituted of three vector-valued components,

$$\mathcal{X}_{i,k}^a = \left[\mathcal{X}_{i,k}^{\boldsymbol{x}}{}^\mathsf{T}, \mathcal{X}_{i,k}^{\boldsymbol{w}}{}^\mathsf{T}, \mathcal{X}_{i,k}^{\boldsymbol{v}}{}^\mathsf{T}\right]^\mathsf{T}. \tag{61}$$

The vector space $\mathbb{R}^{n_x}$ forms a linear subspace of $\mathbb{R}^l$. We denote the set of all sigma points $\mathcal{X}_{i,k}^{\boldsymbol{x}}$ belonging to this subspace with $\boldsymbol{\mathcal{X}}_k^{\boldsymbol{x}}$. Replacing the superscript $\boldsymbol{x}$ with $\boldsymbol{w}$ and $\boldsymbol{v}$, respectively, denotes the sigma points belonging to the subspaces spanned by the noise vectors, that is $\boldsymbol{\mathcal{X}}_k^{\boldsymbol{w}}$ and $\boldsymbol{\mathcal{X}}_k^{\boldsymbol{v}}$.

The augmented unscented Kalman filter algorithm is in structure similar to the that assuming additive noise in Figure 10, but it uses the augmented state to calculate sigma points, so that the set of sigma points is denoted as

$$\boldsymbol{\mathcal{X}}_{k-1}^a = \left\{\hat{\boldsymbol{x}}_{k-1}^a, \quad \hat{\boldsymbol{x}}_{k-1}^a + \gamma_a\sqrt{\mathbf{P}_{k-1}^a}, \quad \hat{\boldsymbol{x}}_{k-1}^a - \gamma_a\sqrt{\mathbf{P}_{k-1}^a}\right\}, \tag{62}$$

with

$$\gamma_a = \sqrt{(l+\lambda)}\,. \tag{63}$$

Since we assume non-additive noise, we individually propagate the components of the sigma points that are associated with the state according to Equation 1,

$$\boldsymbol{\mathcal{X}}_{k|k-1}^{\boldsymbol{x}} = \boldsymbol{\phi}_{k-1}\left(\boldsymbol{\mathcal{X}}_{k-1}^{\boldsymbol{x}}, \mathbf{u}_{k-1}, \boldsymbol{\mathcal{X}}_{k-1}^{\boldsymbol{w}}\right), \tag{64}$$

and predict the measurement transforming the a priori sigma points according to Equation 2,

$$\mathcal{Z}_{k|k-1} = \mathbf{h}_k\left(\mathcal{X}^{\mathbf{x}}_{k|k-1}, \mathcal{X}^{\mathbf{v}}_{k|k-1}\right). \tag{65}$$

Figure 11 depicts the algorithm in its entirety, including the calculation of augmented sigma points and the predict and update operations that are carried out recursively at each time step.

2.4 SEQUENTIAL MONTE CARLO SIMULATION

If the assumptions of the Kalman filter hold, that is a linear process and measurement model and Gaussian distributions, then no other algorithm can outperform it [51]. Real-life problems, however, may not always be described sufficiently accurate by linear-Gaussian models. *Sequential Monte Carlo* (SMC) methods are a general simulation-based approach that essentially converts the intractable integrals of the Bayesian framework into tractable, finite sums, which converge to the exact solution in the limit. In contrast to the Kalman filtering methods above, they are not subject to any linearity or Gaussianity constraints on the model and entail appealing convergence properties [52]. These benefits, in turn, come along with increased computational cost.

If a sufficiently large number of samples drawn from the desired posterior distribution of the Bayesian framework is available, it is straightforward to approximate the intractable integrals appearing in Equations 6 and 9 in Section 2.2. In *perfect Monte Carlo sampling* we assume that we are able to simulate N independent and identically distributed random samples $\mathbf{x}_k^{(i)}$, drawn from the posterior distribution $p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})$. These random samples are also called *particles*. Each particle $\mathbf{x}_k^{(i)}$ is a concrete instantiation of the state at time k. That is, each particle represents a possible hypothesis as to what the true state of the system may be. An empirical estimate of the posterior distribution can then be computed as

$$p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1}) \approx \frac{1}{N}\sum_{i=1}^{N} \delta\left(\mathbf{x}_k - \mathbf{x}_k^{(i)}\right), \tag{66}$$

where $\delta\left(\mathbf{x}_k - \mathbf{x}_k^{(i)}\right)$ denotes the Dirac delta mass located in $\mathbf{x}_k^{(i)}$. Additionally, any expectation of the form

$$\mathbb{E}_{p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})}\left[f(\mathbf{x}_k)\right] = \int f(\mathbf{x}_k) p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1}) \mathrm{d}\mathbf{x}_k \tag{67}$$

can be approximated as

$$\begin{aligned} \mathbb{E}_{p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})}\left[f(\mathbf{x}_k)\right] &\approx \hat{\mathbb{E}}_{p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})}\left[f(\mathbf{x}_k)\right] \\ &= \frac{1}{N}\sum_{i=1}^{N} f\left(\mathbf{x}_k^{(i)}\right). \end{aligned} \tag{68}$$

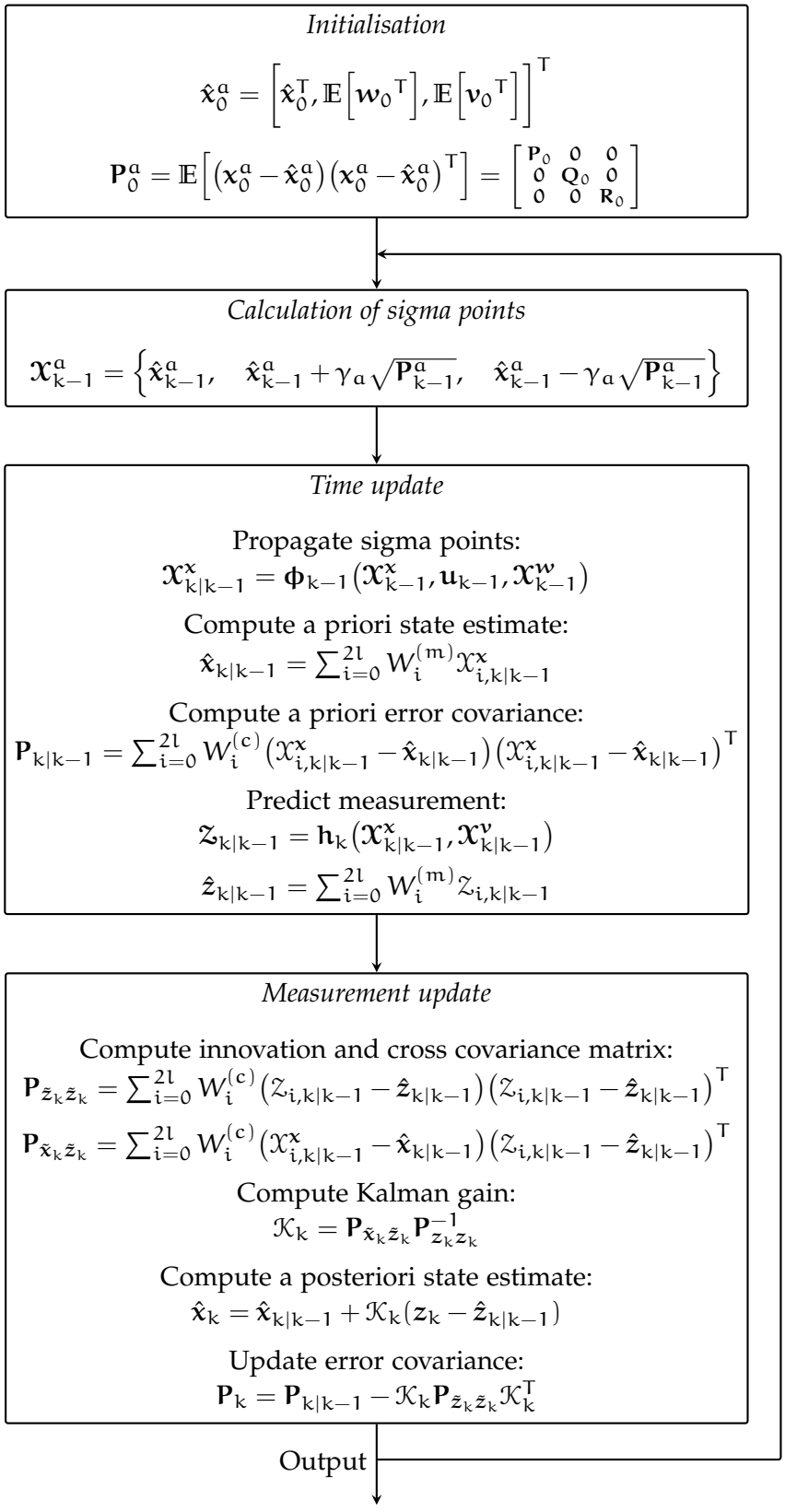


Figure 11: Operation cycle of the unscented Kalman filter for the non-additive non-zero-mean noise case, illustrating the computation of sigma points and its 'predict and correct' behaviour.

As opposed to most deterministic numerical methods, the estimation accuracy is independent of the dimensionality of the state space. According to the law of large numbers, for $N \to \infty$ the estimated expectation almost surely converges to the true expectation,

$$\hat{\mathbb{E}}_{p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})}\left[f(\mathbf{x}_k)\right] \xrightarrow{\text{a.s.}} \mathbb{E}_{p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})}\left[f(\mathbf{x}_k)\right], \quad \text{for } N \to \infty\,. \tag{69}$$

2.4.1 *Sequential Importance Sampling*

Since it is rarely possible to sample directly from the posterior distribution, *importance sampling* is used, which refers to sampling from an alternative distribution [53]. This easy-to-sample importance sampling distribution is also called *proposal distribution*. Expanding Equation 67 with an arbitrary proposal distribution $\pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})$ and applying Bayes' theorem yields

$$\begin{aligned}
\mathbb{E}_{p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})}\left[f(\mathbf{x}_k)\right] &= \int f(\mathbf{x}_k) \frac{p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})}{\pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})} \\
&\qquad \cdot \pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})\,d\mathbf{x}_k \\
&= \int f(\mathbf{x}_k) \frac{p(\mathbf{Z}_k \mid \mathbf{x}_k, \mathbf{u}_{k-1}) p(\mathbf{x}_k \mid \mathbf{u}_{k-1})}{p(\mathbf{Z}_k \mid \mathbf{u}_{k-1}) \pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})} \\
&\qquad \cdot \pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})\,d\mathbf{x}_k \\
&= \int f(\mathbf{x}_k) w_k(\mathbf{x}_k) p(\mathbf{Z}_k \mid \mathbf{u}_{k-1})^{-1} \\
&\qquad \cdot \pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})\,d\mathbf{x}_k\,,
\end{aligned} \tag{70}$$

with the *importance weight*

$$w_k(\mathbf{x}_k) = \frac{p(\mathbf{Z}_k \mid \mathbf{x}_k, \mathbf{u}_{k-1}) p(\mathbf{x}_k \mid \mathbf{u}_{k-1})}{\pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})}\,. \tag{71}$$

The unknown normalising distribution $p(\mathbf{Z}_k \mid \mathbf{u}_{k-1})$ can be written as

$$p(\mathbf{Z}_k \mid \mathbf{u}_{k-1}) = \int p(\mathbf{Z}_k \mid \mathbf{x}_k, \mathbf{u}_{k-1}) p(\mathbf{x}_k \mid \mathbf{u}_{k-1})\,d\mathbf{x}_k\,. \tag{72}$$

Multiplying the integrand with $\frac{\pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})}{\pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})}$ and substituting with the left side of Equation 71, we have

$$p(\mathbf{Z}_k \mid \mathbf{u}_{k-1}) = \int w_k(\mathbf{x}_k) \pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{u}_{k-1})\,d\mathbf{x}_k\,. \tag{73}$$

Plugging the result back into Equation 70 yields

$$\begin{aligned}
\mathbb{E}_{p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})}\left[f(\mathbf{x}_k)\right] &= \frac{\int f(\mathbf{x}_k) w_k(\mathbf{x}_k) \pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1}) \, d\mathbf{x}_k}{\int w_k(\mathbf{x}_k) \pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1}) \, d\mathbf{x}_k} \\
&= \frac{\mathbb{E}_{\pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})}\left[w_k(\mathbf{x}_k) f(\mathbf{x}_k)\right]}{\mathbb{E}_{\pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})}\left[w_k(\mathbf{x}_k)\right]} \\
&\approx \frac{\frac{1}{N} \sum_{i=1}^{N} w_k\left(\hat{\mathbf{x}}_k^{(i)}\right) f\left(\hat{\mathbf{x}}_k^{(i)}\right)}{\frac{1}{N} \sum_{i=1}^{N} w_k\left(\hat{\mathbf{x}}_k^{(i)}\right)} \\
&= \frac{1}{N} \sum_{i=1}^{N} \tilde{w}_k\left(\hat{\mathbf{x}}_k^{(i)}\right) f\left(\hat{\mathbf{x}}_k^{(i)}\right),
\end{aligned} \tag{74}$$

where $\hat{\mathbf{x}}_k^{(i)}$ denotes the i-th of N samples drawn from $\pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})$ and the normalised importance weights $\tilde{w}_k\left(\hat{\mathbf{x}}_k^{(i)}\right)$ are given by

$$\tilde{w}_k\left(\hat{\mathbf{x}}_k^{(i)}\right) = \frac{w_k\left(\hat{\mathbf{x}}_k^{(i)}\right)}{\sum_{i=1}^{N} w_k\left(\hat{\mathbf{x}}_k^{(i)}\right)}. \tag{75}$$

It follows that any expectation of the form given by Equation 67 can be estimated using weighted samples $\hat{\mathbf{x}}_k^{(i)}$ drawn from the proposal distribution $\pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})$. As it involves a ratio of two other estimates, this estimate is biased. However, given that the support of the proposal distribution includes the support of the posterior distribution $p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1})$, that is the following condition is satisfied: $\pi(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1}) \neq 0$ for any $\mathbf{x}_k$ for which $p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1}) \neq 0$, it is shown that this estimate is asymptotically unbiased [52]. Given samples drawn from the proposal distribution, the posterior distribution is approximated by the weighted point-mass estimate

$$p(\mathbf{x}_k \mid \mathbf{Z}_k, \mathbf{U}_{k-1}) \approx \sum_{i=1}^{N} \tilde{w}_k\left(\hat{\mathbf{x}}_k^{(i)}\right) \delta\left(\mathbf{x}_k - \hat{\mathbf{x}}_k^{(i)}\right). \tag{76}$$

Under the premise of the state space assumptions mentioned in Section 2.2, that is Markovianess and observational independence given the state, as shown in [54], a recursive estimate for the importance weights may be obtained as

$$\tilde{w}_k\left(\hat{\mathbf{x}}_k^{(i)}\right) = \tilde{w}_{k-1}\left(\hat{\mathbf{x}}_{k-1}^{(i)}\right) \frac{p\left(\mathbf{z}_k \mid \hat{\mathbf{x}}_k^{(i)}\right) p\left(\hat{\mathbf{x}}_k^{(i)} \mid \hat{\mathbf{x}}_{k-1}^{(i)}, \mathbf{u}_{k-1}\right)}{\pi\left(\hat{\mathbf{x}}_k^{(i)} \mid \hat{\mathbf{X}}_{k-1}^{(i)}, \mathbf{U}_{k-1}, \mathbf{Z}_k\right)}. \tag{77}$$

This variant of importance sampling is called *sequential importance sampling* (SIS), referring to its recursive nature. As it is clear that the weights $w_k\left(\hat{\mathbf{x}}_k^{(i)}\right)$ and normalised weights $\tilde{w}_k\left(\hat{\mathbf{x}}_k^{(i)}\right)$ are a function of the particles $\hat{\mathbf{x}}_k^{(i)}$, respectively, we will use a more succinct notation and from now on write $w_k^{(i)}$ and $\tilde{w}_k^{(i)}$ instead.

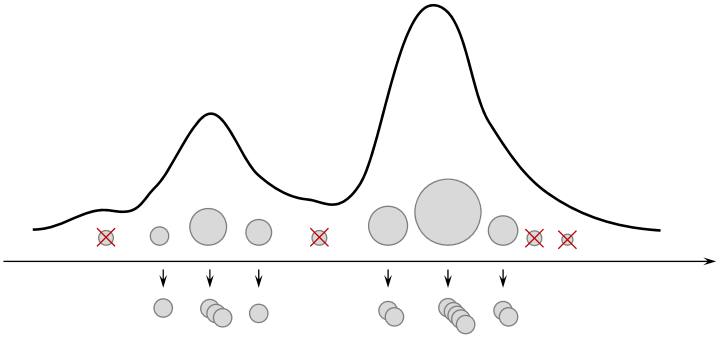


Figure 12: Resampling: replacing the weighted empirical distribution by an unweighted distribution [55]. Before resampling, the size of the samples represents their respective importance weight. Samples with low weights are likely to be eliminated, whereas samples with higher weights are likely to be reproduced.

2.4.2 *Sequential Importance Resampling*

A serious limitation of sequential importance sampling is that it degenerates with time. In fact, it is shown in [56] that SIS is guaranteed to fail for large k, as the distribution of the importance weights becomes more and more skewed [49]. After a few time steps, the majority of particles have a numerically insignificant importance weight, while the weight of one particle tends to one, which is effectively equal to removing the samples from the sample set. The consequence is that the posterior distribution is not adequately represented by the few remaining effective samples.

An additional selection step called *sequential importance resampling* (SIR), or simply *resampling*, partially mitigates this problem. Introduced in [57], the rationale behind the resampling idea is to eliminate the particles with low importance weight and instead multiply particles having high weight. Therefore, the resampling procedure reflects the Darwinian idea of *survival of the fittest* and is illustrated for one dimension in Figure 12.

Formally, resampling is carried out by replacing the weighted empirical distribution in Equation 76 by an unweighted distribution

$$p(\mathbf{x}_k \,|\, \mathbf{Z}_k, \mathbf{U}_{k-1}) = \frac{1}{N} \sum_{i=1}^{N} N_k^{(i)} \delta\left(\mathbf{x}_k - \mathbf{x}_k^{(i)}\right), \tag{78}$$

where $N_k^{(i)}$ is the number of offsprings from particle $\hat{\mathbf{x}}_k^{(i)}$, such that the number of particles remains constant,

$$\sum_{i=1}^{N} N_k^{(i)} = N\,. \tag{79}$$

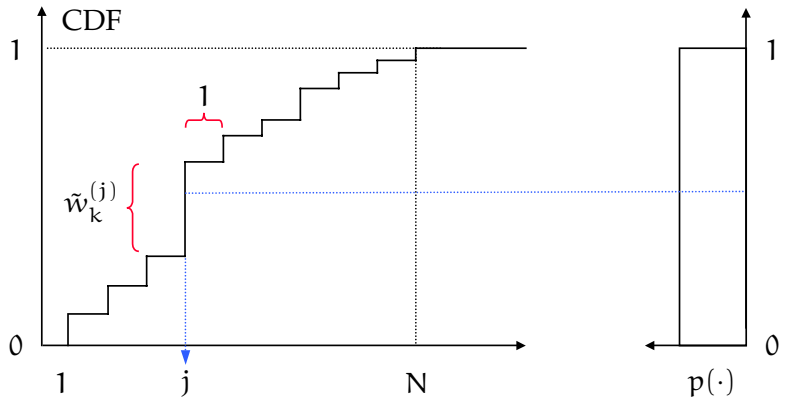


Figure 13: Multinomial resampling [49]. Depicted is the projection of a random sample from the continuous uniform distribution on the interval $[0, 1]$ onto the range of the cumulative distribution function (CDF), which is plotted over the sample numbers $1, \ldots, N$. The intersection with the CDF is projected on the abscissa to obtain the sample number j, which is then used to select a sample from $\{\hat{x}_k^{(j)}\}_{j=1}^N$.

Multinomial Resampling

Employing the most popular resampling scheme termed *multinomial resampling*, which was introduced in [57], the surviving particles are obtained by drawing N samples $x_k^{(i)}$ from the set $\{\hat{x}_k^{(j)}\}_{j=1}^N$, whilst samples are chosen with a probability equal to their normalised importance weight. This yields the resampled set

$$\{x_k^{(i)}\}_{i=1}^N, \text{ such that } \Pr\left(x_k^{(i)} = \hat{x}_k^{(j)}\right) = \tilde{w}_k^{(j)}, \quad i, j \in \{1, \ldots, N\}, \tag{80}$$

with equal weights,

$$w_k^{(i)} = \frac{1}{N}, \quad i = 1, \ldots, N. \tag{81}$$

The mapping from a weighted random measure to an unweighted random measure using a uniform distribution is illustrated in Figure 13. First, the cumulative distribution function (CDF) of the set $\{\hat{x}_k^{(j)}\}_{j=1}^N$ is constructed. Then, a number from the continuous uniform distribution on the interval $[0, 1]$ is sampled and projected on the range of the CDF. The intersection is projected on the sample number, which is used to select the j-th sample from $\{\hat{x}_k^{(j)}\}_{j=1}^N$.

2.4.3 *Importance of the Proposal Distribution*

For a finite set of samples, the importance sampling method performs poorly if only a few particles are placed in regions where the desired posterior is large [58]. This makes the choice of the proposal distribution a critical design issue for SIS algorithms [49, 59].

An optimal proposal distribution may be defined as one that minimises the variance of the importance weights [60]. Then, as shown analogously to the proof in [61], which does not consider a control input, the target distribution is given by

$$\begin{aligned}\pi_{\text{opt}}\left(\mathbf{x}_k \,|\, \mathbf{X}_{k-1}^{(i)}, \mathbf{U}_{k-1}, \mathbf{Z}_k\right) &= p\left(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}^{(i)}, \mathbf{u}_{k-1}, \mathbf{z}_k\right) \\ &= \frac{p\left(\mathbf{z}_k \,|\, \mathbf{x}_k\right) p\left(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}^{(i)}, \mathbf{u}_{k-1}\right)}{p\left(\mathbf{z}_k \,|\, \mathbf{x}_{k-1}^{(i)}, \mathbf{u}_{k-1}\right)}\,. \end{aligned} \tag{82}$$

Plugging the result into Equation 77 yields

$$\begin{aligned}\tilde{w}_k &= \tilde{w}_{k-1}\, p\left(\mathbf{z}_k \,|\, \mathbf{x}_{k-1}^{(i)}, \mathbf{u}_{k-1}\right) \\ &= \tilde{w}_{k-1} \int p\left(\mathbf{z}_k \,|\, \mathbf{x}_k\right) p\left(\mathbf{x}_k \,|\, \hat{\mathbf{x}}_{k-1}^{(i)}, \mathbf{u}_{k-1}\right) d\mathbf{x}_k\,. \end{aligned} \tag{83}$$

However, this density suffers from the following two major drawbacks: it requires the ability to sample from $p\left(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}^{(i)}, \mathbf{u}_{k-1}, \mathbf{z}_k\right)$ and to evaluate the integral in Equation 83, both of which may not be straightforward in practice [51]. Therefore, it is desirable to find an approximation of the optimal proposal distribution. In [62], Salmond and Gordon state the "considerable scope for ingenuity in designing the importance density". A popular method for devising a proposal distribution that approximates the optimal distribution given by Equation 82 is taking into account the latest measurement [52, 63], which we will elaborate on in detail in Section 2.5.3.

2.5 PARTICLE FILTERS

In this section, we shall introduce a non-parametric filtering method known as the *particle filter*. Relying in principle on the Monte Carlo method, particle filters can handle arbitrary multi-modal distributions. However, their computational cost is a monotonically increasing function of the estimation accuracy.

2.5.1 *The Generic Particle Filter*

The generic particle filter is a Monte Carlo algorithm that matches the Bayesian framework. Starting the filter algorithm at time $k = 0$, first, N samples $\mathbf{x}_0^{(i)}$ from the initial state distribution $p(\mathbf{x}_0)$ are drawn,

$$\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0), \quad i \in \{1, \ldots, N\}, \tag{84}$$

and weighted equally as

$$w_0^{(i)} = \frac{1}{N}, \quad i \in \{1, \ldots, N\}. \tag{85}$$

Then, the following steps are carried out recursively at every time step $k > 0$. We draw N equally weighted samples $\hat{\mathbf{x}}_k^{(i)}$ from the proposal distribution $\pi(\mathbf{x}_k \,|\, \mathbf{X}_{k-1}^{(i)}, \mathbf{Z}_k, \mathbf{U}_{k-1})$. The set of particles and their respective weights is denoted as

$$\left\{ \left(\hat{\mathbf{x}}_k^{(i)}, N^{-1} \right) \right\}_{i=1}^{N}, \quad \hat{\mathbf{x}}_k^{(i)} \sim \pi\left(\mathbf{x}_k \,|\, \mathbf{X}_{k-1}^{(i)}, \mathbf{Z}_k, \mathbf{U}_{k-1} \right). \tag{86}$$

In the light of the measurement $\mathbf{z}_k$, the importance weights $w_k^{(i)}$ are computed for each particle according to Equation 77,

$$w_k^{(i)} = w_{k-1}^{(i)} \frac{p\left(\mathbf{z}_k \,|\, \hat{\mathbf{x}}_k^{(i)} \right) p\left(\hat{\mathbf{x}}_k^{(i)} \,|\, \mathbf{x}_{k-1}^{(i)}, \mathbf{u}_{k-1} \right)}{\pi\left(\hat{\mathbf{x}}_k^{(i)} \,|\, \mathbf{X}_{k-1}^{(i)}, \mathbf{Z}_k, \mathbf{U}_{k-1} \right)}, \quad i \in \{1, \ldots, N\}. \tag{87}$$

Normalising yields

$$\tilde{w}_k^{(i)} = w_k^{(i)} \left[\sum_{j=1}^{N} w_k^{(j)} \right]^{-1}, \quad i \in \{1, \ldots, N\}. \tag{88}$$

Now, the set of particles is resampled according to Equation 80 and the weights are set equally according to Equation 81.

The output of the filter is given by a set of N equally weighted particles $\mathbf{x}_k^{(i)}$ that can be used to approximate the posterior distribution, according to Equation 66, and the expectations under it, according to Equation 68. Often, the estimate of the conditional mean

$$\hat{\mathbf{x}}_k = \tilde{\mathbb{E}}_{p(\mathbf{x}_k \,|\, \mathbf{Z}_k, \mathbf{u}_{k-1})} \left[\mathbf{x}_k \right] = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_k^{(i)}, \tag{89}$$

which represents the minimum mean-squared error estimate of the current state, and the covariance

$$\begin{aligned} \mathbf{P}_k &= \tilde{\mathbb{E}}_{p(\mathbf{x}_k \,|\, \mathbf{Z}_k, \mathbf{u}_{k-1})} \left[\left(\mathbf{x}_k - \hat{\mathbf{x}}_k \right) \left(\mathbf{x}_k - \hat{\mathbf{x}}_k \right)^T \right] \\ &= \sum_{i=1}^{N} \left(\mathbf{x}_k^{(i)} - \hat{\mathbf{x}}_k \right) \left(\mathbf{x}_k^{(i)} - \hat{\mathbf{x}}_k \right)^T, \end{aligned} \tag{90}$$

are the quantities of particular interest. The algorithm of the generic particle filter is depicted in Figure 14, illustrating the importance sampling and resampling step.

2.5.2 *The Bootstrap Filter*

Although many authors have recognised the importance of the proposal distribution for a successful application of SIS [64–66], the transition prior probability distribution $p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \mathbf{u}_{k-1})$ is often used as importance function [57, 67–70]. Filters using the transition prior as

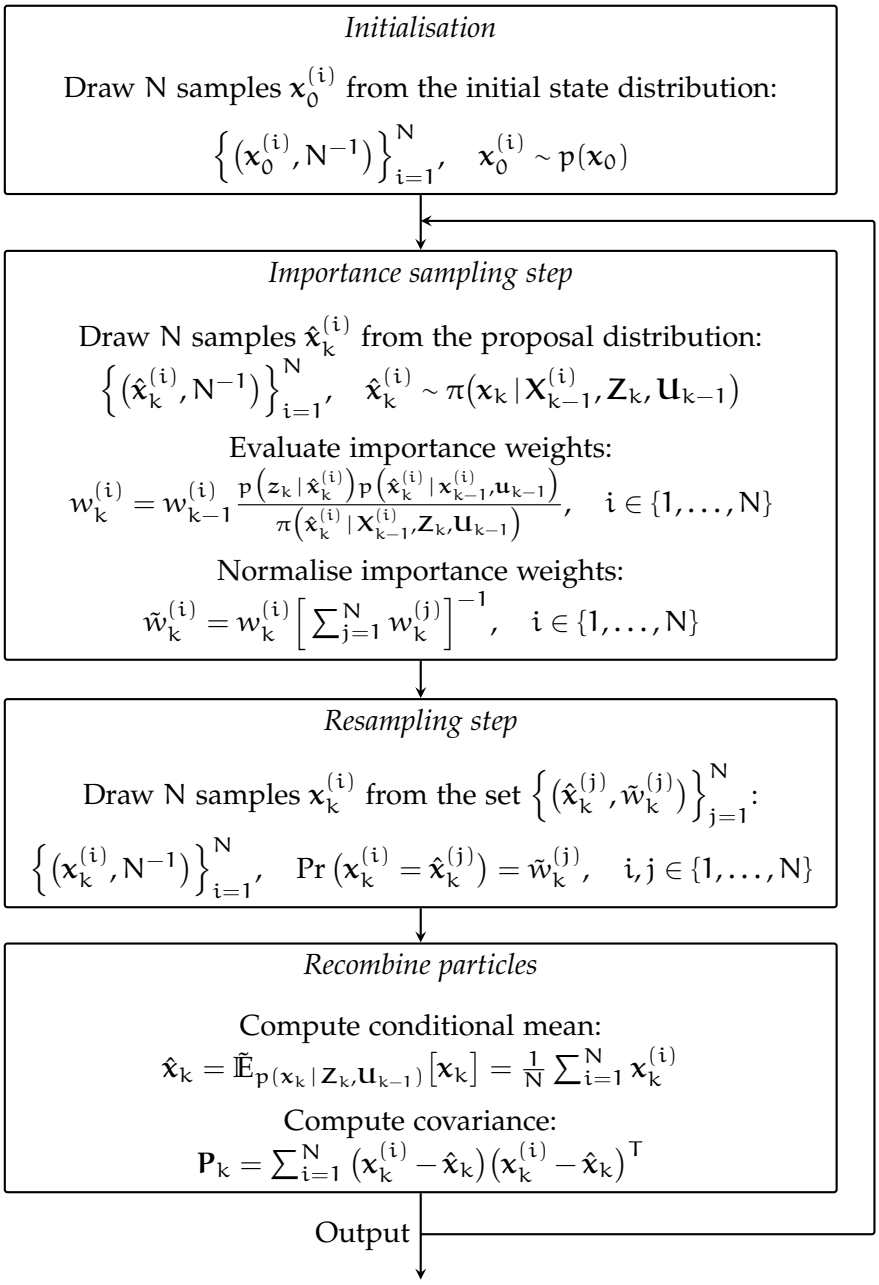


Figure 14: Operation cycle of the generic particle filter, illustrating the importance sampling and resampling step. Finally, the particles are recombined to obtain the minimum mean-squared error estimate and the corresponding covariance.

proposal distribution are commonly known as *bootstrap filter*. The transition prior is easy to sample from and, as a result of it not incorporating the most recent observation, it is usually easier to implement [52, 65, 71]. In the case of an additive Gaussian process noise model, the transition prior is simply

$$p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \mathbf{u}_{k-1}) = \mathcal{N}\big(\boldsymbol{\phi}_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, 0), \mathbf{Q}_{k-1}\big)\,. \tag{91}$$

In the importance sampling step of the bootstrap filter, each of the particles is propagated through the system model according to Equation 1,

$$\hat{\mathbf{x}}_k^{(i)} = \boldsymbol{\phi}_{k-1}\big(\mathbf{x}_{k-1}^{(i)}, \mathbf{u}_{k-1}, \boldsymbol{w}_{k-1}^{(i)}\big), \quad k > 0, \quad i \in \{1, \ldots, N\}. \tag{92}$$

where $\boldsymbol{w}_{k-1}^{(i)}$ represents a sample drawn from the system noise distribution. Adding this noise sample creates variety in the set of hypotheses $\left\{\big(\hat{\mathbf{x}}_k^{(i)}, N^{-1}\big)\right\}_{i=1}^{N}$. Note that this step is equivalent to sampling from the prior distribution, as

$$\hat{\mathbf{x}}_k^{(i)} \sim p\big(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}^{(i)}, \mathbf{u}_{k-1}\big), \quad k > 0, \quad i \in \{1, \ldots, N\}. \tag{93}$$

Now, in the light of the measurement z_k, the importance weights $w_k^{(i)}$ are computed for each particle. When using the transition prior as proposal distribution, with

$$\pi\big(\mathbf{x}_k \,|\, \mathbf{X}_{k-1}^{(i)}, \mathbf{Z}_k, \mathbf{U}_{k-1}\big) = p\big(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}^{(i)}, \mathbf{u}_{k-1}\big)\,, \tag{94}$$

Equation 87 simplifies to

$$w_k^{(i)} = w_{k-1}^{(i)} p\big(z_k \,|\, \hat{\mathbf{x}}_k^{(i)}\big), \quad k > 0, \quad i \in \{1, \ldots, N\}. \tag{95}$$

If resampling is carried out every time step, the propagated samples have uniform weights and Equation 91 reduces to

$$w_k^{(i)} = p\big(z_k \,|\, \hat{\mathbf{x}}_k^{(i)}\big), \quad k > 0, \quad i \in \{1, \ldots, N\}. \tag{96}$$

The algorithm of the bootstrap filter is depicted in Figure 15, illustrating the importance sampling and resampling step. A graphical representation of the evolution of the empirical probability distributions estimated by the bootstrap filter is depicted in Figure 16.

2.5.3 *The Unscented Particle Filter*

Daum pointed out in [59] that in most of the successful implementations of particle filters the proposal distribution is obtained using an extended Kalman filter or an unscented Kalman filter. Both filters

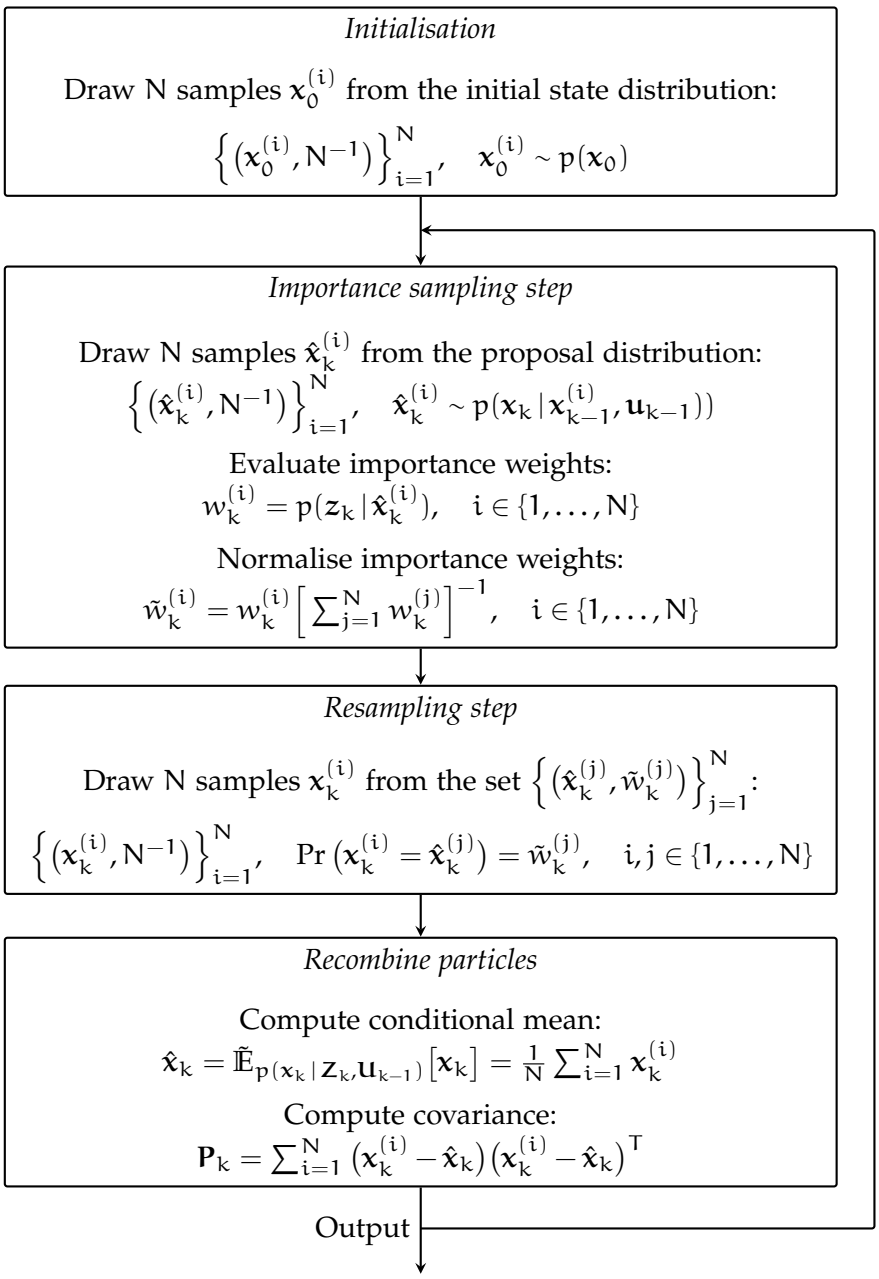


Figure 15: Operation cycle of the bootstrap filter, illustrating the importance sampling and resampling step. Finally, the particles are recombined to obtain the minimum mean-squared error estimate and the corresponding covariance.

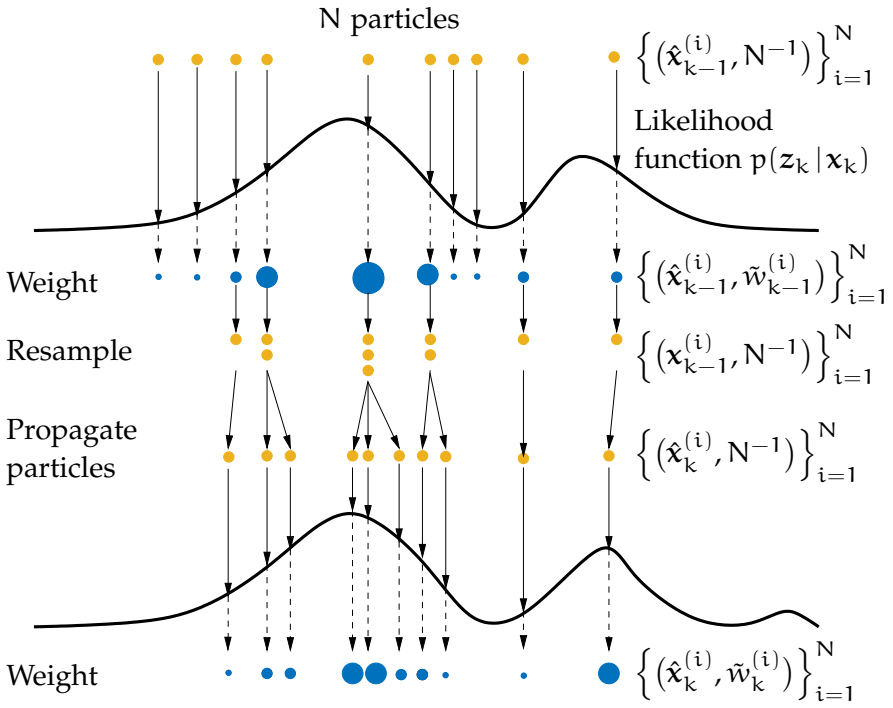


Figure 16: Evolution of the empirical probability distributions estimated by the bootstrap filter. The filter starts at time $k-1$ with the unweighted particles $\{(\hat{x}_{k-1}^{(i)}, N^{-1})\}_{i=1}^{N}$, which provide an approximation of $p(x_{k-1} \mid x_{k-2}^{(i)}, u_{k-2})$. Computing the importance weights yields an approximation of $p(x_{k-1} \mid Y_{k-1}, U_{k-1})$ given by the set $\{(\hat{x}_{k-1}^{(i)}, \tilde{w}_{k-1}^{(i)})\}_{i=1}^{N}$. In the resampling step, the fittest particles are reproduced to obtain $\{(x_{k-1}^{(i)}, N^{-1})\}_{i=1}^{N}$, which again approximates $p(x_{k-1} \mid Y_{k-1}, U_{k-1})$. Finally, closing the filter loop, the prediction step produces variety, resulting in $\{(\hat{x}_{k}^{(i)}, N^{-1})\}_{i=1}^{N}$, an approximation of $p(x_k \mid x_{k-1}^{(i)}, u_{k-1})$ [49].

compute recursive Gaussian approximations of the posterior filtering distribution,

$$p(x_k \mid Z_k, U_{k-1}) \approx p_{\mathcal{N}}(x_k \mid Z_k, U_{k-1}) = \mathcal{N}(\hat{x}_k, P_k)\,. \tag{97}$$

incorporating the latest measurement at each time step. Within the particle filter framework, the EKF or UKF can be used to propagate a Gaussian proposal distribution for each particle,

$$\pi(x_k \mid X_{k-1}^{(i)}, Z_k, U_{k-1}) = \mathcal{N}(\bar{x}_k^{(i)}, P_k^{(i)}), \quad i \in \{1, \ldots, N\}. \tag{98}$$

The i-th particle at time step k is then sampled from this distribution,

$$\hat{x}_k^{(i)} \sim \pi(x_k \mid X_{k-1}^{(i)}, Z_k, U_{k-1}), \quad k > 0, \quad i \in \{1, \ldots, N\}. \tag{99}$$

Although the individual proposal distributions for each particle are Gaussian, in general, the form of the overall proposal distribution

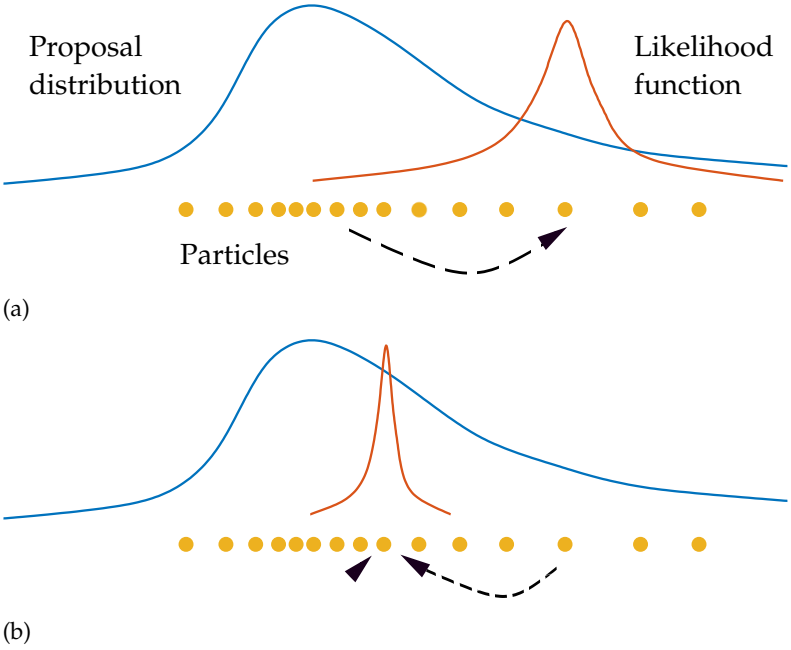
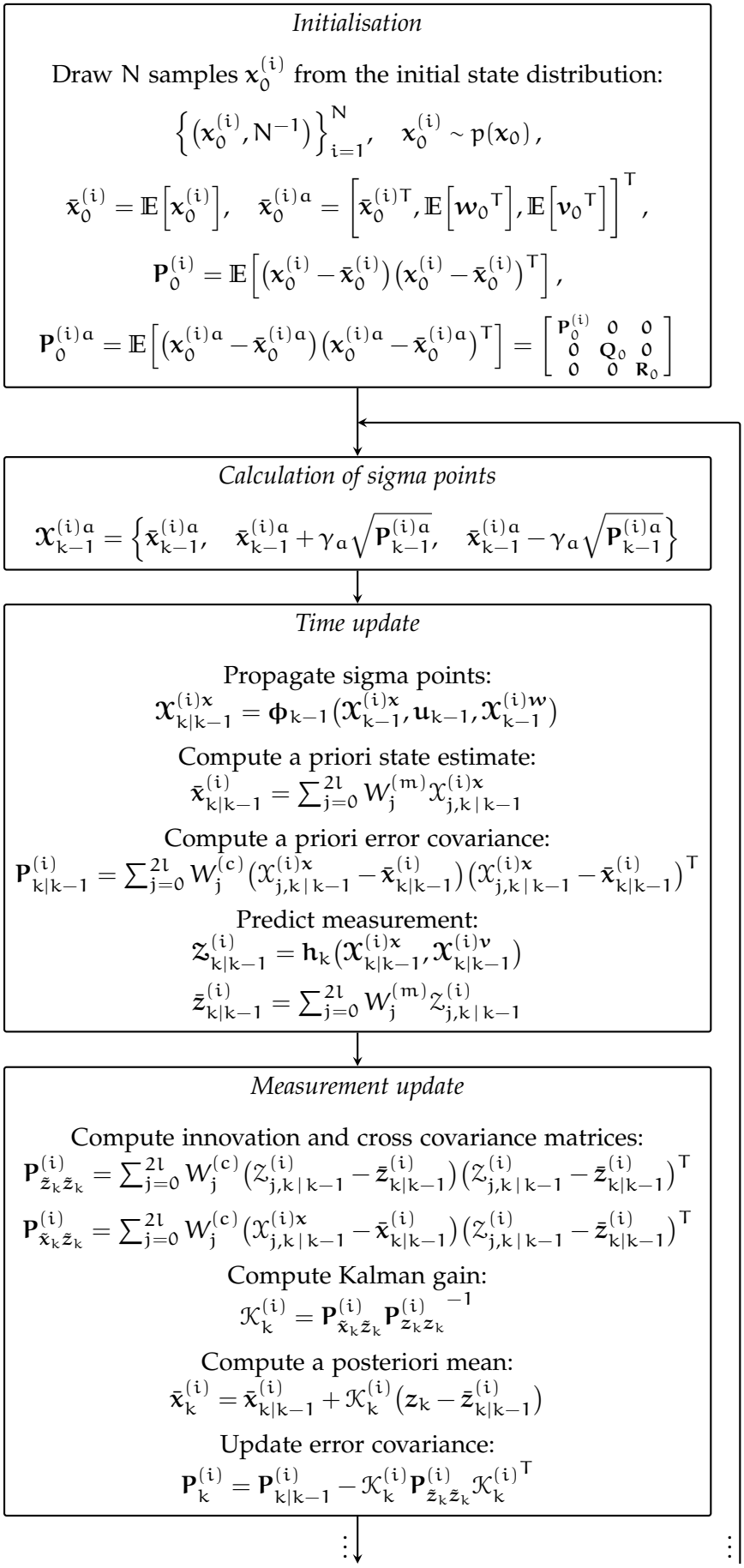


Figure 17: The two scenarios in which a parametric filter such as the EKF or UKF may help generating better proposal distributions by moving particles to regions of high likelihood: (a) when there is little overlap of proposal distribution and likelihood function, that is when the peak of the likelihood happens to lie in one of the tails of the proposal distribution or (b) when the likelihood function is very narrow, which is the case for very accurate measurements.

$\pi(\mathbf{x}_k \,|\, \mathbf{X}_{k-1}, \mathbf{Z}_k, \mathbf{u}_{k-1})$ will not be Gaussian. Though, the peak of the proposal distribution is moved towards the peak of the likelihood function, therefore moving particles to regions of high likelihood, as illustrated in Figure 17. Scenario (a) depicts little overlap of the proposal distribution and the likelihood function, which may be the case when the available model is not sufficiently accurate. Scenario (b) depicts peaked likelihood, which represents very accurate measurements.

Van Der Merwe et al. [50] proposed to use an unscented Kalman filter in order to incorporate the latest measurement and thus to generate better proposal distributions. Based on the ability of the UKF to more accurately propagate the mean and covariance of the state distribution, when compared to the EKF, the result is a bigger support overlap with the true posterior distribution. The resulting filter is referred to as the *unscented particle filter*.

Figure 18 depicts the entire unscented particle filter algorithm, including the calculation of sigma points for each particle, the time and measurement update step, as well as the importance sampling and resampling step. Note that we denote the mean of the Gaussian proposal distribution of the i-th particle with $\bar{\mathbf{x}}_k^{(i)}$, being consistent with former notation of the mean. For the i-th hypothesis $\hat{\mathbf{x}}^{(i)}$ of the state we continue to use a hat.


Initialisation
Draw N samples $\mathbf{x}_0^{(i)}$ from the initial state distribution:
$\left\{\left(\mathbf{x}_0^{(i)}, N^{-1}\right)\right\}_{i=1}^{N}, \quad \mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0)$,
$\bar{\mathbf{x}}_0^{(i)} = \mathbb{E}\left[\mathbf{x}_0^{(i)}\right], \quad \bar{\mathbf{x}}_0^{(i)a} = \left[\bar{\mathbf{x}}_0^{(i)T}, \mathbb{E}\left[\mathbf{w}_0{}^T\right], \mathbb{E}\left[\mathbf{v}_0{}^T\right]\right]^T$,
$\mathbf{P}_0^{(i)} = \mathbb{E}\left[\left(\mathbf{x}_0^{(i)} - \bar{\mathbf{x}}_0^{(i)}\right)\left(\mathbf{x}_0^{(i)} - \bar{\mathbf{x}}_0^{(i)}\right)^T\right]$,
$\mathbf{P}_0^{(i)a} = \mathbb{E}\left[\left(\mathbf{x}_0^{(i)a} - \bar{\mathbf{x}}_0^{(i)a}\right)\left(\mathbf{x}_0^{(i)a} - \bar{\mathbf{x}}_0^{(i)a}\right)^T\right] = \begin{bmatrix} \mathbf{P}_0^{(i)} & 0 & 0 \\ 0 & \mathbf{Q}_0 & 0 \\ 0 & 0 & \mathbf{R}_0 \end{bmatrix}$
Calculation of sigma points
$\mathcal{X}_{k-1}^{(i)a} = \left\{\bar{\mathbf{x}}_{k-1}^{(i)a}, \quad \bar{\mathbf{x}}_{k-1}^{(i)a} + \gamma_a\sqrt{\mathbf{P}_{k-1}^{(i)a}}, \quad \bar{\mathbf{x}}_{k-1}^{(i)a} - \gamma_a\sqrt{\mathbf{P}_{k-1}^{(i)a}}\right\}$
Time update
Propagate sigma points:
$\mathcal{X}_{k|k-1}^{(i)x} = \boldsymbol{\phi}_{k-1}(\mathcal{X}_{k-1}^{(i)x}, \mathbf{u}_{k-1}, \mathcal{X}_{k-1}^{(i)w})$
Compute a priori state estimate:
$\bar{\mathbf{x}}_{k|k-1}^{(i)} = \sum_{j=0}^{2l} W_j^{(m)} \mathcal{X}_{j,k\,|\,k-1}^{(i)x}$
Compute a priori error covariance:
$\mathbf{P}_{k|k-1}^{(i)} = \sum_{j=0}^{2l} W_j^{(c)}\left(\mathcal{X}_{j,k\,|\,k-1}^{(i)x} - \bar{\mathbf{x}}_{k|k-1}^{(i)}\right)\left(\mathcal{X}_{j,k\,|\,k-1}^{(i)x} - \bar{\mathbf{x}}_{k|k-1}^{(i)}\right)^T$
Predict measurement:
$\mathcal{Z}_{k|k-1}^{(i)} = \mathbf{h}_k(\mathcal{X}_{k|k-1}^{(i)x}, \mathcal{X}_{k|k-1}^{(i)v})$
$\bar{\mathbf{z}}_{k|k-1}^{(i)} = \sum_{j=0}^{2l} W_j^{(m)} \mathcal{Z}_{j,k\,|\,k-1}^{(i)}$
Measurement update
Compute innovation and cross covariance matrices:
$\mathbf{P}_{\tilde{\mathbf{z}}_k\tilde{\mathbf{z}}_k}^{(i)} = \sum_{j=0}^{2l} W_j^{(c)}\left(\mathcal{Z}_{j,k\,|\,k-1}^{(i)} - \bar{\mathbf{z}}_{k|k-1}^{(i)}\right)\left(\mathcal{Z}_{j,k\,|\,k-1}^{(i)} - \bar{\mathbf{z}}_{k|k-1}^{(i)}\right)^T$
$\mathbf{P}_{\tilde{\mathbf{x}}_k\tilde{\mathbf{z}}_k}^{(i)} = \sum_{j=0}^{2l} W_j^{(c)}\left(\mathcal{X}_{j,k\,|\,k-1}^{(i)x} - \bar{\mathbf{x}}_{k|k-1}^{(i)}\right)\left(\mathcal{Z}_{j,k\,|\,k-1}^{(i)} - \bar{\mathbf{z}}_{k|k-1}^{(i)}\right)^T$
Compute Kalman gain:
$\mathcal{K}_k^{(i)} = \mathbf{P}_{\tilde{\mathbf{x}}_k\tilde{\mathbf{z}}_k}^{(i)}\mathbf{P}_{z_kz_k}^{(i)}{}^{-1}$
Compute a posteriori mean:
$\tilde{\mathbf{x}}_k^{(i)} = \bar{\mathbf{x}}_{k|k-1}^{(i)} + \mathcal{K}_k^{(i)}\left(\mathbf{z}_k - \bar{\mathbf{z}}_{k|k-1}^{(i)}\right)$
Update error covariance:
$\mathbf{P}_k^{(i)} = \mathbf{P}_{k|k-1}^{(i)} - \mathcal{K}_k^{(i)}\mathbf{P}_{\tilde{\mathbf{z}}_k\tilde{\mathbf{z}}_k}^{(i)}\mathcal{K}_k^{(i)T}$

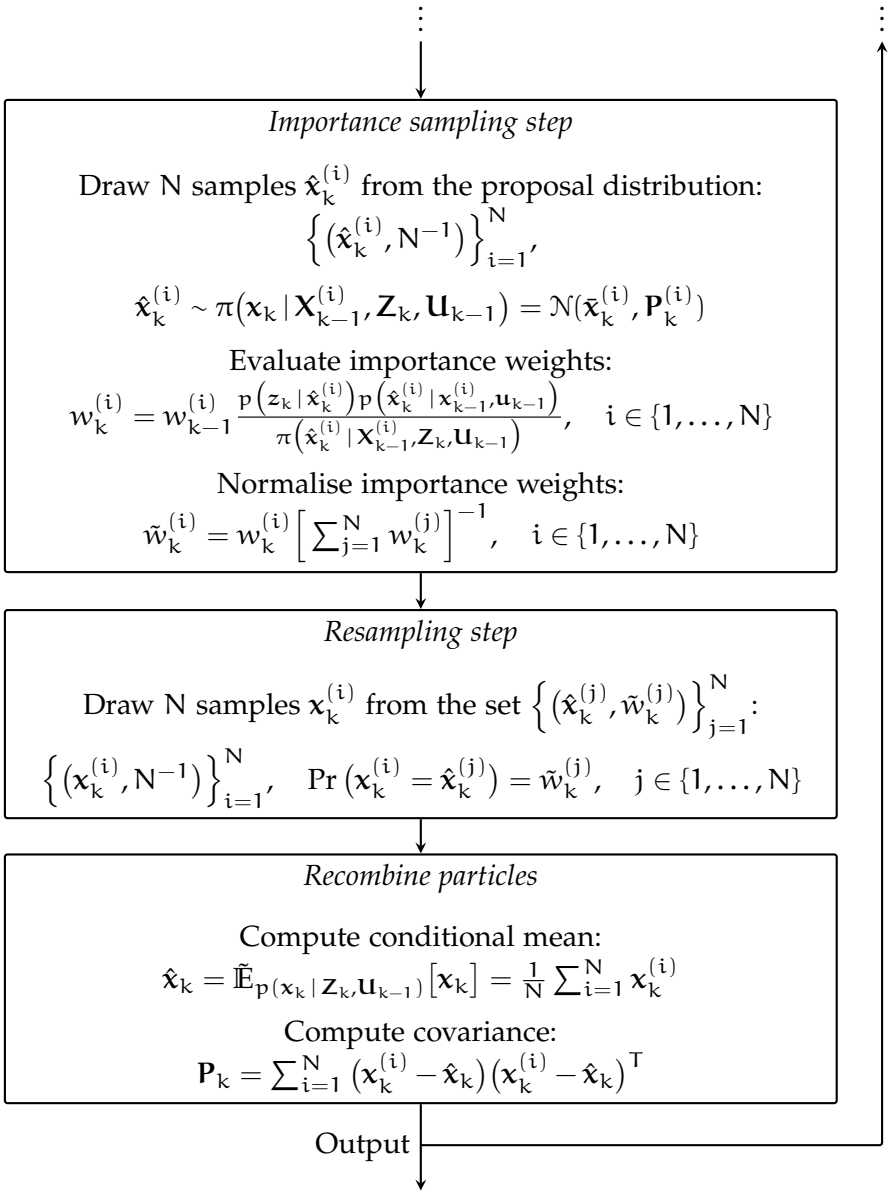


Figure 18: Operation cycle of the unscented particle filter, illustrating the calculation of sigma points for each particle, the time and measurement update step, as well as the importance sampling and resampling step. Finally, the particles are recombined to obtain the minimum mean-squared error estimate and the corresponding covariance.

2.6 INTERVAL ANALYSIS

Built on the foundation of set theory, *interval analysis* encompasses numerical methods that yield reliable results. Originating back to Moore's doctorate on the use of intervals to analyse and control numerical errors in digital computers in 1962, interval computation has come a long way and has been subject to extensive research since. As opposed to the probabilistic methods presented above, which provide a point estimate associated with a given uncertainty in the form of a probability distribution, interval computations provide a box guaranteed to contain the solution, given that the problem is well modelled and the assumptions are sufficiently conservative. However, this guarantee comes with the trade-off that any point in the confined solution space is assumed to be equally likely.

As we shall see below, self-localisation can be modelled as an optimisation problem. In [72], Rokne states the following advantages of interval methods to global optimisation problems. They

- do not rely on starting points,
- can prove existence, absence, and uniqueness of solutions,
- can easily accommodate external constraints,
- and are reliable in that they never discard a feasible solution.

An in-depth presentation of the following basic concepts can be found in [73].

2.6.1 *Basic Concepts*

Let $[x]$ denote a *closed interval*, defined as a closed subset of the real numbers given by

$$[x] = \left[\underline{x}, \overline{x}\right] = \left\{x \in \mathbb{R} \mid \underline{x} \leqslant x \leqslant \overline{x}\right\}, \tag{100}$$

where $\underline{x}$ and $\overline{x}$ are called the *lower bound* and the *upper bound* of the interval, respectively. An interval $[x]$ can be assigned a *width* $w\left([x]\right)$, a *midpoint* $\operatorname{mid}\left([x]\right)$, and an *absolute value* $\left|[x]\right|$, given by

$$w\left([x]\right) = \overline{x} - \underline{x}, \tag{101}$$

$$\operatorname{mid}\left([x]\right) = \frac{\underline{x} + \overline{x}}{2}, \tag{102}$$

$$\left|[x]\right| = \max\left(|\underline{x}|, |\overline{x}|\right), \tag{103}$$

respectively. We denote the set of closed intervals with endpoints in the set of real numbers augmented by plus and minus infinity with

$$[\mathbb{R}] = \left\{ \left[\underline{x}, \overline{x}\right] \mid \underline{x}, \overline{x} \in \mathbb{R} \cup \{-\infty, \infty\}, \quad \underline{x} \leqslant \overline{x} \right\} \cup \emptyset. \tag{104}$$

Note that we include the empty set, in order to form a closed interval system, that is, a system in which there are no undefined operator-operand or function-argument combinations [74].

The *intersection* of two intervals $[x]$ and $[y] \in [\mathbb{R}]$ is defined as the interval

$$\begin{aligned}[x] \cap [y] &= \left\{ z \in \mathbb{R} \mid z \in [x] \text{ and } z \in [y] \right\} \\ &= \begin{cases} \left[\max(\underline{x}, \underline{y}), \min(\overline{x}, \overline{y}) \right] & \text{if } \max(\underline{x}, \underline{y}) \leqslant \min(\overline{x}, \overline{y}) \\ \emptyset & \text{otherwise} \end{cases} .\end{aligned} \tag{105}$$

While the intersection is always an interval, this is not true for the *union* of $[x]$ and $[y]$, given by

$$[x] \cup [y] = \left\{ z \in \mathbb{R} \mid z \in [x] \text{ or } z \in [y] \right\} . \tag{106}$$

To make the set of intervals closed with respect to the union, we define the *interval hull* of a subset $\mathbb{X} \in \mathbb{R}$ as the smallest interval $[\mathbb{X}]$ that contains it. Now, we can define the *interval union* of $[x]$ and $[y]$ as the interval hull of the union $[x] \cup [y]$,

$$\begin{aligned}[x] \sqcup [y] &= \left[[x] \cup [y] \right] \\ &= \left[\min(\underline{x}, \underline{y}), \max(\overline{x}, \overline{y}) \right] .\end{aligned} \tag{107}$$

Multiplication of a real number α with an interval $[x]$ is defined as

$$\begin{aligned}\alpha[x] &= \left\{ \alpha x \mid x \in [x] \right\} \\ &= \begin{cases} \left[\alpha\underline{x}, \alpha\overline{x} \right] & \text{if } \alpha \geqslant 0 \\ \left[\alpha\overline{x}, \alpha\underline{x} \right] & \text{if } \alpha < 0 \end{cases} .\end{aligned} \tag{108}$$

An inner binary operation $\circ$ on the real intervals $[x], [y] \in [\mathbb{R}]$ is defined by

$$[x] \circ [y] = \left\{ x \circ y \in \mathbb{R} \mid x \in [x] \text{ and } y \in [y] \right\} . \tag{109}$$

Hence, the four basic arithmetic operations on intervals are given by

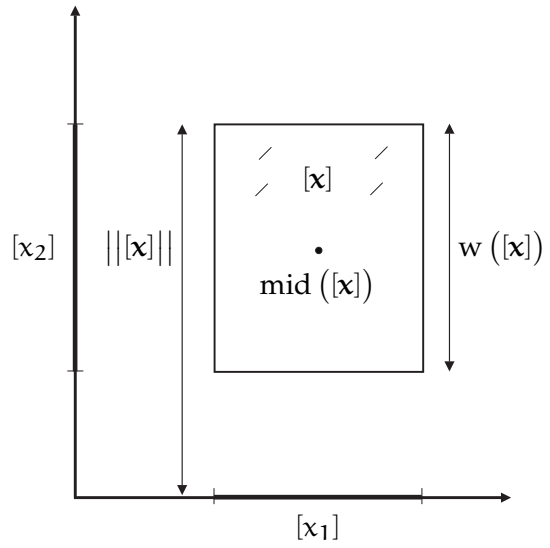


Figure 19: The projection of an interval vector $[\mathbf{x}] = \left[[x_1], [x_2]\right] \in [\mathbb{R}]^2$ onto the two axes as well as its width $w\left([\mathbf{x}]\right)$, norm $\left|\left|[\mathbf{x}]\right|\right|$, and midpoint $\text{mid}\left([\mathbf{x}]\right)$, [75].

$$[x] + [y] = \left[\underline{x} + \underline{y}, \overline{x} + \overline{y}\right] \tag{110}$$

$$[x] - [y] = \left[\underline{x} - \overline{y}, \overline{x} - \underline{y}\right] \tag{111}$$

$$[x] \cdot [y] = \left[\begin{array}{l} \min\left(\underline{x}\cdot\underline{y}, \underline{x}\cdot\overline{y}, \overline{x}\cdot\underline{y}, \overline{x}\cdot\overline{y}\right), \\ \quad \max\left(\underline{x}\cdot\underline{y}, \underline{x}\cdot\overline{y}, \overline{x}\cdot\underline{y}, \overline{x}\cdot\overline{y}\right)\end{array}\right] \tag{112}$$

$$\frac{[x]}{[y]} = \left[\begin{array}{l} \min\left(\frac{\underline{x}}{\underline{y}}, \frac{\underline{x}}{\overline{y}}, \frac{\overline{x}}{\underline{y}}, \frac{\overline{x}}{\overline{y}}\right), \\ \quad \max\left(\frac{\underline{x}}{\underline{y}}, \frac{\underline{x}}{\overline{y}}, \frac{\overline{x}}{\underline{y}}, \frac{\overline{x}}{\overline{y}}\right)\end{array}\right], \quad 0 \notin [y] \tag{113}$$

These definitions are motivated by the following argument: Given two intervals $[x]$ and $[y]$ and two exact values $x \in [x]$ and $y \in [y]$, it is guaranteed that $x \circ y \in \left([x] \circ [y]\right)$, even though the exact values of x and y may not be known [72].

A *real interval vector* $[\mathbf{x}] = \left([x_1], [x_2], \ldots, [x_n]\right)^T \in [\mathbb{R}]^n$, also called a *box*, is defined as the Cartesian product of n real intervals,

$$[\mathbf{x}] = [x_1] \times [x_2] \times \cdots \times [x_n], \tag{114}$$

where the i-th interval component $[x_i]$ represents the projection of $[\mathbf{x}]$ onto the i-th axis. We define multiplication of an interval vector with a real number and the addition and multiplication of two interval vectors as follows:

$$\alpha[\mathbf{x}] = \big(\alpha[x_1], \ldots, \alpha[x_n]\big)\,, \tag{115}$$

$$[\mathbf{x}] + [\mathbf{y}] = \big([x_1] + [y_1], \ldots, [x_n] + [y_n]\big)\,, \tag{116}$$

$$[\mathbf{x}]^T \cdot [\mathbf{y}] = [x_1] \cdot [y_1] + \cdots + [x_n] \cdot [y_n]\,. \tag{117}$$

Likewise, the width and the midpoint can be extended naturally to interval vectors as follows:

$$\mathrm{w}\big([\mathbf{x}]\big) = \max_{1 \leqslant i \leqslant n} \Big(\mathrm{w}\big([x_i]\big)\Big) \tag{118}$$

$$\mathrm{mid}\big([\mathbf{x}]\big) = \Big[\mathrm{mid}\big([x_1]\big), \ldots, \mathrm{mid}\big[x_n]\big)\Big]^T\,. \tag{119}$$

A box can be assigned a norm $\big|\big|[\mathbf{x}]\big|\big|$, which represents a generalisation of the absolute value, given by

$$\big|\big|[\mathbf{x}]\big|\big| = \max_{1 \leqslant i \leqslant n} \Big(\big|[x_i]\big|\Big)\,. \tag{120}$$

The projection of an interval vector $[\mathbf{x}]$ onto the i-th axis as well as its width, norm, and midpoint is depicted in Figure 19 for $n = 2$.

Let $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$, $[\mathbf{x}] \in [\mathbb{R}]^n$, and $\mathbf{f}\big([\mathbf{x}]\big)$ the image of $[\mathbf{x}]$ under $\mathbf{f}$, given by

$$\mathbf{f}\big([\mathbf{x}]\big) = \big\{\mathbf{f}(\mathbf{x}) \,|\, \mathbf{x} \in [\mathbf{x}]\big\}\,. \tag{121}$$

An interval function $[\mathbf{f}] : [\mathbb{R}]^n \to [\mathbb{R}]^m$ is called *inclusion function* of $\mathbf{f}$ if

$$\mathbf{f}\big([\mathbf{x}]\big) \subseteq [\mathbf{f}]\big([\mathbf{x}]\big), \quad \forall [\mathbf{x}] \in [\mathbb{R}]^n\,. \tag{122}$$

The simplest way of obtaining an inclusion function $[\mathbf{f}]$ of a given function $\mathbf{f}$ is to replace each real variable by their *natural interval extension*. The resulting function is called the *natural inclusion function*. If $[\mathbf{f}]$ is the inclusion function that determines the smallest possible box comprising $\mathbf{f}\big([\mathbf{x}]\big)$ it is called *minimal* and denoted by $[\mathbf{f}]^*\big([\mathbf{x}]\big)$. Figure 20 depicts the image of a box $[\mathbf{x}]$ under $\mathbf{f}$ and two of its possible inclusion functions, one of them the minimal inclusion function $[\mathbf{f}]^*\big([\mathbf{x}]\big)$.

Since the minimal inclusion function is usually not available a so-called contractor can be used to minimise a known inclusion function instead. Before we elaborate on contractors in Section 2.6.3, we will introduce a generic description of problems involving constraints in the next section.

2.6.2 *Constraint Satisfaction Problems*

Utilising the notion of constraints is a formal and declarative way of describing certain problems, abstracting from a domain-specific description in order to allow the solution of the problem with efficient,

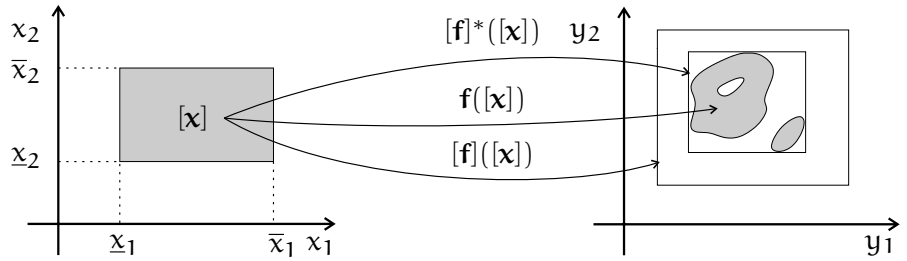


Figure 20: Image $\mathbf{f}([\mathbf{x}])$ of a box $[\mathbf{x}] \in [\mathbb{R}]^2$ under $\mathbf{f} : \mathbb{R}^2 \to \mathbb{R}^2$ and two inclusion functions $[\mathbf{f}]([\mathbf{x}])$ and $[\mathbf{f}]^*([\mathbf{x}])$. The interval function $[\mathbf{f}]^*$ is the minimal inclusion function [73].

generic solution methods. Formally, a *constraint satisfaction problem* (CSP) is defined as a triple $(\mathbb{X}, \mathbb{D}, \mathbb{C})$, where $\mathbb{X} = \{x_1, \ldots, x_n\}$ is a finite set of variables with their associated non-empty domains $\mathbb{D} = \{D_1, \ldots, D_n\}$, so that $x_i \in D_i$, and $\mathbb{C} = \{C_1, \ldots, C_m\}$ is a finite set of constraints. Each of the constraints $C_j \in \mathbb{C}$ in turn is a pair $(\mathbb{T}_j, R_j)$, where $\mathbb{T}_j = \{x_{j,1}, \ldots, x_{j,k}\} \subseteq \mathbb{X}$ is a subset of the variables and R_j is a k-ary relation on the corresponding subset of domains $\mathbb{D}_j = \{D_{j1}, \ldots, D_{jk}\} \subseteq \mathbb{D}$, reducing possible combinations of the values of variables to a subset of $D_{j1} \times \cdots \times D_{jk}$.

If the domains D_i are real intervals $[x_i]$ and the constraints have the form of equalities or inequalities, so that the relation R_j is either of the form

$$f_j(x_1, \ldots, x_k) = 0, \quad j \in \{1, \ldots, m\}, \tag{123}$$

or of the form

$$f_j(x_1, \ldots, x_k) \leqslant 0, \quad j \in \{1, \ldots, m\}, \tag{124}$$

we call this a *numerical constraint satisfaction problem* (NCSP). In the latter case we can introduce a slack variable $x_{sj} \geqslant 0$ to cast the inequality constraint to an equality constraint as follows:

$$f_j(x_1, \ldots, x_k) + x_{sj} = 0, \quad j \in \{1, \ldots, m\}. \tag{125}$$

Now, we combine the n scalar variables to form a real vector $\mathbf{x} = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$ with its domain $[\mathbf{x}_0] = \left([x_1], \ldots, [x_n]\right)^T \in [\mathbb{R}]^n$. Let $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ denote a function whose coordinate functions are given by the f_js. Then, Equation 123 can be written as $\mathbf{f}(\mathbf{x}) = 0$, which corresponds to the constraint satisfaction problem $\mathcal{H}$, formulated as

$$\mathcal{H} : \left(\mathbf{f}(\mathbf{x}) = 0, \mathbf{x} \in [\mathbf{x}_0]\right), \tag{126}$$

with its *solution set* given by

$$\mathbb{S} = \left\{\mathbf{x} \in [\mathbf{x}_0] \mid \mathbf{f}(\mathbf{x}) = 0\right\}. \tag{127}$$

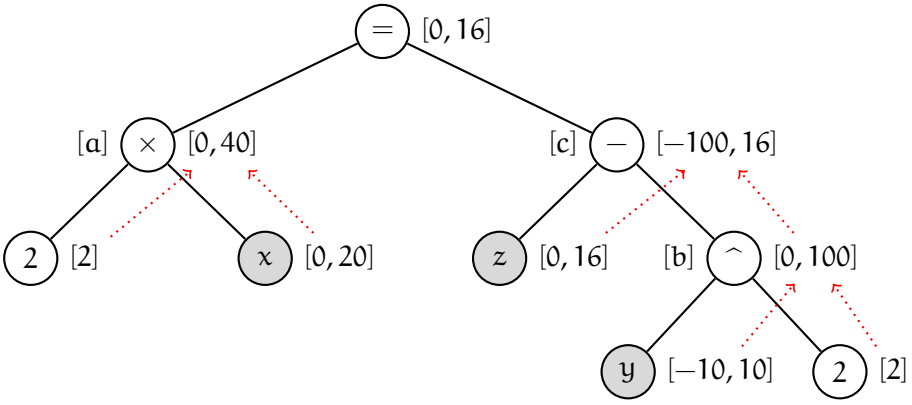


Figure 21: Algorithm HC4: annotated tree for the forward evaluation in the constraint $2x = z - y^2$ and the domains $[x] = [0, 20]$, $[y] = [0, 16]$, and $[z] = [-10, 10]$.

We say a point $\mathbf{x}$ is *feasible* if $\mathbf{x} \in \mathbb{S}$. Then, $\mathbf{x}$ is said to solve the constraint satisfaction problem. Otherwise $\mathbf{x}$ is *infeasible*.

As characterising the solution set is NP-hard in general [73], so-called *consistency techniques* finding outer approximations of $\mathbb{S}$, whilst keeping complexity polynomial in time, have been used [76–78]. One such consistency technique is *contracting* $\mathcal{H}$, which means replacing $[\mathbf{x}]$ by a smaller box $[\mathbf{x}]'$, such that no feasible solution is discarded:

$$\mathbb{S} \subseteq [\mathbf{x}]' \subset [\mathbf{x}]\,. \tag{128}$$

If $[\mathbf{x}]$ is replaced by the smallest possible box $[\mathbf{x}]'$ that contains $\mathbb{S}$ this is called the optimal contraction of $\mathcal{H}$.

2.6.3 *Contractors*

Given the numerical constraint satisfaction problem $\mathcal{H}$, a *contractor* $\mathcal{C} : [\mathbb{R}]^n \to [\mathbb{R}]^n$ is an operator that contracts the box $[\mathbf{x}] \in [\mathbb{R}]^n$ by eliminating values inconsistent with the constraints C_i. It does so without bisecting $[\mathbf{x}]$, so as to keep complexity polynomial. Explicitly, a contractor satisfies the two following properties:

$$\text{Contractance} \quad \mathcal{C}\big([\mathbf{x}]\big) \subseteq [\mathbf{x}], \quad \forall [\mathbf{x}] \in [\mathbb{R}]^n\,, \tag{129}$$

$$\text{Completeness} \quad \mathcal{C}\big([\mathbf{x}]\big) \cap \mathbb{S} = [\mathbf{x}] \cap \mathbb{S}, \quad \forall [\mathbf{x}] \in [\mathbb{R}]^n\,. \tag{130}$$

A simple contractor demanding little computational resources is the forward-backward contractor [79], denoted by $\mathcal{C}_{\uparrow\downarrow}$. The underlying HC4 algorithm [80] uses a tree representation of each of the individual constraints C_i, where leaves correspond to variables or constants, internal nodes correspond to unary or binary primitive operators, and the root node contains the k-ary relation symbol.

After the construction of an expression tree from a single constraint of the given NCSP, the tree is traversed from the leaves to the root and

the subexpression at each node is evaluated using its natural interval extension. In this *forward phase* the domains of the variables are used to obtain interval values for the intermediate nodes of the tree, as is depicted in Figure 21 for the variables x, y, and z, an exemplary constraint $2x = z - y^2$, and the domains $[x] = [0, 20], [y] = [0, 16]$, and $[z] = [-10, 10]$. Adopting this example, as shown in [81], the operations carried out in the forward phase are the following:

$$\begin{array}{lll} 1 & [a] := 2[x] & [0, 40] \\ 2 & [b] := [y]^2 & [0, 100] \\ 3 & [c] := [z] - [b] & [-100, 16] \\ 4 & [r] := [a] - [c] & [0, 16] \end{array}$$

The three intermediate nodes are denoted with a, b and c, respectively, the root node with r, and $:=$ denotes an assignment of the value of the right hand side expression to the interval variable on the left. The intervals in the last column are the values assigned for the above example.

In the *backward phase*, these intervals are then contracted, applying in every node a narrowing operator by isolating the nodes and using inverse operations, yielding reduced domains of the variables [82]. The operations carried out in the backward propagation are the following:

$$\begin{array}{lllll} 5 & [a] := [r] \cap [a] & [0, 16] & // \text{ see Step } 4 \\ 6 & [c] := [r] \cap [c] & [0, 16] & // \text{ see Step } 4 \\ 7 & [z] := \big([c] + [b]\big) \cap [z] & [0, 16] & // \text{ see Step } 3 \\ 8 & [b] := \big([z] - [c]\big) \cap [b] & [0, 16] & // \text{ see Step } 3 \\ 9 & [y] := \sqrt{[b]} \cap [y] & [-4, 4] & // \text{ see Step } 2 \\ 10 & [x] := \big([a]/2\big) \cap [x] & [0, 8] & // \text{ see Step } 1 \end{array}$$

These operations have been derived by isolating each variable on the right hand side of the equations in Step 1 to 4. Steps 5 and 6 result from the interval equality represented by the root node. In `HC4`, the application of cascading projections of primitive constraints is implemented by `HC4Revise`, while the manual isolation in Steps 5 to 10 of this example is only carried out for demonstration purposes. After the two phases have been carried out, each node but the root node contains two interval attributes as depicted in Figure 22. If $[r]$ as computed in Step 4 turns out to be empty, then the NCSP has no solution.

2.6.4 *Set Inversion Problems*

Allowing only domains represented by bounded real intervals, the NCSP reduces to a *set inversion problem*, which is characterised by

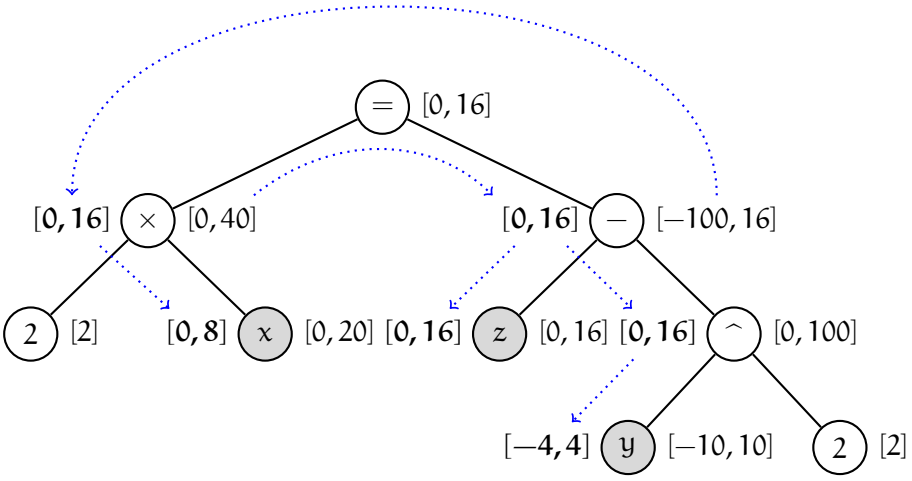


Figure 22: Algorithm HC4: annotated tree for the backward propagation in the constraint $2x = z - y^2$ and the domains $[x] = [0, 20]$, $[y] = [0, 16]$, and $[z] = [-10, 10]$. The results of the backward phase are depicted in bold typeface.

finding the preimage $\mathbb{S}$ of a set $\mathbb{Y} \subset \mathbb{R}^p$ under the possibly non-linear function $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^p$, defined as

$$\mathbb{S} = \mathbf{f}^{-1}(\mathbb{Y}) = \left\{ \mathbf{x} \in \mathbb{R}^n \,|\, \mathbf{f}(\mathbf{x}) \in \mathbb{Y} \right\}. \tag{131}$$

For any bounded set $\mathbb{Y}$ and an inclusion function $[\mathbf{f}] : [\mathbb{R}]^n \to [\mathbb{R}]^p$ of $\mathbf{f}$ two regular subpavings $\underline{\mathbb{S}}$ and $\overline{\mathbb{S}}$, such that

$$\underline{\mathbb{S}} \subseteq \mathbb{S} \subseteq \overline{\mathbb{S}}, \quad \text{with } \overline{\mathbb{S}} = \underline{\mathbb{S}} \cup \Delta\mathbb{S}, \tag{132}$$

may be obtained using a set inversion algorithm. A *subpaving* of a subset $\mathbb{S} \subset \mathbb{R}^n$ is a union of non-overlapping subboxes $[\mathbf{x}_j]$ with non-zero width. Two boxes in the same subpaving may have a non-empty intersection if they have a boundary in common, but their interiors must have an empty intersection. A subpaving is called *regular* when it may be obtained from the initial search box $[\mathbf{x}_0]$ by a finite succession of bisections and selections.

2.6.5 *Set Inverter via Interval Analysis*

A popular set inversion algorithm is the vectorisable [83] non-linear bounded-error estimator SIVIA (Set Inverter via Interval Analysis) introduced by Jaulin and Walter in [84]. Its main idea may be summarised as bisecting and testing the search space, narrowing down the set of feasible solutions.

Given a list $\mathcal{L} = \{[\mathbf{x}_0]\}$ of boxes containing the initial search box $[\mathbf{x}_0]$ to which $\overline{\mathbb{S}}$ is guaranteed to belong, the SIVIA algorithm may encounter the following four cases:

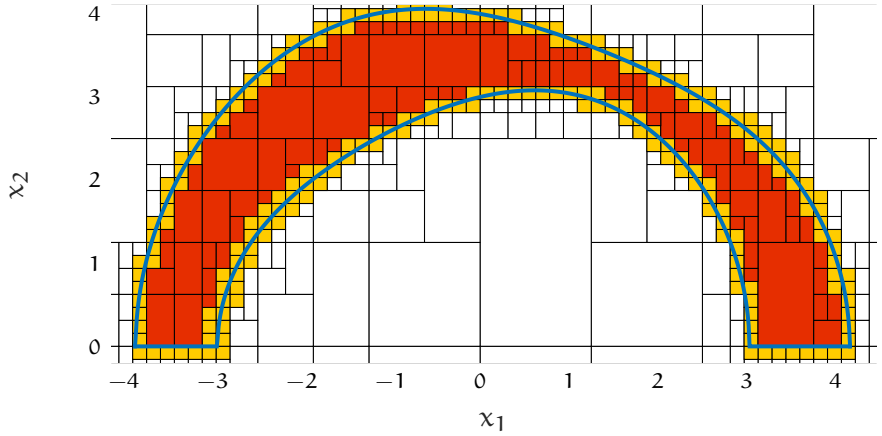


Figure 23: The result of the SIVIA algorithm for $\epsilon = 0.25$ and the solution set given by Equation 135. The inner approximation $\underline{S}$ is depicted in red and ΔS is depicted in yellow, while the blue line bounds the true solution set. The complement of the outer approximation $\overline{S} = \underline{S} \cup \Delta S$, which is depicted in white, does not contain any solutions.

- If $[\mathbf{f}]([\mathbf{x}])$ does not intersect with $\mathbb{Y}$, $[\mathbf{x}]$ is discarded as it does not belong to the solution set $\mathbb{S}$. This follows from the inclusion test

$$[\mathbf{f}]([\mathbf{x}]) \cap \mathbb{Y} = \emptyset \implies [\mathbf{x}] \cap \mathbb{S} = \emptyset\,. \tag{133}$$

- If $[\mathbf{f}]([\mathbf{x}])$ is contained in $\mathbb{Y}$, then $[\mathbf{x}]$ belongs to the solution set and is assigned to the inner approximation $\underline{S}$. This follows from the inclusion test

$$[\mathbf{f}]([\mathbf{x}]) \subset \mathbb{Y} \implies [\mathbf{x}] \subset \mathbb{S}\,. \tag{134}$$

- If $[\mathbf{f}]([\mathbf{x}])$ intersects with $\mathbb{Y}$, but is not contained in $\mathbb{Y}$, $[\mathbf{x}]$ is bisected, given its width is bigger than a predefined limit ϵ, and the recursion is entered by adding it to $\mathcal{L}$.

- If the width of $[\mathbf{x}]$ is not bigger than ϵ, $[\mathbf{x}]$ is stored in ΔS and therefore belongs to the outer approximation $\overline{S}$.

These steps are carried out while $\mathcal{L}$ is non-empty. The parameter $\epsilon > 0$ determines the maximum width of the boxes in ΔS and thus the precision of the result. Algorithm 1 presents the Set Inverter via Interval Analysis. Figure 23 depicts the result of the SIVIA algorithm for $\epsilon = 0.25$ and the solution set

$$\begin{aligned} \mathbb{S} = \big\{(x_1, x_2) \in \mathbb{R}^2 \,|\, & x_1^2 + x_2^2 + 2\sin(x_1) \geqslant 9, \\ & x_1^2 + x_2^2 - 2\sin(x_1) \leqslant 16, x_2 \geqslant 0\big\}\,. \end{aligned} \tag{135}$$

The figure was generated using the toolbox developed in [82].

Algorithm 1 `SIVIA` (Set Inverter via Interval Analysis) [73]

Input: $[\mathbf{x}_0] \in [\mathbb{R}]^n, \quad \mathbb{Y} \in \mathbb{R}^p, \quad [\mathbf{f}] : [\mathbb{R}]^n \to [\mathbb{R}]^p, \quad \epsilon > 0$
Output: $\underline{\mathbb{S}}, \overline{\mathbb{S}}$ such that $\underline{\mathbb{S}} \subseteq \mathbb{S} \subseteq \overline{\mathbb{S}}$, with $\overline{\mathbb{S}} = \underline{\mathbb{S}} \cup \Delta\mathbb{S}$ and $w\left([\mathbf{x}]\right) \leqslant \epsilon \quad \forall\, [\mathbf{x}] \in \Delta\mathbb{S}$

```
1:  function SIVIA([x_0], Y, [f], ε)
2:      L ← {[x_0]}
3:      S̲ ← S̄ ← ΔS ← ∅
4:      while L ≠ ∅ do
5:          [x] ← POP(L)            ▷ retrieve and remove a box from the list
6:          if f([x]) ∩ Y = ∅ then          ▷ [x] does not belong to S
7:              discard [x]
8:          else if f([x]) ⊂ Y then          ▷ [x] belongs to S
9:              S̲ ← S̲ ∪ [x]
10:         else if w([x]) ⩽ ε then          ▷ [x] belongs to ΔS
11:             ΔS ← ΔS ∪ [x]
12:         else                             ▷ cannot be decided
13:             bisect [x] into [x_1] and [x_2]
14:             PUSH(L, [x_1])               ▷ add box to the list
15:             PUSH(L, [x_2])
16:         end if
17:     end while
18:     S̄ ← S̲ ∪ ΔS
19:     return (S̲, S̄)          ▷ return inner and outer approximation
20: end function
```

With sufficient computational resources, inner and outer interval approximations of the solution set can be made arbitrarily precise and can therefore provide a good estimate of the real shape of the set [82]. However, subpavings are adapted to low-dimensional problems, since they form an expensive representation of sets in terms of memory space. Both contractors and `SIVIA` can be combined in order to improve the performance of the latter. Reducing the initial search box for the `SIVIA` algorithm by contracting it first can lower the number of bisections and may therefore lead to reduced computational cost.

3

THE HYBRID LOCALISATION ALGORITHMS

In this chapter we will describe in detail the four new hybrid localisation algorithms, the key idea of which is to only perform Monte Carlo localisation over a delimited region of the search space that is constituted of feasible robot positions. This delimited region is determined using the two bounded-error estimators `HC4` and `SIVIA`, as described in Section 2.6.3 and 2.6.5, respectively. Thus, by increasing the particle density in interesting regions, the localisation accuracy should increase. Specifically, we will present

- a bootstrap particle filter with `HC4` contractor (`PFC`),
- an unscented particle filter with `HC4` contractor (`UPC`),
- a bootstrap particle filter with `SIVIA` (`PFS`),
- and an unscented particle filter with `SIVIA` (`UPFS`).

Depending on the amount of available information in terms of the number of visible landmarks, the delimited region may still be large. Therefore, in addition to the simple bootstrap particle filter, which is simply denoted as particle filter in the following, an unscented particle filter was employed in combination with the respective bounded-error estimator.

As described in Section 2.5.3, the unscented particle filter can move particles to regions of the search space that are associated with a high observation likelihood and therefore potentially improve the estimation accuracy, especially when dealing with very accurate measurements. As no particles remain in regions where they would receive an insignificantly low importance weight and therefore simply die in the next resampling step, it may also be possible to further reduce the number of particles, when compared to the particle filter. Finally, the speed of convergence should be influenced positively by actively moving particles towards the peak of the likelihood function.

In a symmetric environment both conventional probabilistic filters may possibly converge to the wrong place and may not be able to recover from this. Using geometrical considerations of the environment, which are further explained below, as well as basic plausibility considerations, convergence to the wrong position can be prevented by excluding the opposing solution from the set of feasible solutions and therefore bringing the initial particles closer to the peak of the true posterior distribution.

A fundamental concept that we will need for the introduction of the state-space and bounded-error model in Section 3.2 and 3.3, respectively, is the attitude of a robot, which we will elaborate on in the following section. In Section 3.4, we show how constraints are incorporated in the inherently unconstraint generic particle filter framework. Finally, the four new localisation algorithms are presented in Section 3.5.

3.1 ATTITUDE REPRESENTATION

The orientation of the coordinate frame of a robot, referred to as the *body frame*, with respect to a reference coordinate frame, termed the *world frame*, is known as *attitude*. The origin of the body coordinate system is usually chosen to be the robot's center of gravity. Establishing correspondence between the two frames is one of the goals of the localisation process. In the following section, we will introduce a common attitude representation known as Euler angles.

3.1.1 *Euler Angles*

Euler angles are one of several mathematical ways to describe the attitude of an object in three-dimensional Euclidean space. They represent a sequence of three elemental rotations about the axes of the world coordinate system, defined as follows:

- The *roll angle* ϕ determines the rotation around the x-axis.
- The *pitch angle* θ determines the rotation around the y-axis.
- The *yaw angle* ψ determines the rotation around the z-axis.

Figure 24 depicts the rotation about the axes z, y', X by ψ, θ, ϕ, respectively, according to the Tait-Bryan convention. The colour blue indicates the world frame $\{x, y, z\}$, which matched the body frame $\{X, Y, Z\}$ before the rotations. The colour red indicates the orientation of the body frame after the rotations were carried out. In contrast to *extrinsic rotations*, where each of the three elemental rotations may occur about the axes of the original coordinate system, the Tait-Bryan rotations are *intrinsic rotations* that occur about the axes of the rotating coordinate system, which changes its orientation after each rotation.

Euler angles are a simple and intuitive means to represent rotations in three-dimensional space. However, for the above mentioned parameterisation they have singularities at values of $\theta = n\pi$, $n \in \mathbb{Z}$. At these points a rotation about the x-axis and the z-axis constitute the same motion, which results in the loss of one degree of freedom and makes changes in ϕ and ψ indistinguishable. This phenomenon is called *gimbal lock*.

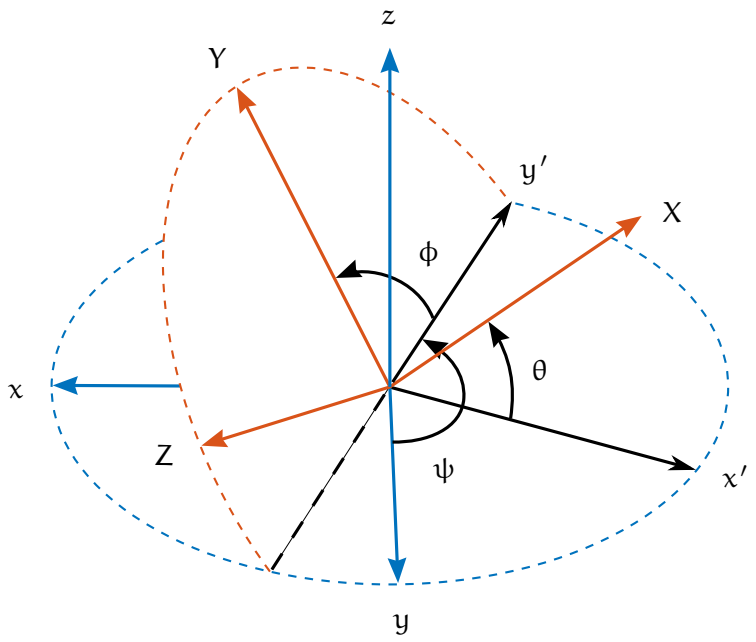


Figure 24: Representation of the body frame, depicted in red, with respect to the world frame, depicted in blue [85]. The body frame was rotated by the Euler angles ψ, θ, ϕ about the axes z, y', X, respectively.

3.1.2 *Transformation Matrices*

Coordinates representing a point in one coordinate system can be transformed to another, given the angles between the two coordinate systems are known. Such a transformation can be expressed as a multiplication of a matrix with the coordinate vector that is to be transformed. Let $\mathbf{E}$ denote the orthonormal basis $\{x, y, z\} \in \mathbb{R}^3$ and let $\mathbf{E}'$ denote the orthonormal basis $\{X, Y, Z\} \in \mathbb{R}^3$. Furthermore, let $\mathbf{p}$ denote the position vector of an arbitrary point in three-dimensional Euclidean space, expressed in terms of $\mathbf{E}$. The coordinate transformation that maps $\mathbf{p}$ to its representation in $\mathbf{E}'$ is given by the linear transformation $\boldsymbol{\Omega}_{\mathbf{E}\to\mathbf{E}'}$,

$$\begin{aligned} \boldsymbol{\Omega}_{\mathbf{E}\to\mathbf{E}'} : \mathbb{R}^3 &\to \mathbb{R}^3 \\ \mathbf{p} &\mapsto \mathbf{T}(\psi, \theta, \phi)\mathbf{p}\,, \end{aligned} \tag{136}$$

where the *transformation matrix* $\mathbf{T}$ is a function of the rotation angles ψ, θ, ϕ between the coordinate axes.

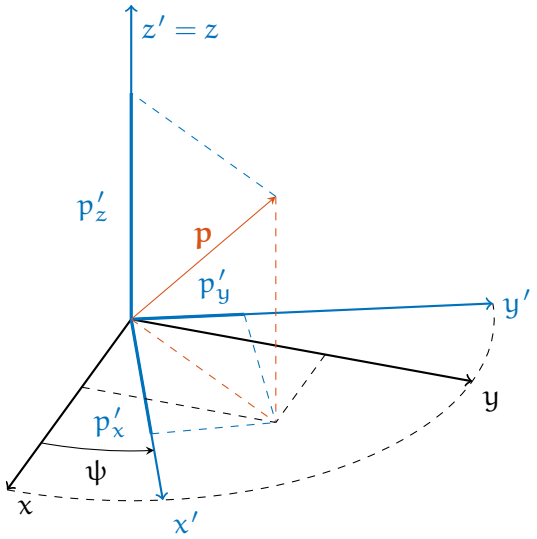


Figure 25: An exemplary coordinate rotation about the z-axis by an angle ψ, illustrating the orthogonal projection (p'_x, p'_y, p'_z) of the position vector $\mathbf{p}$ on the resulting axes x', y', z'.

In order to transform a coordinate vector from the world frame to the body frame, according to the common aerospace rotation sequence mentioned above, the transformation matrix $\mathbf{T}_{wb}$ is given by

$$\begin{aligned}\mathbf{T}_{wb} &= \mathbf{T}_x(\phi)\mathbf{T}_y(\theta)\mathbf{T}_z(\psi) \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\phi\cos\theta \end{bmatrix}\end{aligned} \tag{137}$$

Plugged in Equation 136, the pre-multiplications of the matrices $\mathbf{T}_x(\phi)$, $\mathbf{T}_y(\theta)$, and $\mathbf{T}_z(\psi)$ to the vector $\mathbf{p}$ represent the coordinate rotations about the single axes x, y', Z, according to the right hand rule, respectively. That is, the function $\mathbf{\Omega}_{E \to E'}$ maps the vector $\mathbf{p}$ to its orthogonal projection onto the axes of the coordinate system that results from the respective two-dimensional rotation of ϕ, θ, ψ about the axes x, y', Z. This is illustrated for a single rotation around the z-axis by an angle ψ in Figure 25. Note that $\{x', y', z'\}$ denotes the coordinate frame after the first elemental rotation. The matrices $\mathbf{T}_x(\phi)$, $\mathbf{T}_y(\theta)$, and $\mathbf{T}_z(\psi)$ are also known as *direction cosine matrices*, since their elements are the cosines of the unsigned angles between the body-fixed axes and the axes of the world frame, as shown in [86]. The Blender Game Engine [87] used for the simulations in Chapter 4 assumes a right-handed east-north-up coordinate system. Then, the matrix $\mathbf{T}_{bw}$ that transforms a coordinate vector from the body frame to the world frame is given by

$$\mathbf{T}_{bw} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}. \tag{138}$$

Note that $\mathbf{T}_{bw} = \mathbf{T}_{wb}^{\mathsf{T}} = \mathbf{T}_{wb}^{-1}$. Thus, $\mathbf{T}_{bw}$ and $\mathbf{T}_{wb}$ are orthogonal matrices so that $\mathbf{T}_{bw}\mathbf{T}_{wb} = \mathbf{I}_3$, where $\mathbf{I}_3 \in \mathbb{R}^{3\times 3}$ is the identity matrix. This makes sense as forward and backward rotations should not alter the vector $\mathbf{p}$.

3.2 THE PROBABILISTIC STATE-SPACE MODEL

In this section we will derive the state-space model for the particle filters, which is comprised of two components, i. e. the system model and the measurement model, in accordance with Equations 1 and 2.

3.2.1 *The System Model*

The plant dynamics of a continuous-time system can be expressed as a set of n_x coupled first-order ordinary differential equations, known as the *state equations* [88],

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \tag{139}$$

where $\dot{\mathbf{x}}$ consists of the component-wise time derivatives of the state vector $\mathbf{x}$, expressed in terms of the state variables $x_1(t), \ldots, x_{n_x}(t)$, and the control inputs $u_1(t), \ldots, u_{n_u}(t)$. Explicitly, its elements are given by

$$\dot{x}_i = f_i(\mathbf{x}, \mathbf{u}, t) = \frac{dx_i}{dt}, \quad i \in \{1, \ldots, n_x\}. \tag{140}$$

Given this state-space representation, the system state at any instant t may be interpreted as a point in an n_x-dimensional state space whose axes are the state variables. The dynamic state response $\mathbf{x}(t)$ can be interpreted as a trajectory traced out in the state space.

Discretisation of the State Equations

Given a *linear time-invariant* system in state-space form,

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{G}\mathbf{u}(t), \tag{141}$$

and its solution

$$\mathbf{x}(t) = e^{\mathbf{A}(t-t_0)}\mathbf{x}(t_0) + \int_{t_0}^{t} e^{\mathbf{A}(t-\tau)}\mathbf{G}\mathbf{u}(\tau)d\tau, \tag{142}$$

we shall discretise the continuous-time system using a discrete time index k, such that

$$t = kT_s, \tag{143}$$

where T_s is the sampling period. Our goal then is to determine a dynamical relation of the following form:

$$\mathbf{x}_{k+1} = \mathbf{\Phi}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \quad k \in \{0, 1, 2, \dots\}, \tag{144}$$

where we replaced $\mathbf{x}(kT_s)$ by $\mathbf{x}_k$ and $\mathbf{u}(kT_s)$ by $\mathbf{u}_k$ for the sake of brevity.

In the following derivation of the matrices $\mathbf{\Phi}$ and $\mathbf{B}$, we assume the control input $\mathbf{u}(t)$ is constant over the sampling interval, such that the following holds:

$$\mathbf{u}(t) = \mathbf{u}(kT_s), \quad kT_s \leqslant t < (k+1)T_s\,. \tag{145}$$

Setting $t_0 = kT_s$ and $t = (k+1)T_s$ in Equation 142 yields

$$\mathbf{x}\big((k+1)T_s\big) = e^{\mathbf{A}T_s}\mathbf{x}(kT_s) + \int_{kT_s}^{(k+1)T_s} e^{\mathbf{A}\left[(k+1)T-\tau\right]}\mathbf{G}\mathbf{u}(\tau)d\tau\,. \tag{146}$$

Using that the control input is constant within the interval from kT_s to $(k+1)T_s$, as is the matrix $\mathbf{G}$, we can rewrite $\mathbf{x}\big((k+1)T_s\big)$ as

$$\mathbf{x}\big((k+1)T_s\big) = e^{\mathbf{A}T_s}\mathbf{x}(kT_s) + \int_{kT_s}^{(k+1)T_s} e^{\mathbf{A}\left[(k+1)T-\tau\right]}d\tau\mathbf{G}\mathbf{u}(kT_s)\,, \tag{147}$$

or shorter as

$$\mathbf{x}_{k+1} = e^{\mathbf{A}T_s}\mathbf{x}_k + \int_{kT_s}^{(k+1)T_s} e^{\mathbf{A}\left[(k+1)T-\tau\right]}d\tau\mathbf{G}\mathbf{u}_k\,. \tag{148}$$

Changing variables, such that $(k+1)T - \tau = \lambda$, yields

$$\mathbf{x}_{k+1} = e^{\mathbf{A}T_s}\mathbf{x}_k + \int_0^{T_s} e^{\mathbf{A}\lambda}d\lambda\mathbf{G}\mathbf{u}_k\,. \tag{149}$$

By comparison with Equation 144 we may now identify the matrices $\mathbf{\Phi}$ and $\mathbf{B}$ as

$$\mathbf{\Phi} = e^{\mathbf{A}T_s}\,, \tag{150}$$

$$\mathbf{B} = \int_0^{T_s} e^{\mathbf{A}\lambda}d\lambda\mathbf{G}\,. \tag{151}$$

In order to obtain the state transition matrix $\mathbf{\Phi}$, as outlined in [40], we can use a Taylor-series expansion of $e^{\mathbf{A}T_s}$,

$$\begin{aligned}\mathbf{\Phi} = e^{\mathbf{A}T_s} &= \sum_{n=0}^{\infty}\frac{\mathbf{A}^n T_s^n}{n!} \\ &= \mathbf{I}_{n_x} + \mathbf{A}T_s + \frac{\mathbf{A}^2T_s^2}{2!} + \frac{\mathbf{A}^3T_s^3}{3!} + \dots\,,\end{aligned} \tag{152}$$

where $\mathbf{I}_{n_x} \in \mathbb{R}^{n_x \times n_x}$ is the identity matrix. Truncating the Taylor series after the first order terms yields the *linear approximation* of the state transition matrix:

$$\boldsymbol{\Phi} \approx \mathbf{I}_{n_x} + \mathbf{A}T_s \,. \tag{153}$$

Using a Taylor series expansion of Equation 151 and integrating term-by-term, we may obtain $\mathbf{B}$ as

$$\begin{aligned}
\mathbf{B} &= \int_0^{T_s} e^{\mathbf{A}\lambda} d\lambda \mathbf{G} \\
&= \int_0^{T_s} \left(\mathbf{I}_{n_u} + \mathbf{A}\lambda + \frac{\mathbf{A}^2\lambda^2}{2!} + \frac{\mathbf{A}^3\lambda^3}{3!} + \cdots \right) d\lambda \mathbf{G} \\
&= \left(\int_0^{T_s} \mathbf{I}_{n_u} d\lambda + \int_0^{T_s} \mathbf{A}\lambda d\lambda + \int_0^{T_s} \frac{\mathbf{A}^2\lambda^2}{2!} d\lambda + \int_0^{T_s} \frac{\mathbf{A}^3\lambda^3}{3!} d\lambda + \cdots \right) \mathbf{G} \\
&= \mathbf{G}T_s + \frac{\mathbf{A}\mathbf{G}T_s^2}{2!} + \frac{\mathbf{A}^2\mathbf{G}T_s^3}{3!} + \dots \,.
\end{aligned} \tag{154}$$

Again, taking the first term only, such that

$$\mathbf{B} \approx \mathbf{G}T \,, \tag{155}$$

and plugging $\mathbf{B}$ together with $\boldsymbol{\Phi}$ of Equation 153 into Equation 144 yields Euler's forward approximation:

$$\mathbf{x}_{k+1} \approx (\mathbf{I}_{n_x} + \mathbf{A}T_s)\mathbf{x}_k + \mathbf{G}\mathbf{u}_k T_s, \quad k \in \{0, 1, 2, \dots\}. \tag{156}$$

However, Equations 152 and 154 can easily be implemented in software. Therefore, terminating the series when the terms become smaller than some desired threshold provides arbitrarily precise results [89].

Discretisation of the Modelled System

The state vector of the robot is given by

$$\mathbf{x}_k = \begin{bmatrix} x_k, & y_k, & z_k \end{bmatrix}^T \in \mathbb{R}^{n_x = 3}, \quad k \in \{0, 1, 2, \dots\}, \tag{157}$$

where its elements correspond to the coordinates describing the three-dimensional position (x_k, y_k, z_k) of the robot at time step k, expressed in the world frame. The plant dynamics of the modelled system are governed by the linear differential equation

$$\dot{\mathbf{x}} = \mathbf{A}\boldsymbol{x}(t) + \mathbf{G}\mathbf{u}(t) = \begin{bmatrix} v_x^{[w]}(t) \\ v_y^{[w]}(t) \\ v_z^{[w]}(t) \end{bmatrix} \,, \tag{158}$$

with the control input $\mathbf{u}(t) = \begin{bmatrix} v_x^{[w]}(t), v_y^{[w]}(t), v_z^{[w]}(t) \end{bmatrix}^T$, whose components denote the nominal linear velocity in x, y, and z direction,

respectively, stated in terms of the world frame. Equating coefficients in Equation 158 lets us identify the remaining variables as $\mathbf{A} = 0$ and $\mathbf{G} = \mathbf{I}_{n_u}$. Plugging $\mathbf{A}$ and $\mathbf{G}$ into Equations 152 and 154 yields $\mathbf{\Phi} = \mathbf{I}_{n_x}$ and $\mathbf{B} = \mathbf{I}_{n_u} T_s$. Plugging now both $\mathbf{\Phi}$ and $\mathbf{B}$ into Equation 144, we have

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{u}_k T_s, \quad k \in \{0, 1, 2, \dots\}. \tag{159}$$

Since the robot's nominal linear velocity is naturally stated in terms of its body frame, as $\mathbf{u}_k^{[b]} = \left[v_{x,k}^{[b]}, v_{y,k}^{[b]}, v_{z,k}^{[b]}\right]^T$, it has to be transformed to the world frame according to Equation 136, using the transformation matrix $\mathbf{T}_{bw}(\psi_k, \theta_k, \phi_k)$ of Equation 138,

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{T}_{bw}(\psi_k, \theta_k, \phi_k)\mathbf{u}_k^{[b]} T_s, \quad k \in \{0, 1, 2, \dots\} \tag{160}$$

Note that with the state equations at hand this is the best model available and a consequence of $\mathbf{A} = 0$. Although the form is similar, contrary to the claim in [29], this result is no Euler approximation but instead reflects the limited knowledge about the modelled system.

Adhering to the notation used above, we subtract 1 from each discrete time index k and add noise terms to the individual variables, to take into account the inherent uncertainty in the model,

$$\begin{aligned} \mathbf{x}_k &= \boldsymbol{\phi}_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \boldsymbol{w}_{k-1}) \\ &= \mathbf{x}_{k-1} + \mathbf{T}_{bw}\left(\psi_{k-1} + w_{\psi,k-1}, \theta_{k-1} + w_{\theta,k-1}, \phi_{k-1} + w_{\phi,k-1}\right) \\ &\qquad \cdot \left(\mathbf{u}_{k-1}^{[b]} + \boldsymbol{w}_{v,k-1}\right) T_s, \quad k \in \{1, 2, 3, \dots\}. \end{aligned} \tag{161}$$

The process noise vector $\boldsymbol{w}_k = \left[\boldsymbol{w}_{v,k}, \boldsymbol{w}_{\text{Eul},k}\right] \in \mathbb{R}^{n_w = 6}$, with $\boldsymbol{w}_{\text{Eul},k} = \left[w_{\psi,k}, w_{\theta,k}, w_{\phi,k}\right]^T$ and $\boldsymbol{w}_{v,k} = \left[w_{v_x,k}, w_{v_y,k}, w_{v_z,k}\right]^T$, is assumed to be normally distributed with zero mean and constant covariance matrix $\mathbf{Q}_k \in \mathbb{R}^{n_w \times n_w}$, given by

$$\mathbf{Q}_k = \begin{bmatrix} \sigma_v^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_v^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_v^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{\text{Eul}}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\text{Eul}}^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{\text{Eul}}^2 \end{bmatrix}, \quad k \in \{0, 1, 2, \dots\}. \tag{162}$$

Due to the relation $\text{Cov}(w_i, w_i) = \text{Var}(w_i)$, the diagonal elements represent the respective variances of the elements $w_1, \dots, w_n$ of the process noise vector $\boldsymbol{w}$. The other elements are the covariances of all possible pairs of the random variables of the process noise vector. The

noise processes interfering with the individual Euler angles and the velocity inputs, respectively, are modelled as uncorrelated so that the remaining elements of $\mathbf{Q}_k$ are zero.

With the process model of Equation 161 at hand, we can now state the transition prior probability distribution, which is employed in the computation of the importance weights of the unscented particle filter, as a Gaussian with mean $\mathbf{x}_k - \boldsymbol{\phi}_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0})$ and covariance $\mathbf{Q}_k$,

$$p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \mathbf{u}_{k-1}) = \mathcal{N}\big(\mathbf{x}_k - \boldsymbol{\phi}_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0}), \mathbf{Q}_k\big)\,. \tag{163}$$

3.2.2 *The Measurement Model*

Provided the position of n_z landmarks is known, theoretically the position of the robot can be computed from the distances between the robot and multiple landmarks using lateration. Considering the distance between the robot and one landmark alone, possible positions the robot may have are situated on a sphere around the landmark, with its radius equal to the distance to the robot. In order to unambiguously determine the exact position in three-dimensional space four landmarks are required. Then, their surrounding spheres with the respective radii d_i ideally intersect in one point identical with the robot's position. Mathematically, lateration can be expressed as finding the solution of the following system of quadratic equations:

$$(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 = d_i^2, \quad i \in \{1, \ldots, n_z\}, \tag{164}$$

where (x, y, z) are the coordinates of the robot and (x_i, y_i, z_i) are the known coordinates of the i-th of n_z landmarks in three-dimensional space, respectively. For the sake of clarity of presentation, Figure 26 depicts exemplary the scenario for the determination of the two-dimensional position (x, y) of a robot and the known positions (x_i, y_i) of the landmarks $i \in \{1, 2, 3\}$.

Due to measurement noise, the radii morph into concentric spheres around the respective landmarks, so that there is not a precise intersection point anymore and simply solving a system of quadratic equations is not sufficient to determine the robot's position. Incorporating the uncertainty into the model using the noise term $\boldsymbol{\nu}_k$ yields the measurement vector $\mathbf{z}_k \in \mathbb{R}^{n_z}$ of the modelled system as

$$\mathbf{z}_k = \begin{bmatrix} d_{1,k} \\ d_{2,k} \\ \vdots \\ d_{n_z,k} \end{bmatrix} = \begin{bmatrix} \sqrt{(x_k - x_1)^2 + (y_k - y_1)^2 + (z_k - z_1)^2} \\ \sqrt{(x_k - x_2)^2 + (y_k - y_2)^2 + (z_k - z_2)^2} \\ \vdots \\ \sqrt{(x_k - x_{n_z})^2 + (y_k - y_{n_z})^2 + (z_k - z_{n_z})^2} \end{bmatrix} + \boldsymbol{\nu}_k\,, \tag{165}$$

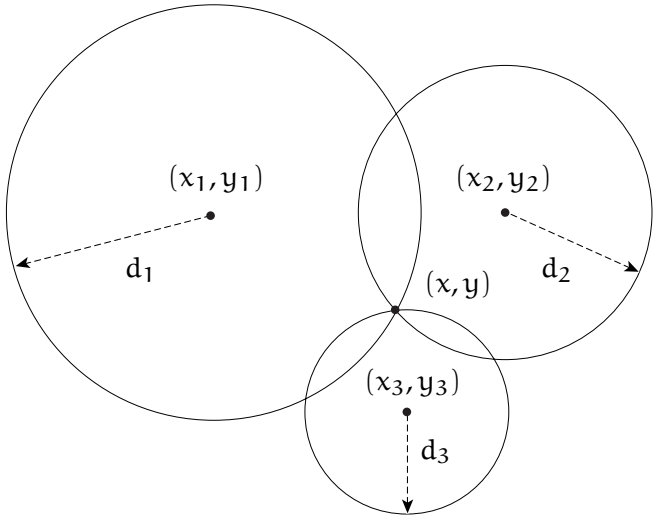


Figure 26: Trilateration: depicted is the determination of the two-dimensional position (x, y) of a robot from the distances d_i to the three landmarks $i \in \{1, 2, 3\}$ with known positions (x_i, y_i).

where the d_i-s represent the distance between the robot and the i-th of n_z landmarks and the measurement noise process $\boldsymbol{\nu}_k \in \mathbb{R}^{n_\nu}$ is modelled as zero-mean Gaussian white noise. The length n_z of the measurement vector is determined by the number of landmarks and is kept variable intentionally for experiments with different numbers of landmarks. The constant measurement noise covariance matrix $\mathbf{R}_k \in \mathbb{R}^{n_\nu \times n_\nu}$ is given by

$$\mathbf{R}_k = \begin{bmatrix} \sigma_d^2 & 0 & 0 & 0 \\ 0 & \sigma_d^2 & 0 & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & \sigma_d^2 \end{bmatrix} . \tag{166}$$

With the measurement model of Equation 165 at hand, we can now state the likelihood function to be a Gaussian with mean $\boldsymbol{z}_k - \mathbf{h}_k(\boldsymbol{x}_k, \mathbf{0})$ and covariance $\mathbf{R}_k$,

$$p(\boldsymbol{z}_k \,|\, \boldsymbol{x}_k) = \mathcal{N}\big(\boldsymbol{z}_k - \mathbf{h}_k(\boldsymbol{x}_k, \mathbf{0}), \mathbf{R}_k\big) . \tag{167}$$

3.3 THE BOUNDED-ERROR MODEL

In bounded-error state estimation, all model and measurement errors are assumed to vary around a central value within certain bounds [90]. Only when the bounds are chosen conservatively enough, the estimation result may be trusted. Under real-life conditions, however, the measurement error is usually normally distributed and therefore inherently unbound. The probability that a normal deviate x, with

mean μ and standard deviation σ, lies in the range between $\mu - \xi\sigma$ and $\mu + \xi\sigma$, with $\xi \in \mathbb{R}^+$, can be computed using the Gauss error function as

$$\Pr\big((x > \mu - \xi\sigma) \wedge (x < \mu + \xi\sigma)\big) = \operatorname{erf}\left(\frac{\xi}{\sqrt{2}}\right) = \frac{1}{\sqrt{\pi}} \int_{-\frac{\xi}{\sqrt{2}}}^{\frac{\xi}{\sqrt{2}}} e^{-t^2}\, dt\,. \tag{168}$$

To obtain a desired confidence interval, we inflate the real quantity x to an interval $[x]$ as follows:

$$[x] = \big[x - \xi\sigma, x + \xi\sigma\big]\,. \tag{169}$$

For $\xi = 3$ this leads to a probability that the interval covers a sample from the above mentioned Gaussian distribution of 99.73 percent.

3.3.1 *The System Model*

The robot's interval state vector $[\mathbf{x}_k] \in [\mathbb{R}]^{n_x=3}$ is given by

$$[\mathbf{x}_k] = \Big[\big[\underline{x}_k, \overline{x}_k\big],\quad \big[\underline{y}_k, \overline{y}_k\big],\quad \big[\underline{z}_k, \overline{z}_k\big]\Big]^T,\quad k \in \{0, 1, 2, \dots\}. \tag{170}$$

To predict the interval position $[\mathbf{x}_k]$, given the previous interval position $[\mathbf{x}_{k-1}]$, it requires an inclusion function $[\boldsymbol{\phi}_k] : [\mathbb{R}]^{n_x} \to [\mathbb{R}]^{n_x}$ of the state transition function $\boldsymbol{\phi}_k$. A straightforward way to obtain an inclusion function is replacing each expression in the system model given by Equation 161 by its natural interval extension. Then we have

$$\begin{aligned}[\mathbf{x}_k] = [\boldsymbol{\phi}_{k-1}]\big([\mathbf{x}_{k-1}], [\mathbf{u}_{k-1}], [\boldsymbol{w}_{k-1}]\big) = [\mathbf{x}_{k-1}] \\ +[\mathbf{T}_{bw}]\big([\psi_{k-1}] + [w_{\psi,k-1}], [\theta_{k-1}] + [w_{\theta,k-1}], [\phi_{k-1}] + [w_{\phi,k-1}]\big) \\ \cdot \big([\mathbf{u}_{k-1}^{[b]}] + [\boldsymbol{w}_{v,k-1}]\big) T_s,\quad k \in \{1, 2, 3, \dots\},\end{aligned} \tag{171}$$

where $[\mathbf{T}_{bw}]$ denotes the natural interval extension of the transformation matrix $\mathbf{T}_{bw}$ of Equation 138.

3.3.2 *The Measurement Model*

Extending Equation 178 to intervals as follows, with $\boldsymbol{\nu}_k = 0$, yields the bounded-error measurement model $[\mathbf{h}_k] : [\mathbb{R}]^{n_x} \to [\mathbb{R}]^{n_z}$, which is used to construct the set $\mathbb{C}_k$ of n_z constraints of the form

$$
\begin{aligned}
[\mathbf{z}_k] &= \begin{bmatrix} [d_{1,k} - \xi\sigma_d, d_{1,k} + \xi\sigma_d] \\ [d_{2,k} - \xi\sigma_d, d_{2,k} + \xi\sigma_d] \\ \vdots \\ [d_{n_z,k} - \xi\sigma_d, d_{n_z,k} + \xi\sigma_d] \end{bmatrix} \\
&= [\mathbf{h}_k]\big([\mathbf{x}_k], 0\big) \qquad (172) \\
&= \begin{bmatrix} \sqrt{([x_k] - x_1)^2 + ([y_k] - y_1)^2 + ([z_k] - z_1)^2} \\ \sqrt{([x_k] - x_2)^2 + ([y_k] - y_2)^2 + ([z_k] - z_2)^2} \\ \vdots \\ \sqrt{([x_k] - x_{n_z})^2 + ([y_k] - y_{n_z})^2 + ([z_k] - z_{n_z})^2} \end{bmatrix}.
\end{aligned}
$$

This model is motivated by the following notion. Given a noisy measurement z_k the true distance lies in the interval $[z_k]$, with a probability that can be determined by Equations 168. The concentric spheres described by the above constraints will overlap in certain regions of the search space. The region that is confined by the respective inner and outer sphere satisfies all the constraints and thus contains feasible robot positions. On the other hand, any point that lies outside does not satisfy the constraints and is therefore no solution.

3.4 CONSTRAINED PARTICLE FILTERS

The above Bayesian filtering techniques have been developed for unconstrained conditions, which means that essentially any estimate in the state space is a possible solution to the filtering problem. However, in practical applications additional information about a process in the form of constraints are commonly encountered [91]. These constraints may stem from physical laws, model restrictions or technological limitations [92]. Taking into account such important supplementary information about the state may improve the state estimation performance [93]. The aim of constrained Bayesian filtering then is to find an estimate of the posterior probability distribution $p_C(\mathbf{x}_k \,|\, \mathbf{Z}_k, \mathbf{U}_{k-1})$ subject to the constraints,

$$
\begin{aligned}
p_C(\mathbf{x}_k \,|\, \mathbf{Z}_k, \mathbf{U}_{k-1}) &= p(\mathbf{x}_k \,|\, \mathbf{Z}_k, \mathbf{U}_{k-1}, \mathbf{x}_k \in \mathbb{S}_k) \\
&= \begin{cases} \zeta_k^{-1} p(\mathbf{x}_k \,|\, \mathbf{Z}_k, \mathbf{U}_{k-1}) & \text{if } \mathbf{x}_k \in \mathbb{S}_k \\ 0 & \text{otherwise} \end{cases}, \qquad (173)
\end{aligned}
$$

with the normalising constant

$$\zeta_k = \Pr(\mathbf{x}_k \in \mathbb{S}_k \,|\, \mathbf{Z}_k, \mathbf{U}_{k-1}) = \int_{\mathbb{S}_k} p(\mathbf{x}_k \,|\, \mathbf{Z}_k, \mathbf{U}_{k-1})\, d\mathbf{x}_k \tag{174}$$

and the set $\mathbb{S}_k$ of states satisfying the constraints, according to Equation 127.

3.4.1 *Clipping*

A very simple approach to satisfy linear constraints is *clipping*, which denotes mapping a point estimate lying outside the constrained region into it or project it right onto the boundary [94]. We will use clipping for all six filters used for the experiments described in the next chapter. That is, whenever a particle or a sigma point lies outside the map, which is assumed to have linear boundaries parallel to one of the three axes of the world coordinate system, it is projected onto the boundary. Especially the covariance estimate of the unscented Kalman filter is positively influenced by mapping the sigma points instead of the estimate itself [95]. In our estimation scenario, the clipping approach reflects a plausibility assessment of the environment. For instance, given a map as for the experiments below that assumes a position in z-direction of 0 metres to be the sea surface, the underwater robot cannot be located above it, so that $z \leqslant 0$ holds for every position estimate.

3.4.2 *Non-linear Constraints*

When dealing with multiple non-linear interval constraints according to Equation 172, the projection on the boundary may not be straightforward in practice. Instead, one may simply discard particles outside the constrained regions and thereby trim the conditional probability distribution of the state with respect to the constraints, whilst preserving the shape of the PDF within the boundaries [96]. This method has moderate computational demands and generally leads to an improvement of the estimation accuracy [97]. Modifying the generic particle filter so that it discards particles that violate constraints, we adopt the acceptance-rejection scheme in [98],

$$w_k^{(i)} = L_k^{(i)}\big(\hat{\mathbf{x}}_k^{(i)}\big) w_{k-1}^{(i)} \frac{p\big(\mathbf{z}_k \,|\, \hat{\mathbf{x}}_k^{(i)}\big) p\big(\hat{\mathbf{x}}_k^{(i)} \,|\, \mathbf{x}_{k-1}^{(i)}, \mathbf{u}_{k-1}\big)}{\pi\big(\hat{\mathbf{x}}_k^{(i)} \,|\, \mathbf{X}_{k-1}^{(i)}, \mathbf{Z}_k, \mathbf{U}_{k-1}\big)}, \quad i \in \{1, \ldots, N\}, \tag{175}$$

with

$$L_k^{(i)}\big(\hat{\mathbf{x}}_k^{(i)}\big) = \begin{cases} 1 & \text{if } \hat{\mathbf{x}}_k^{(i)} \in \mathbb{S}_k \\ 0 & \text{otherwise} \end{cases} \tag{176}$$

Hence, the particles with zero weight will die out while resampling.

3.5 THE NEW LOCALISATION ALGORITHMS

We shall now describe the four newly proposed filter algorithms, all of which share the following features. The initial search space $[\mathbf{x}_0] \in [\mathbb{R}]^{n_x}$ encompasses the entire map. The sequence of the n_s control inputs and measurements is denoted by $\mathbf{U}_{n_s}$ and $\mathbf{Z}_{n_s}$, respectively. The probabilistic system model $\boldsymbol{\phi}_k : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ as well as the bounded-error system model $[\boldsymbol{\phi}_k] : [\mathbb{R}]^{n_x} \to [\mathbb{R}]^{n_x}$ and measurement model $[\mathbf{h}_k] : [\mathbb{R}]^{n_x} \to [\mathbb{R}]^{n_z}$ are assumed to be constant, therefore we can omit the index k.

3.5.1 *Initialisation*

A robot solving the wake-up robot problem, by definition, does not know where it is and is aware of this fact. Thus, in the beginning of conventional Monte Carlo localisation using a bootstrap filter or an unscented particle filter, hypotheses of the robot's position are spread over the entire map. However, when additional information that confines the initial search space is available it may be used to restrict the spreading of particles to regions that potentially contain the robot's true position and therefore increase the localisation accuracy.

A possible way of generating particles in the first iteration is to check whether a particle is inside a box or subpaving obtained by means of interval analysis and replacing it in case it is outside [27, 28, 99]. This may take a long time depending on the shape of the subpaving. A more efficient approach is uniformly generating particles inside the hull of a subpaving and killing the ones that are inconsistent with the constraints used for the bounded-error localisation by setting their importance weights to zero [91].

The bounded-error measurement model $[\mathbf{h}]$ can be used to construct a set $\mathbb{C}_k$ of n_z constraints that are used to narrow down the search space by means of a contractor or by means of `SIVIA`. Since the first measurement, per definition of the Bayesian filter, only comes available at time step $k = 1$, this measurement and its associated constraints are used to contract the search space or obtain a subpaving $\bar{\mathbb{S}}_1$ that approximates the solution set $\mathbb{S}_1$, respectively. Figure 27 shows the result of the `SIVIA` algorithm in three dimensions with respect to an exemplary trajectory and $\epsilon = 1.5\,\mathrm{m}$ as well as its interval hull. Then, the respective contracted box or interval hull of the subpaving,

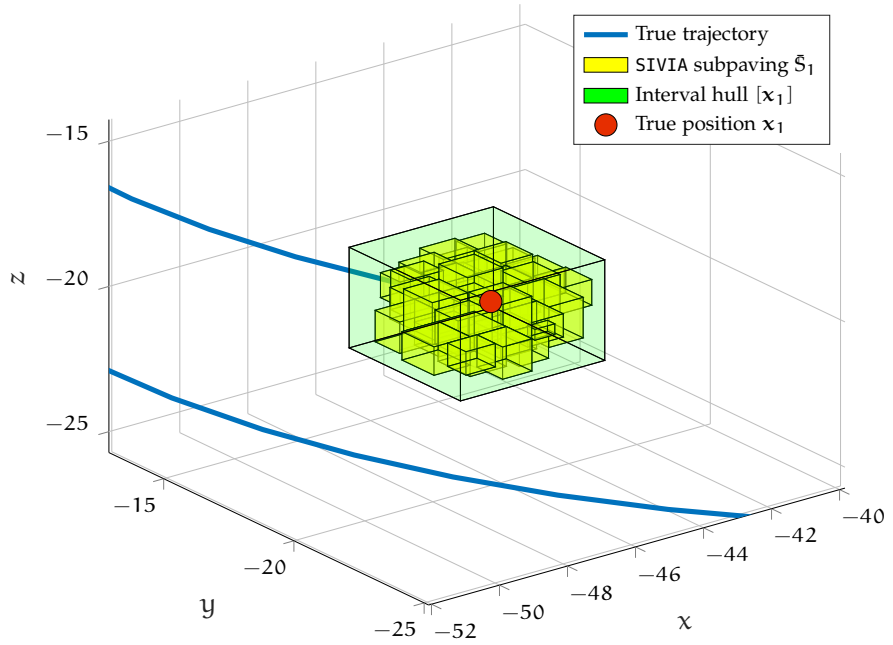


Figure 27: The result of the SIVIA algorithm in three dimensions for an exemplary trajectory and $\epsilon = 1.5\,\mathrm{m}$ depicted in yellow and its interval hull depicted in green.

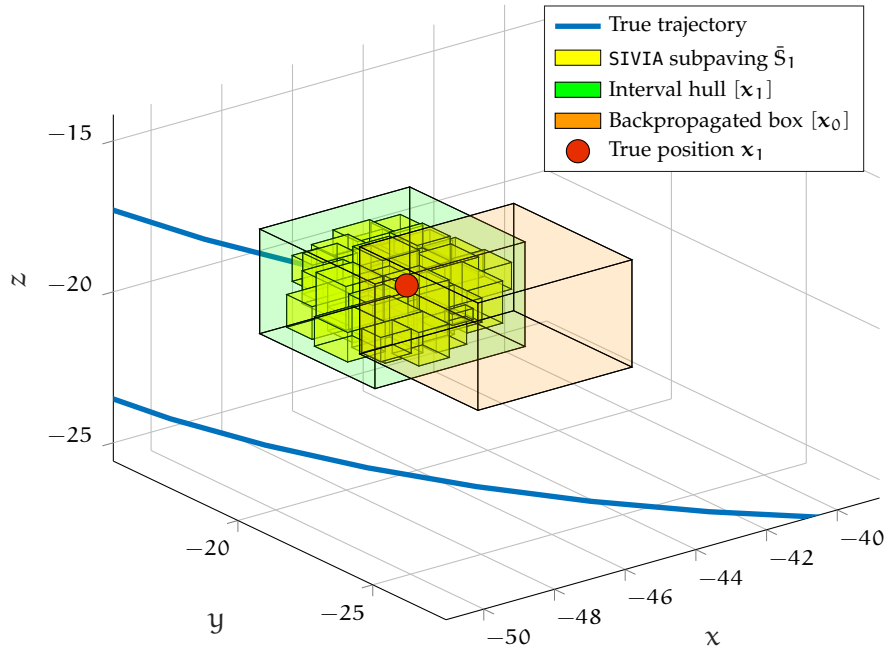


Figure 28: Backpropagated box $[\mathbf{x}_0]$ in relation to the interval hull $[\mathbf{x}_1]$ of the SIVIA subpaving $\bar{S}_1$.

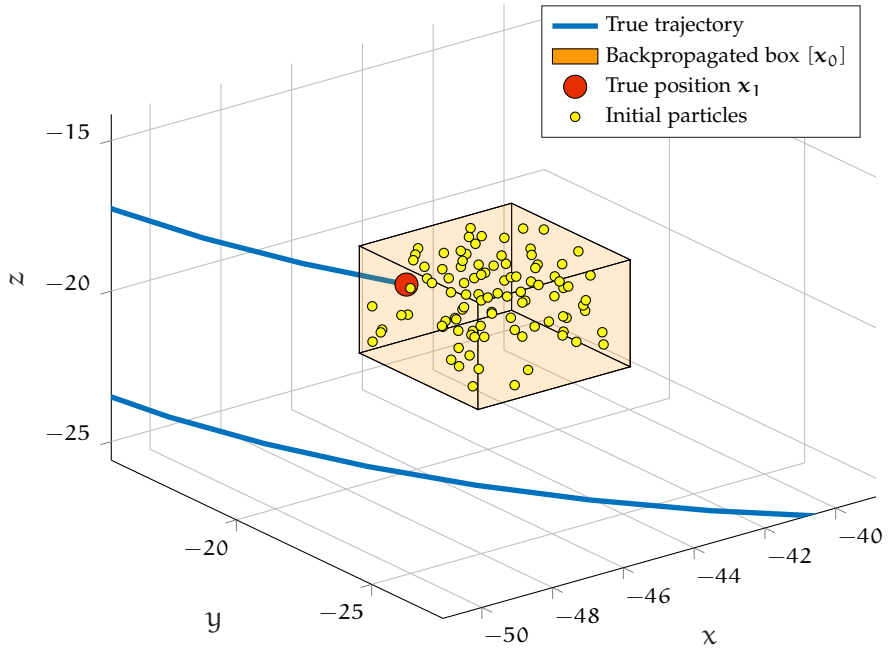


Figure 29: Uniformly spread initial particles in the backpropagated box $[\mathbf{x}_0]$.

both denoted by $[\mathbf{x}_1]$, are propagated backwards by one time step. Rearranging Equation 171, we have

$$\begin{aligned}[\mathbf{x}_{k-1}] = [\mathbf{x}_k] \\ - [\mathbf{T}_{bw}]\big([\psi_{k-1}] + [w_{\psi,k-1}], [\theta_{k-1}] + [w_{\theta,k-1}], [\phi_{k-1}] + [w_{\phi,k-1}]\big) \\ \cdot \big([\mathbf{u}_{k-1}^{[b]}] + [\boldsymbol{w}_{v,k-1}]\big)T_s, \quad k \in \{1, 2, 3, \dots\}.\end{aligned} \tag{177}$$

Plugging in $[\mathbf{x}_1]$ and the zeroth control input, that is $[\mathbf{u}_{k-1}^{[b]}] = \mathbf{u}_0^{[b]}$, results in the box $[\mathbf{x}_0]$, which is depicted in Figure 28. This box confines the uniform spreading of equally weighted particles drawn from $p(\mathbf{x}_0)$, which constitute the set $\left\{\left(\mathbf{x}_0^{(i)}, N^{-1}\right)\right\}_{i=1}^{N}$ as depicted in Figure 29. Given this initial distribution, in principle, any standard Bayesian filter algorithm could be applied.

3.5.2 *Detection of Kidnapping*

Previous hybrid methods based on Monte Carlo simulation and interval analysis maintain the respective contracted box [27] or subpaving [28] throughout the whole estimation process. Each particle that does not satisfy the constraints is replaced by a particle sampled from a uniform distribution that is confined by the respective bounded-error estimate or its interval hull. However, instead of generating new particles at every time step, the ones that are inconsistent with the constraints can be killed and the remaining ones can be reproduced

Algorithm 2 PFC (bootstrap particle filter with HC4 contractor)

Input: $[\mathbf{x}_0]$, $\mathbf{U}_{n_s}$, $\mathbf{Y}_{n_s}$, $\boldsymbol{\phi}$, $[\boldsymbol{\phi}]$, $[\mathbf{h}]$, $\xi > 0$
Output: $\{\hat{\mathbf{x}}_k\}$, $\{\mathbf{P}_k\}$, $k \in \{1, \ldots, n\}$

1: **function** PFC($[\mathbf{x}_0], \mathbf{U}_{n_s}, \mathbf{Y}_{n_s}, \boldsymbol{\phi}, [\boldsymbol{\phi}], [\mathbf{h}], \xi$)
2: **for** $k \leftarrow 1, n_s$ **do**
3: $\mathbb{C}_k \leftarrow$ CONSTRUCTCONSTRAINTS($[\mathbf{h}], z_k, \xi$) ▷ Eq. 172
4: **if** $k = 1$ **then** ▷ initialise filter
5: $[\mathbf{x}_k] \leftarrow$ CONTRACT($[\mathbf{x}_0], \mathbb{C}_k$) ▷ contract using HC4
6: $[\mathbf{x}_{k-1}] \leftarrow$ PROPAGATEBACKWARDS($[\mathbf{x}_k]$) ▷ Eq. 177
7: $\mathbb{P}_{k-1} \leftarrow$ DRAWPARTICLES($N, [\mathbf{x}_{k-1}]$) ▷ Eq. 84
8: **end if**
9: $\mathbb{P}_k \leftarrow$ PROPAGATEPARTICLES($\mathbb{P}_{k-1}$) ▷ Eq. 92
10: $\mathbb{W}_k \leftarrow$ WEIGHTPARTICLES($\mathbb{P}_k, \mathbb{C}$) ▷ Eq. 175
11: **if** $\sum_{W \in \mathbb{W}_k} W = 0$ **then** ▷ restart localisation process
12: **go to** 6
13: **end if**
14: $\mathbb{P}_k \leftarrow$ RESAMPLE($\mathbb{P}_k, \mathbb{W}_k$) ▷ Eq. 80
15: $\hat{\mathbf{x}}_k \leftarrow$ MEAN($\mathbb{P}_k$) ▷ Eq. 89
16: $\mathbf{P}_k \leftarrow$ COVARIANCE($\hat{\mathbf{x}}_k, \mathbb{P}_k$) ▷ Eq. 90
17: **end for**
18: **return** $\left(\{\hat{\mathbf{x}}_k\}, \{\mathbf{P}_k\}\right)$
19: **end function**

while resampling. When all particles have zero weight, that is none of them satisfy all constraints, there has been a localisation failure. In other words, the robot has been kidnapped. Then, for the robot a reasonable action is to perform global localisation over the whole map. Therefore, the same steps as during the initialisation described in the previous section are carried out but using the current measurement instead of the first one. Carrying out a contraction of the search space or generating a SIVIA subpaving only in the very first iteration and in the iteration just after kidnapping, respectively, immensely reduces computational cost while preserving the benefits of bounded-error state estimation.

3.5.3 *Particle Filters with HC4 Contractor*

Algorithm 2 depicts the particle filter with contractor (PFC). Initially, after the construction of the constraints, the search space is contracted using the HC4 algorithm. The resulting box is propagated backwards by one time step and particles are spread uniformly within the box. These particles are propagated by the system model and weighted. Given there are non-zero weights, resampling is carried out and the empirical particle mean and covariance is returned. Otherwise, when

Algorithm 3 UPFC (unscented particle filter with HC4 contractor)

Input: $[\mathbf{x}_0]$, $\mathbf{U}_{n_s}$, $\mathbf{Y}_{n_s}$, $\boldsymbol{\phi}$, $[\boldsymbol{\phi}]$, $[\mathbf{h}]$, $\xi > 0$
Output: $\{\hat{\mathbf{x}}_k\}$, $\{\mathbf{P}_k\}$, $k \in \{1, \ldots, n\}$

1: **function** UPFC($[\mathbf{x}_0], \mathbf{U}_{n_s}, \mathbf{Y}_{n_s}, \boldsymbol{\phi}, [\boldsymbol{\phi}], [\mathbf{h}], \xi$)
2: **for** $k \leftarrow 1, n_s$ **do**
3: $\mathbb{C}_k \leftarrow$ CONSTRUCTCONSTRAINTS($[\mathbf{h}], \mathbf{z}_k, \xi$) ▷ Eq. 172
4: **if** $k = 1$ **then** ▷ initialise filter
5: $[\mathbf{x}_k] \leftarrow$ CONTRACT($[\mathbf{x}_0], \mathbb{C}_k$) ▷ contract using HC4
6: $[\mathbf{x}_{k-1}] \leftarrow$ PROPAGATEBACKWARDS($[\mathbf{x}_k]$) ▷ Eq. 177
7: $\left(\mathbb{P}_{k-1}, \left\{\mathbf{P}_{k-1}^{(i)}\right\}\right) \leftarrow$ DRAWPARTICLES($N, [\mathbf{x}_{k-1}]$) ▷ Eq. 84
8: **end if**
9: $\left(\overline{\mathbb{P}}_k, \left\{\mathbf{P}_k^{(i)}\right\}\right) \leftarrow$ UKF($\mathbb{P}_{k-1}, \left\{\mathbf{P}_{k-1}^{(i)}\right\}$) ▷ Algorithm 11
10: $\mathbb{P}_k \leftarrow$ SAMPLE($\overline{\mathbb{P}}_k, \left\{\mathbf{P}_k^{(i)}\right\}$) ▷ Eq. 99
11: $\mathbb{W}_k \leftarrow$ WEIGHTPARTICLES($\mathbb{P}_k, \mathbb{C}$) ▷ Eq. 175
12: **if** $\sum_{W \in \mathbb{W}_k} W = 0$ **then** ▷ restart localisation process
13: **go to** 6
14: **end if**
15: $\mathbb{P}_k \leftarrow$ RESAMPLE($\mathbb{P}_k, \mathbb{W}_k$) ▷ Eq. 80
16: $\hat{\mathbf{x}}_k \leftarrow$ MEAN($\mathbb{P}_k$) ▷ Eq. 89
17: $\mathbf{P}_k \leftarrow$ COVARIANCE($\hat{\mathbf{x}}_k, \mathbb{P}_k$) ▷ Eq. 90
18: **end for**
19: **return** $\left(\{\hat{\mathbf{x}}_k\}, \{\mathbf{P}_k\}\right)$
20: **end function**

all weights are zero, the localisation process is restarted by going to line 5. The filter loop is repeated for each of the n_s samples.

Algorithm 3 depicts the unscented particle filter with contractor (UPFC). Initially, after the construction of the constraints, the search space is contracted using the HC4 algorithm. The resulting box is propagated backwards by one time step and particles are spread uniformly within the box. The respective particle mean and covariance is propagated by an unscented Kalman filter for each particle, taking into account the latest measurement. Then, N particles are drawn from a Gaussian proposal distribution with the respective mean and covariance, and weighted using the measurement model. Given there are non-zero weights, resampling is carried out and the empirical particle mean and covariance is returned. Otherwise, when all weights are zero, the localisation process is restarted by going to line 5. The filter loop is repeated for each of the n_s samples.

3.5.4 *Particle Filters with SIVIA*

Algorithm 4 depicts the particle filter with SIVIA (UPFS). Initially, after the construction of the constraints, a subpaving of the solution set is

Algorithm 4 PFS (boostrap particle filter with SIVIA)

Input: $[\mathbf{x}_0]$, $\mathbf{U}_{n_s}$, $\mathbf{Y}_{n_s}$, $\boldsymbol{\phi}$, $[\boldsymbol{\phi}]$, $[\mathbf{h}]$, $\xi > 0$, $\epsilon > 0$
Output: $\{\hat{\mathbf{x}}_k\}$, $\{\mathbf{P}_k\}$, $k \in \{1, \ldots, n\}$

1: **function** PFS($[\mathbf{x}_0], \mathbf{U}_{n_s}, \mathbf{Y}_{n_s}, \boldsymbol{\phi}, [\boldsymbol{\phi}], [\mathbf{h}], \xi, \epsilon$)
2: **for** $k \leftarrow 1, n_s$ **do**
3: $\mathbb{C}_k \leftarrow$ CONSTRUCTCONSTRAINTS($[\mathbf{h}], z_k, \xi$) ▷ Eq. 172
4: **if** $k = 1$ **then** ▷ initialise filter
5: $\mathbb{X}_k \leftarrow$ SIVIA($[\mathbf{x}_0], \mathbb{C}_k, \epsilon$) ▷ Algorithm 1
6: $[\mathbf{x}_k] \leftarrow$ HULL($\mathbb{X}_k$)
7: $[\mathbf{x}_{k-1}] \leftarrow$ PROPAGATEBACKWARDS($[\mathbf{x}_k]$) ▷ Eq. 177
8: $\mathbb{P}_{k-1} \leftarrow$ DRAWPARTICLES($N, [\mathbf{x}_{k-1}]$) ▷ Eq. 84
9: **end if**
10: $\mathbb{P}_k \leftarrow$ PROPAGATEPARTICLES($\mathbb{P}_{k-1}$) ▷ Eq. 92
11: $\mathbb{W}_k \leftarrow$ WEIGHTPARTICLES($\mathbb{P}_k, \mathbb{C}$) ▷ Eq. 175
12: **if** $\sum_{W \in \mathbb{W}_k} W = 0$ **then** ▷ restart localisation process
13: **go to** 6
14: **end if**
15: $\mathbb{P}_k \leftarrow$ RESAMPLE($\mathbb{P}_k, \mathbb{W}_k$) ▷ Eq. 80
16: $\hat{\mathbf{x}}_k \leftarrow$ MEAN($\mathbb{P}_k$) ▷ Eq. 89
17: $\mathbf{P}_k \leftarrow$ COVARIANCE($\hat{\mathbf{x}}_k, \mathbb{P}_k$) ▷ Eq. 90
18: **end for**
19: **return** $\left(\{\hat{\mathbf{x}}_k\}, \{\mathbf{P}_k\}\right)$
20: **end function**

computed using the SIVIA algorithm. The hull of the subpaving is propagated backwards by one time step and particles are spread uniformly within the box. These particles are propagated by the system model and weighted. Given there are non-zero weights, resampling is carried out and the empirical particle mean and covariance is returned. Otherwise, when all weights are zero, the localisation process is restarted by going to line 5. The filter loop is repeated for each of the n_s samples.

Algorithm 5 depicts the unscented particle filter with SIVIA (UPFS). Initially, after the construction of the constraints, a subpaving of the solution set is computed using the SIVIA algorithm. The hull of the subpaving is propagated backwards by one time step and particles are spread uniformly within the box. The respective particle mean and covariance is propagated by an unscented Kalman filter for each particle, taking into account the latest measurement. Then, N particles are drawn from a Gaussian proposal distribution with the respective mean and covariance, and weighted using the measurement model. Given there are non-zero weights, resampling is carried out and the empirical particle mean and covariance is returned. Otherwise, when all weights are zero, the localisation process is restarted by going to line 5. The filter loop is repeated for each if the n_s samples.

Algorithm 5 UPFS (unscented particle filter with SIVIA)

Input: $[\mathbf{x}_0]$, $\mathbf{U}_{n_s}$, $\mathbf{Y}_{n_s}$, $\boldsymbol{\phi}$, $[\boldsymbol{\phi}]$, $[\mathbf{h}]$, $\xi > 0$, $\epsilon > 0$
Output: $\{\hat{\mathbf{x}}_k\}$, $\{\mathbf{P}_k\}$, $k \in \{1, \ldots, n\}$

1: **function** UPFS($[\mathbf{x}_0], \mathbf{U}_{n_s}, \mathbf{Y}_{n_s}, \boldsymbol{\phi}, [\boldsymbol{\phi}], [\mathbf{h}], \xi, \epsilon$)
2: **for** $k \leftarrow 1, n_s$ **do**
3: $\mathbb{C}_k \leftarrow$ CONSTRUCTCONSTRAINTS($[\mathbf{h}], \mathbf{z}_k, \xi$) ▷ Eq. 172
4: **if** $k = 1$ **then** ▷ initialise filter
5: $\mathbb{X}_k \leftarrow$ SIVIA($[\mathbf{x}_0], \mathbb{C}_k, \epsilon$) ▷ Algorithm 1
6: $[\mathbf{x}_k] \leftarrow$ HULL($\mathbb{X}_k$)
7: $[\mathbf{x}_{k-1}] \leftarrow$ PROPAGATEBACKWARDS($[\mathbf{x}_k]$) ▷ Eq. 177
8: $\left(\mathbb{P}_{k-1}, \left\{\mathbf{P}_{k-1}^{(i)}\right\}\right) \leftarrow$ DRAWPARTICLES($N, [\mathbf{x}_{k-1}]$) ▷ Eq. 84
9: **end if**
10: $\left(\overline{\mathbb{P}}_k, \left\{\mathbf{P}_k^{(i)}\right\}\right) \leftarrow$ UKF($\mathbb{P}_{k-1}, \left\{\mathbf{P}_{k-1}^{(i)}\right\}$) ▷ Algorithm 11
11: $\mathbb{P}_k \leftarrow$ SAMPLE($\overline{\mathbb{P}}_k, \left\{\mathbf{P}_k^{(i)}\right\}$) ▷ Eq. 99
12: $\mathbb{W}_k \leftarrow$ WEIGHTPARTICLES($\mathbb{P}_k, \mathbb{C}$) ▷ Eq. 175
13: **if** $\sum_{W \in \mathbb{W}_k} W = 0$ **then** ▷ restart localisation process
14: **go to** 6
15: **end if**
16: $\mathbb{P}_k \leftarrow$ RESAMPLE($\mathbb{P}_k, \mathbb{W}_k$) ▷ Eq. 80
17: $\hat{\mathbf{x}}_k \leftarrow$ MEAN($\mathbb{P}_k$) ▷ Eq. 89
18: $\mathbf{P}_k \leftarrow$ COVARIANCE($\hat{\mathbf{x}}_k, \mathbb{P}_k$) ▷ Eq. 90
19: **end for**
20: **return** $\left(\{\hat{\mathbf{x}}_k\}, \{\mathbf{P}_k\}\right)$
21: **end function**

4

EXPERIMENTS

In order to evaluate the performance of the four newly proposed algorithms in Section 3.5, they were applied to simulated data and the results were compared to those of the two conventional probabilistic methods, namely the bootstrap filter as described in Section 2.5.2 and the unscented particle filter as described in Section 2.5.3. The implementation and visualisation of the probabilistic and bounded-error state estimation are described in the following section. Subsequently, the experimental setup including the robot simulator and the simulated scenarios will be described in detail in Section 4.2, followed by the performance measures used to validate the algorithms in Section 4.3. Section 4.4 presents the estimation results, which are discussed in Section 4.5.

4.1 IMPLEMENTATION

All six algorithms were implemented in MATLAB® using the INTLAB® Toolbox [100] for interval computations. The C++ `SIVIA` implementation with a MATLAB® MEX [101] interface was borrowed from [82]. The entire developed code and the data used for the experiments can be found on Github [102], a web-based hosting service for distributed version control using Git. Both code and data are accessible freely for repetition and comparison as well as for all other scientific purposes. Each of the results presented below was obtained running a distinguished MATLAB® script, which includes the parameterisation of the respective experiment. The script is contained with the respective data and results in a subfolder for each experiment. The entire estimation process including bounded-error localisation, sampling, weighting, and resampling as described in Section 3.5 was visualised for verification purposes and tuning of the filters. The Figures 27, 28 and 29 show screenshots of the real-time visualisation of the estimation process.

4.2 EXPERIMENTAL SETUP

In the following two sections, we will describe the experimental setup, including the robot simulator used for the experiments and the three simulated scenarios.

Signal	Symbol	Standard deviation
Distances	σ_d	0.30 m
Linear velocity	σ_v	0.04 $^{m}/_{s}$
Euler angles	σ_{Eul}	0.10 °

Table 1: Standard deviations of the simulated data.

4.2.1 *Robot Simulator*

The raw data of the simulation was generated by Nicola [103], using the Modular OpenRobots Simulation Engine (MORSE) [104], an academic robotic simulator based on the Blender Game Engine [87] and the Bullet Physics Engine [105]. The simulated autonomous underwater vehicle *Redermor* [106] was equipped with sensors to measure its distance to each of $n_z \in \{2, 4, 9\}$ landmarks. The simulated vessel is depicted in Figure 55 in the appendix. The simulator logged the robot's nominal linear velocity in its three-dimensional body frame as well as its orientation with respect to the world frame at a rate of one sample per second. Both the latter quantities served as a control input for the system model as described in Section 3.2.1. Moreover, the true position of the robot was logged and served as a reference in the validation of the state estimation. The input and measurement data were distorted by adding zero-mean white Gaussian noise samples drawn from distributions with standard deviations as depicted in Table 1.

4.2.2 *Simulated Scenarios*

The underwater robot started its manoeuvre several metres under the surface of the sea, sank down to the seabed following a helical movement and went back up again with an opposite sense of rotation. Figure 30 shows the entire trajectory, which has a duration of 200 seconds, where the upper left of the trajectory represents the start point and the upper right represents the end of the simulation. The trajectory in Figure 31 shows the location where the robot was kidnapped. After it has sunken down finishing one whole arc of a circle, when projecting the trajectory on the x-y-plane, the robot is kidnapped and brought to the opposite side of the original trajectory, virtually skipping the yellow dashed part of it, as depicted in Figure 31. The kidnapping after 65 seconds alters the robot's position and orientation before it makes its way up in the same manner as in the scenario without kidnapping and finishes its manoeuvre after 135 seconds in the upper right corner.

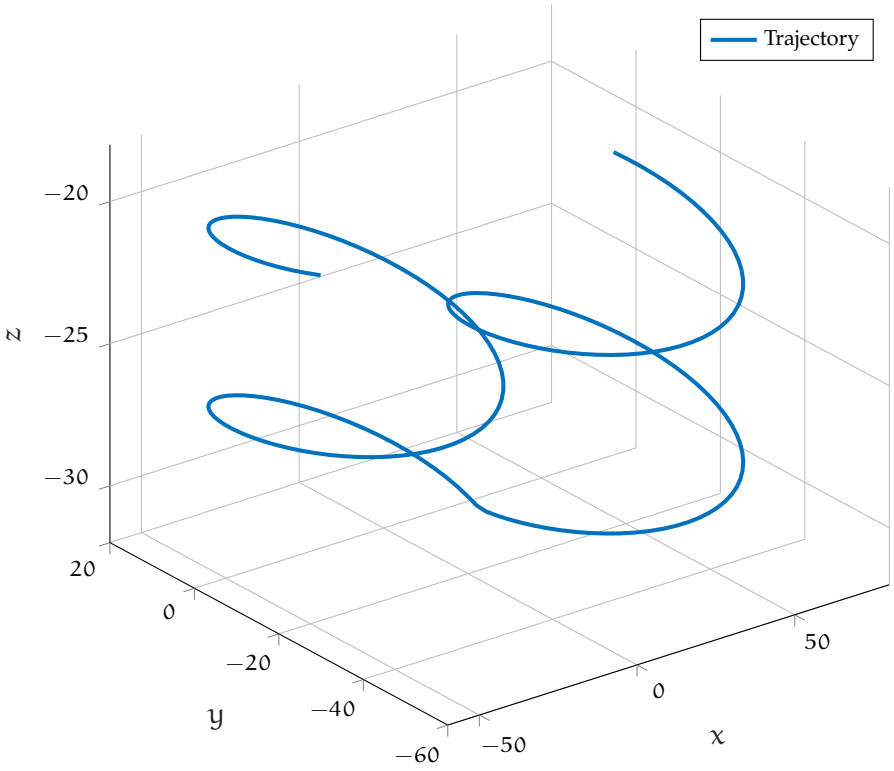


Figure 30: Simulated three-dimensional trajectory with the start point in the upper left and the end point in the upper right hand corner.

Landmark	x in m	y in m	z in m
1	54.212	−234.051	−225.619
2	72.673	225.574	−262.043

Table 2: Positions of the landmarks in Scenario 1.

The simulations were carried out with two, four, and nine distinguishable landmarks, respectively, in order to test the performance of the filters with ambiguous measurement data. In each of the three scenarios, the landmarks were placed in a grid above the seafloor, located about 250 metres deep. The individual position of each landmark was chosen randomly with a standard deviation of 10 metres. Figures 32, 33, and 34 show the trajectory in relation with the two, four, and nine landmarks, respectively. The positions of the landmarks are given in Tables 2, 3, and 4 for the respective scenario.

4.3 PERFORMANCE MEASURES

The performance of the algorithms was measured computing the error e_k at time step k as the Euclidean distance between the estimated pos-

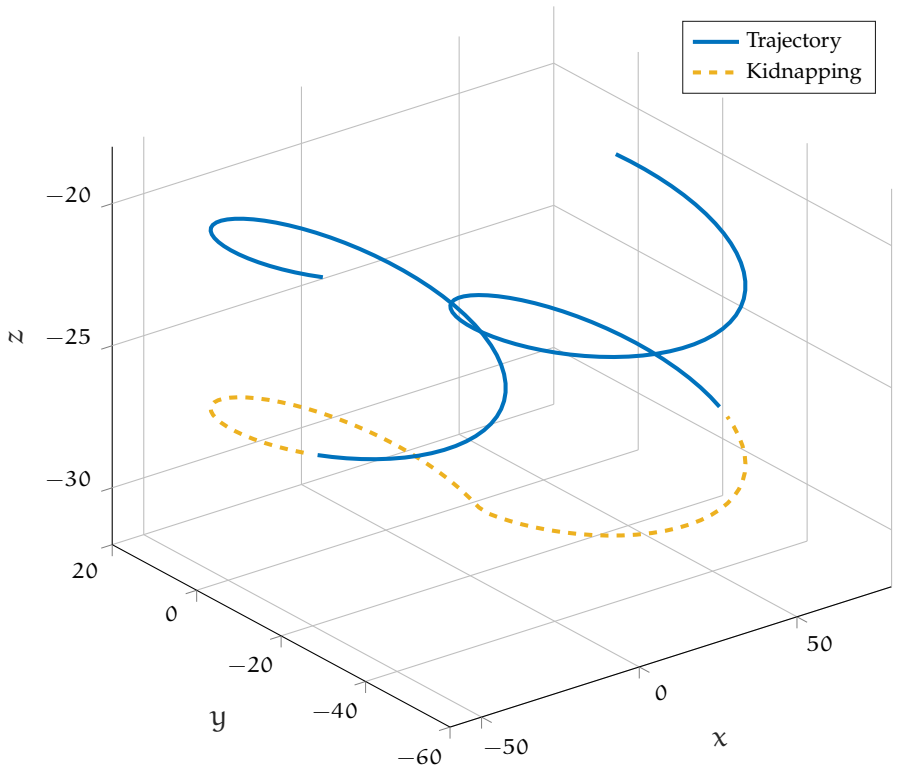


Figure 31: Simulated three-dimensional trajectory with the start point in the upper left and the end point in the upper right hand corner. The part of the trajectory that is skipped through kidnapping is depicted as a yellow dashed line.

Landmark	x in m	y in m	z in m
1	−236.365	−226.664	−233.550
2	231.643	−242.535	−257.183
3	−255.960	226.613	−231.048
4	247.786	209.724	−289.594

Table 3: Positions of the landmarks in Scenario 2.

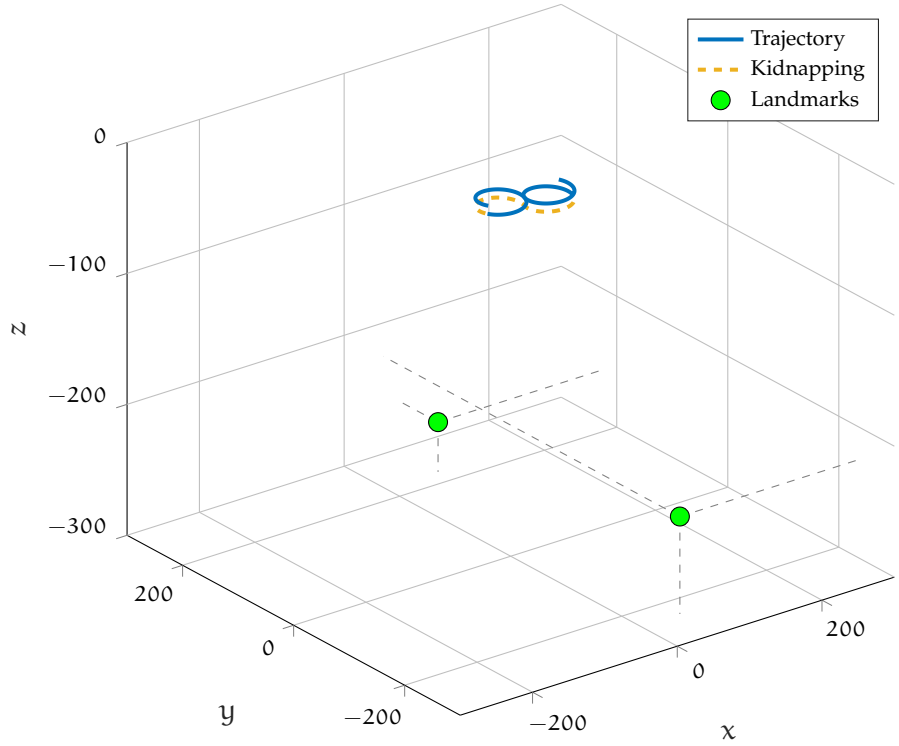


Figure 32: Three-dimensional trajectory in relation with the two landmarks spread over the seabed in Scenario 1.

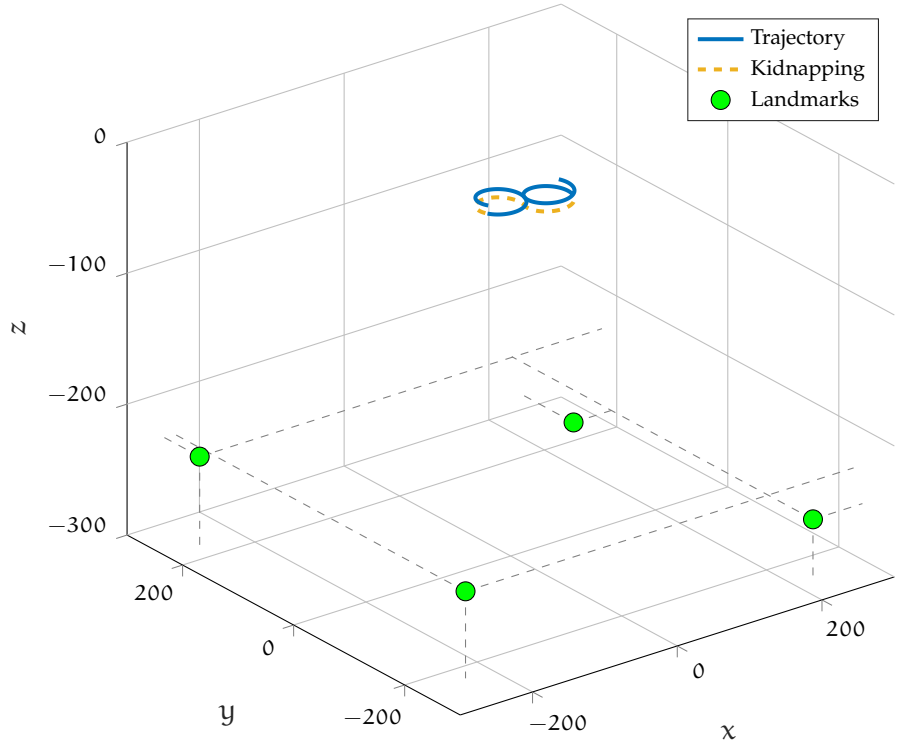


Figure 33: Three-dimensional trajectory in relation with the four landmarks spread over the seabed in Scenario 2.

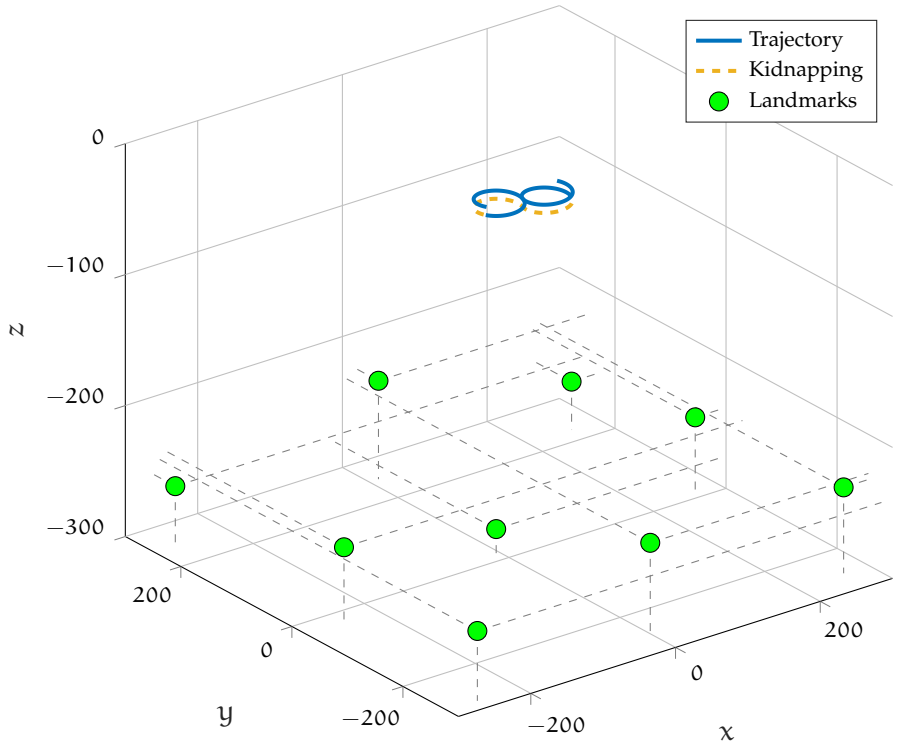


Figure 34: Three-dimensional trajectory in relation with the nine landmarks spread over the seabed in Scenario 3.

Landmark	x in m	y in m	z in m
1	−258.443	−279.902	−246.641
2	−1.464	−256.392	−229.858
3	265.583	−256.648	−234.381
4	−250.721	−29.613	−241.389
5	−10.034	9.951	−278.862
6	259.375	2.562	−239.162
7	−267.623	252.185	−256.285
8	12.539	251.370	−224.841
9	259.698	225.658	−263.047

Table 4: Positions of the landmarks in Scenario 3.

ition $\hat{\mathbf{x}}_k = [\hat{x}_k, \hat{y}_k, \hat{z}_k]$ and the true position $\mathbf{x}_k = [x_k, y_k, z_k]$ computed by the simulator,

$$e_k = \sqrt{(\hat{x}_k - x_k)^2 + (\hat{y}_k - y_k)^2 + (\hat{z} - z_k)^2}\,. \tag{178}$$

Given the entire sequence of estimated states $\hat{\mathbf{X}}_{n_s} = \{\hat{\mathbf{x}}_k\}_{k=1}^{n_s}$, the root-mean-square error RMSE $\left(\hat{\mathbf{X}}_{n_s}\right)$ is given by

$$\text{RMSE}\left(\hat{\mathbf{X}}_{n_s}\right) = \sqrt{\frac{1}{n_s}\sum_{k=1}^{n_s} e_k^2}\,. \tag{179}$$

In addition to the RMSE, the initial error e_1 was assessed. The experiments were carried out with 100000 particles for the `PF`, `PFC`, and `PFS` and 1000 particles for the `UPF`, `UPFC`, `UPFS`. In order to obtain representative results, each of the six filters was applied to the respective estimation task at hand in a hundred runs.

4.4 RESULTS

In the following, the results of the experiments described above are presented. For each of the three scenarios there are six figures, three for the respective scenario with kidnapping, and three without kidnapping. Each of the groups of three figures consists of a box plot of the estimation error distribution, a plot of the mean error sequence vs. time, and a magnified plot of the mean error sequence vs. time depicting the interesting interval with a width of 40 seconds. That is, the first 40 seconds when solving the wake-up robot problem and the 10 seconds before plus the 30 seconds after kidnapping, when solving the kidnapped robot problem. The same set of figures is presented in the appendix for 10 times fewer particles, respectively.

On each box of a box plot, the central mark indicates the median and the bottom and top edges of the box indicate the 25th and 75th percentiles, respectively. The whiskers extend to the most extreme data points not considered outliers, and the outliers are plotted individually using a red '+' symbol. Points are drawn as outliers if they are greater than $Q_{25} + l(Q_{75}{-}Q_{25})$ or less than $Q_{25} - l(Q_{75}{-}Q_{25})$, where l is the maximum whisker length corresponding to approximately $\pm 2.7\sigma$ and 99.3 percent coverage if the data are normally distributed.

Plots of the individual runs including the estimated trajectory in relation with the reference trajectory, the number of particles that received an importance weight of zero, and `.mat`-files containing the estimation results, with and without kidnapping, respectively, can be found on GitHub [102]. In addition to that, all exact results of the individual runs and of the entire experiments, including the time of convergence and the root-mean-squared errors, can be found in the log files provided.

4.4.1 *Results with 2 Landmarks*

Figures 35 - 37 depicting the estimation results for Scenario 1.

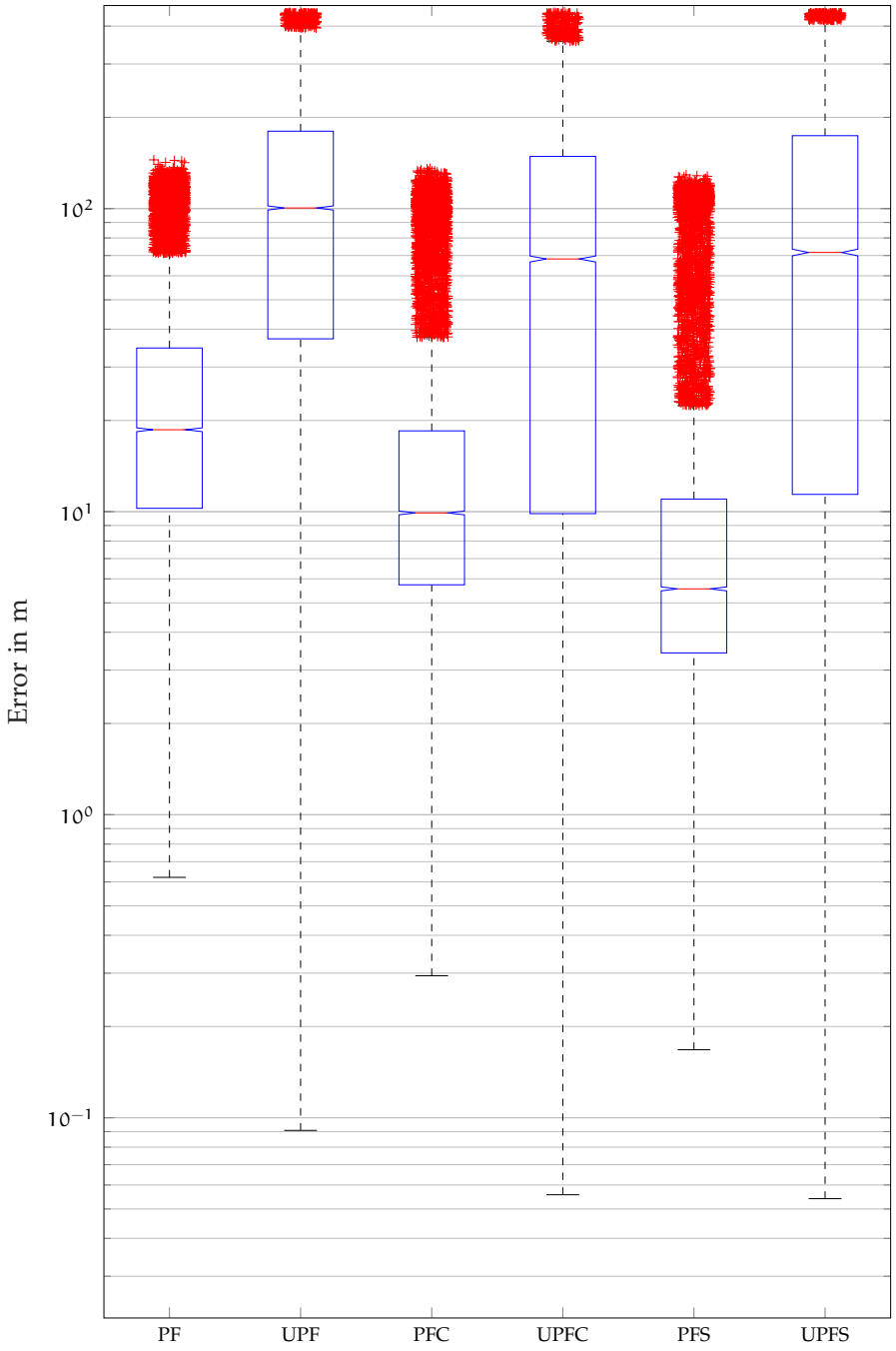


Figure 35: Box plot of the estimation errors in Scenario 1 (logarithmic scale). `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 1000 particles.

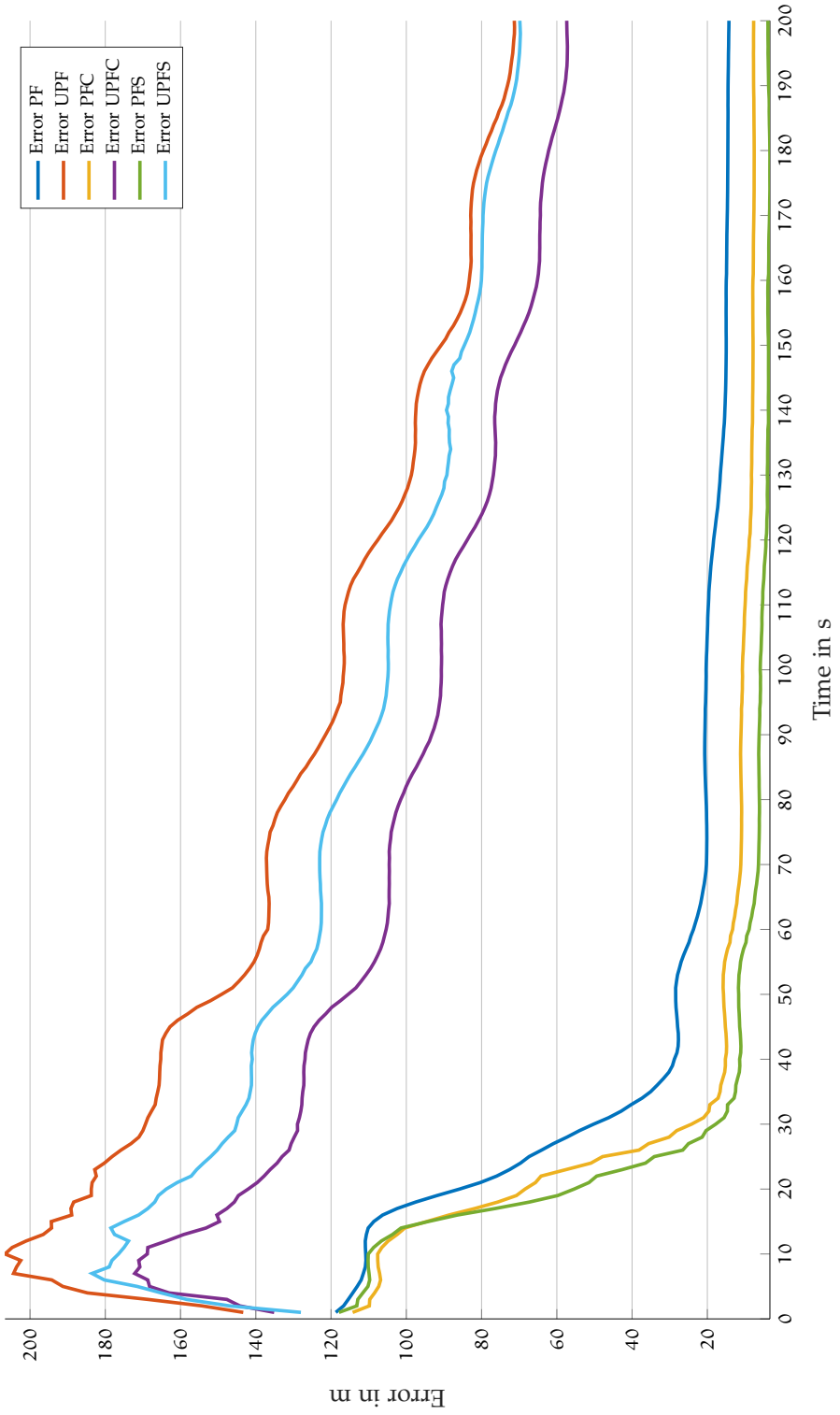


Figure 36: Mean estimation error over time in Scenario 1. `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 1000 particles.

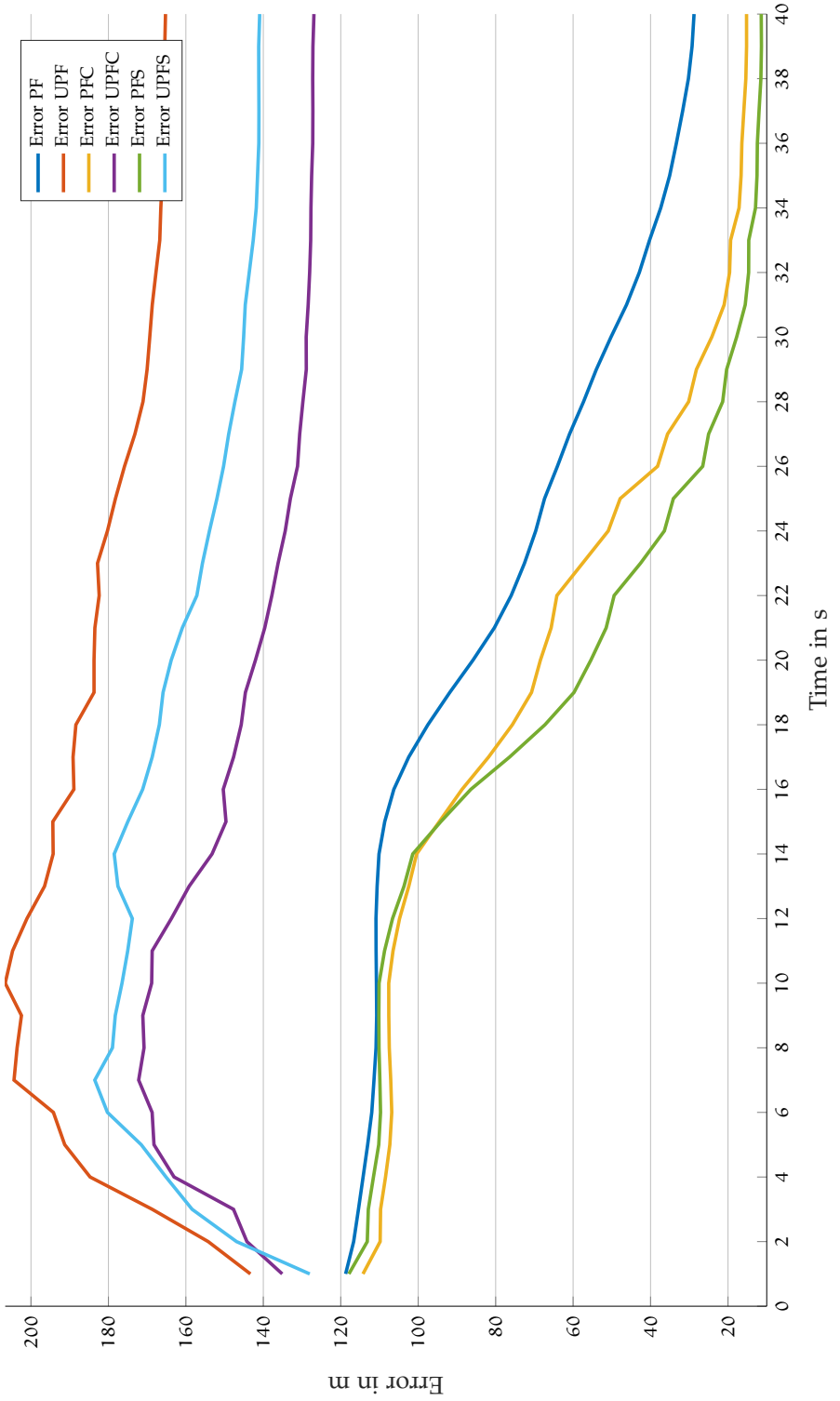


Figure 37: First 40 seconds of the mean estimation error over time in Scenario 1. `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 1000 particles.

4.4.2 *Results with 2 Landmarks and Kidnapping*

Figures 38 - 40 depicting the estimation results for Scenario 1 with kidnapping.

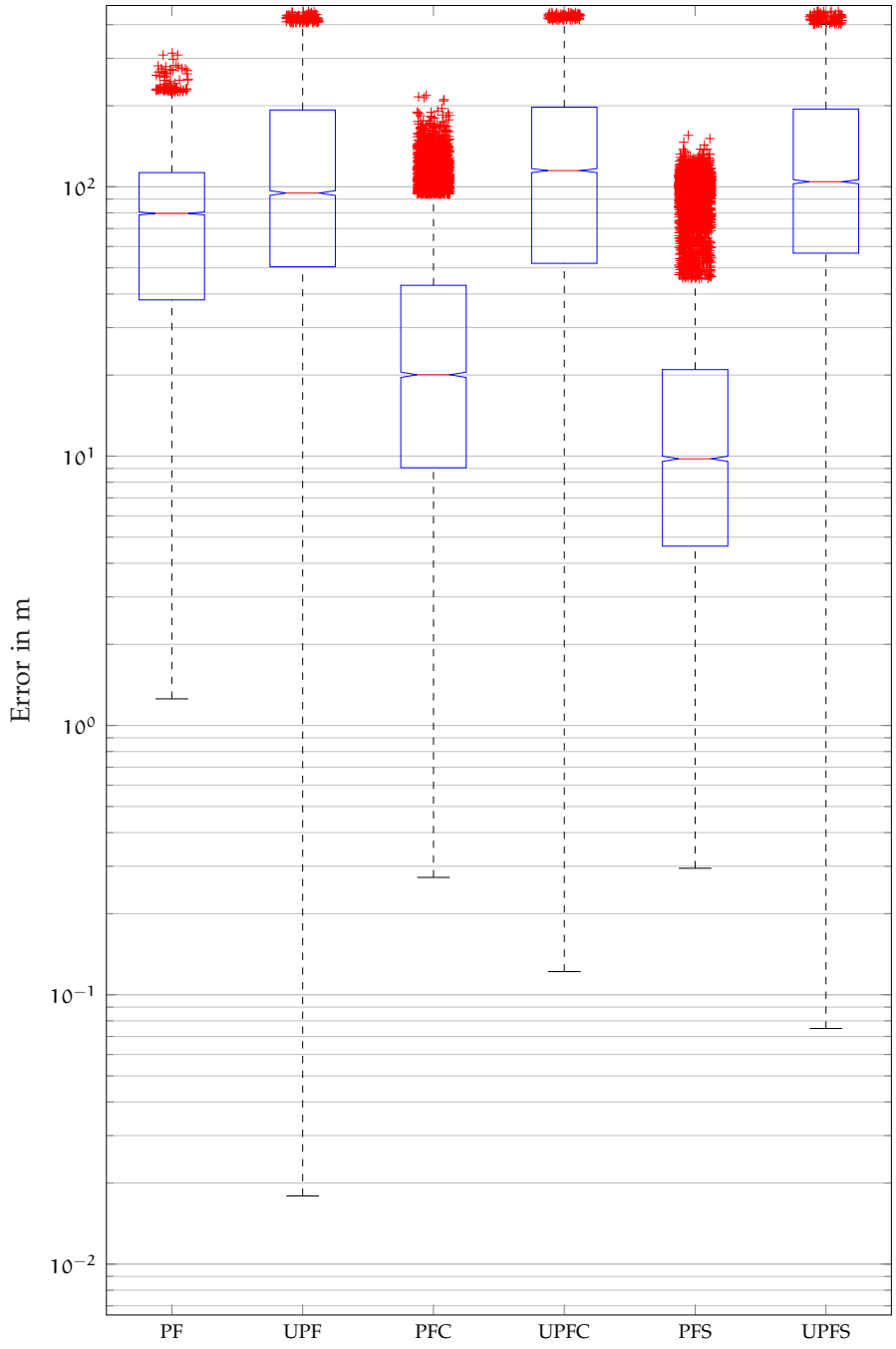


Figure 38: Box plot of the estimation errors in Scenario 1 with kidnapping (logarithmic scale). `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 1000 particles.

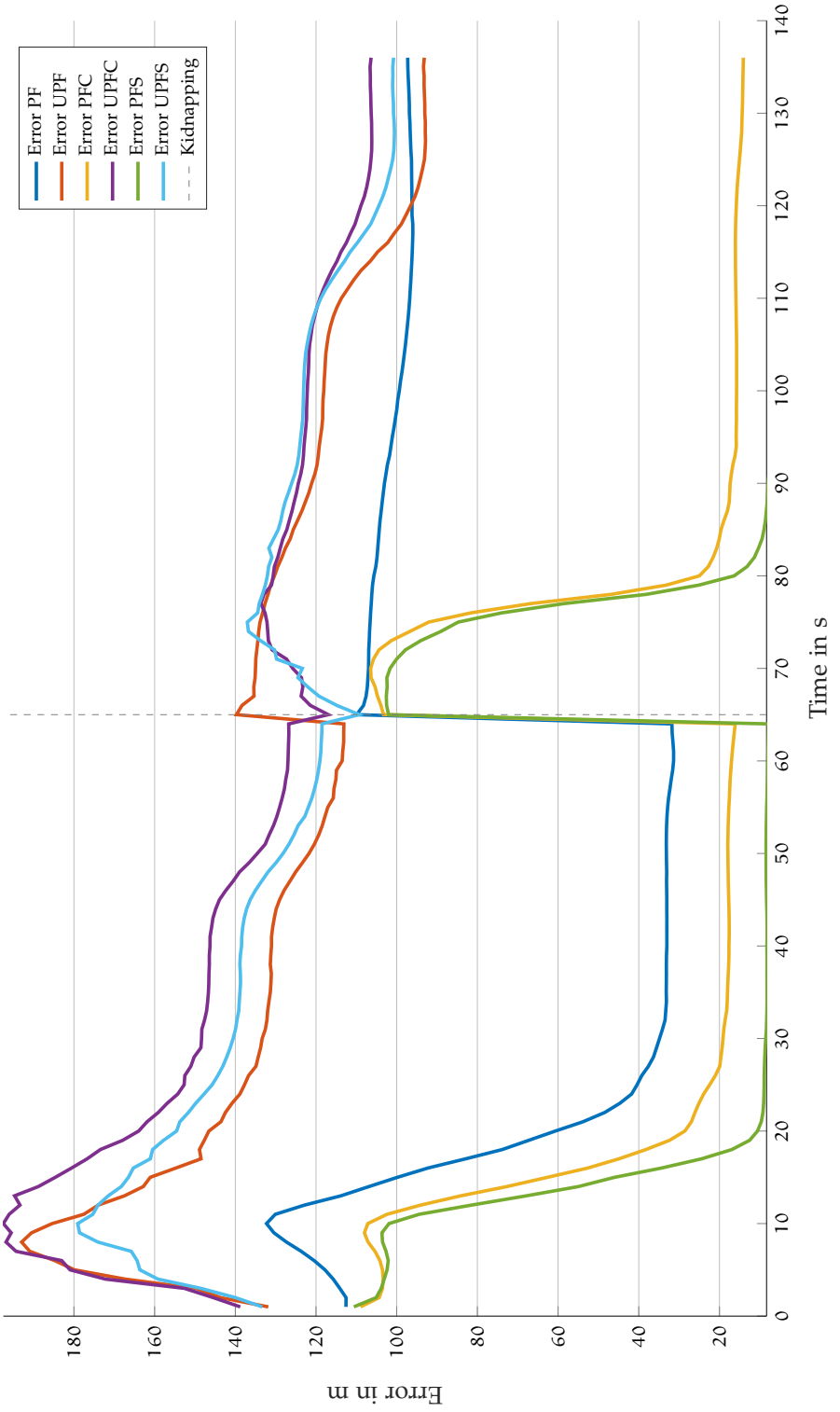


Figure 39: Mean estimation error over time in Scenario 1 with kidnapping. `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 1000 particles.

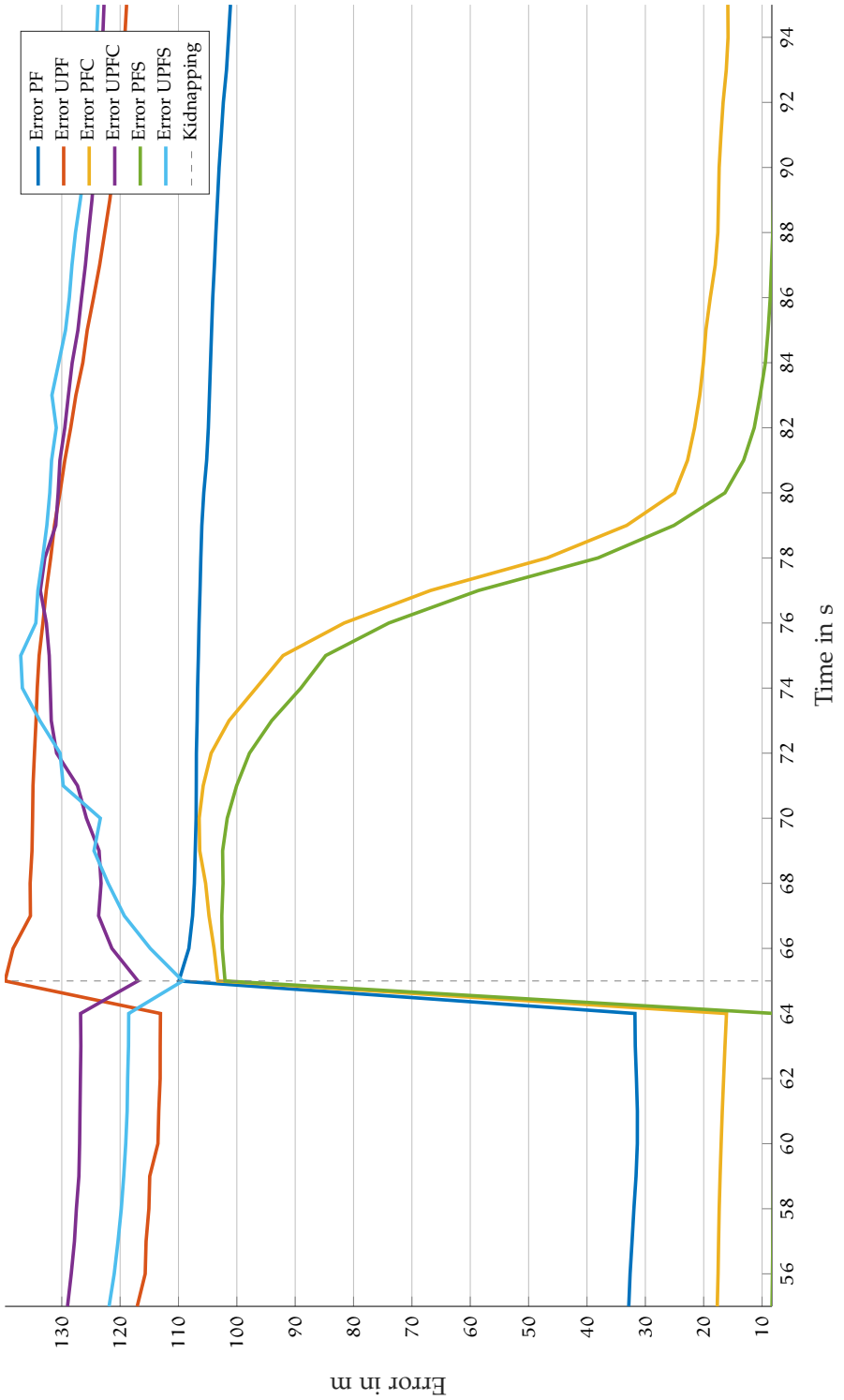


Figure 40: First 30 seconds of the mean estimation error over time after kidnapping in Scenario 1. `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 1000 particles.

4.4.3 *Results with 4 Landmarks*

Figures 41 - 43 depicting the estimation results for Scenario 2.

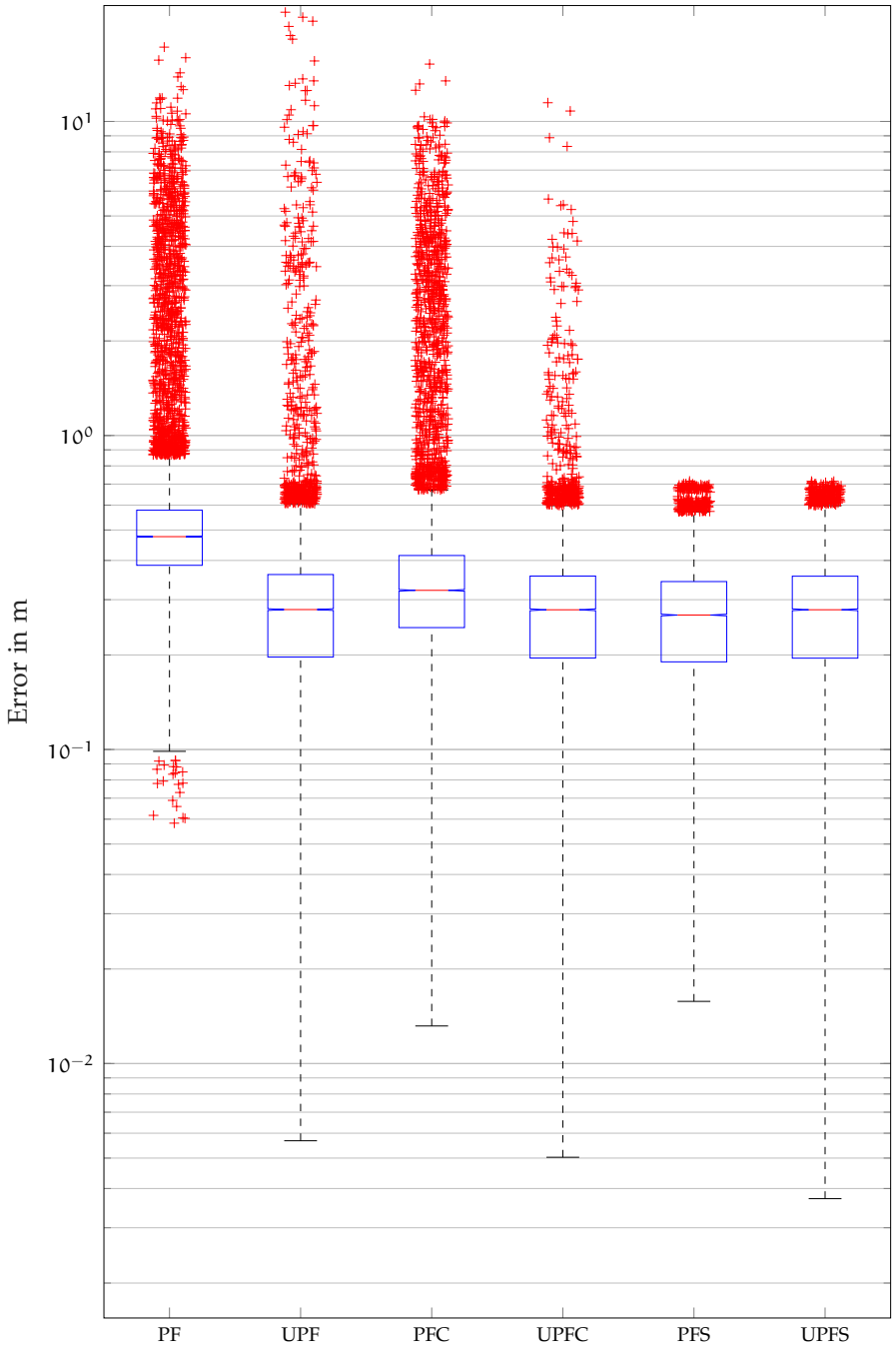


Figure 41: Box plot of the estimation errors in Scenario 2 (logarithmic scale). `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

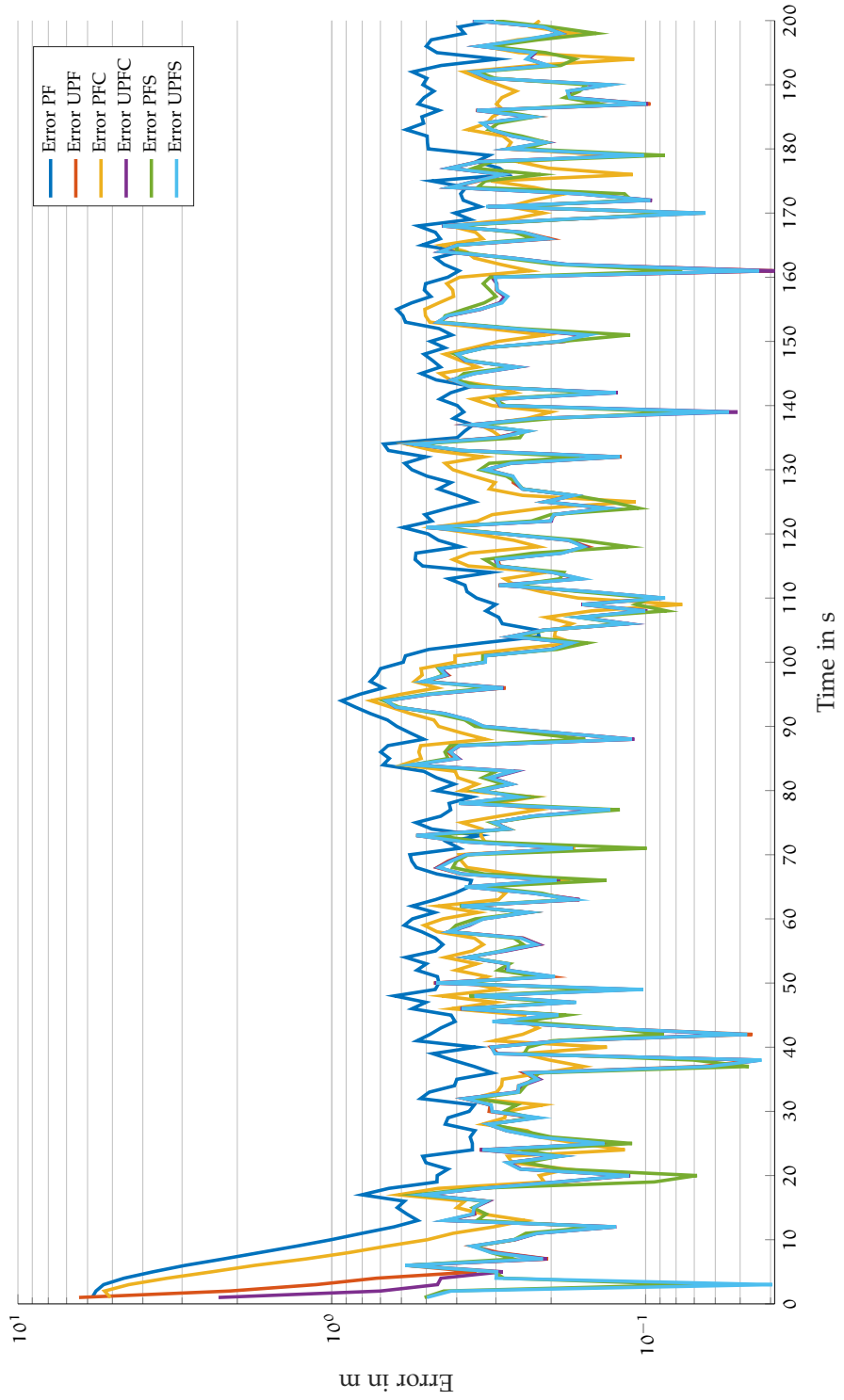


Figure 42: Mean estimation error over time in Scenario 2 (logarithmic scale). `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

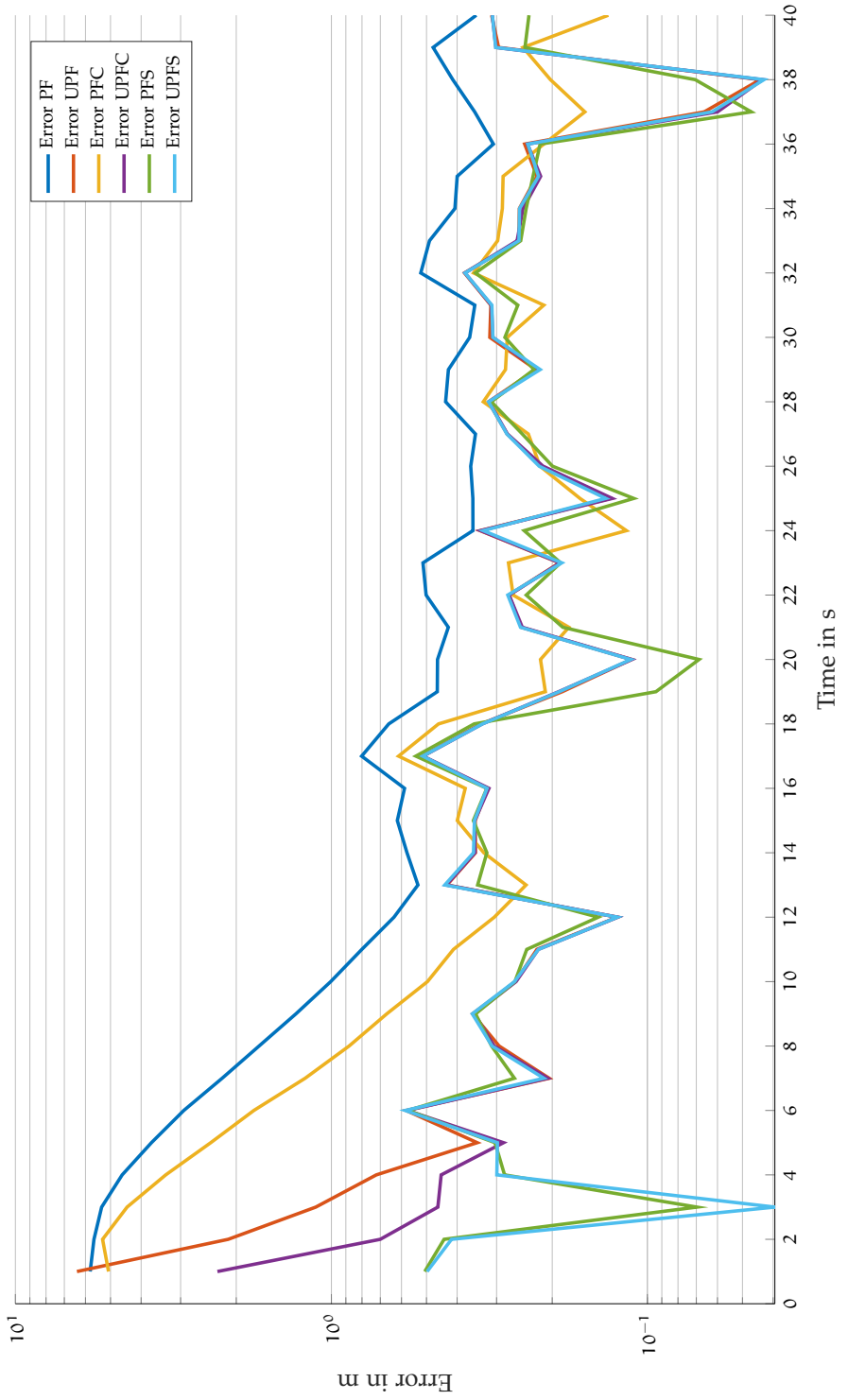


Figure 43: First 40 seconds of the mean estimation error over time in Scenario 2 (logarithmic scale). PF, PFC, PFS: 10000 particles. UPF, UPFC, UPFS: 100 particles.

4.4.4 *Results with 4 Landmarks and Kidnapping*

Figures 44 - 46 depicting the estimation results for Scenario 2 with kidnapping.

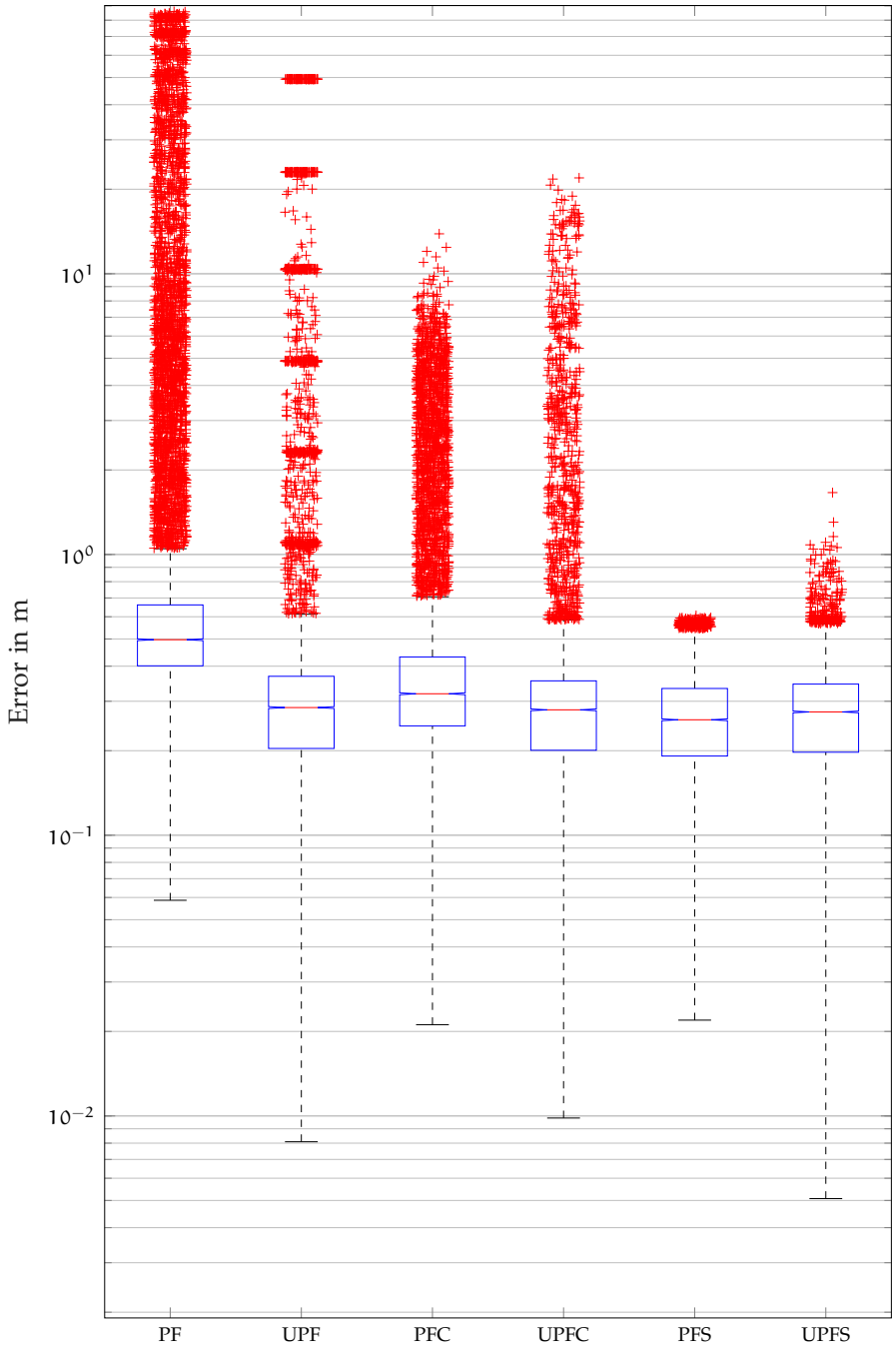


Figure 44: Box plot of the estimation errors in Scenario 2 with kidnapping (logarithmic scale). `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

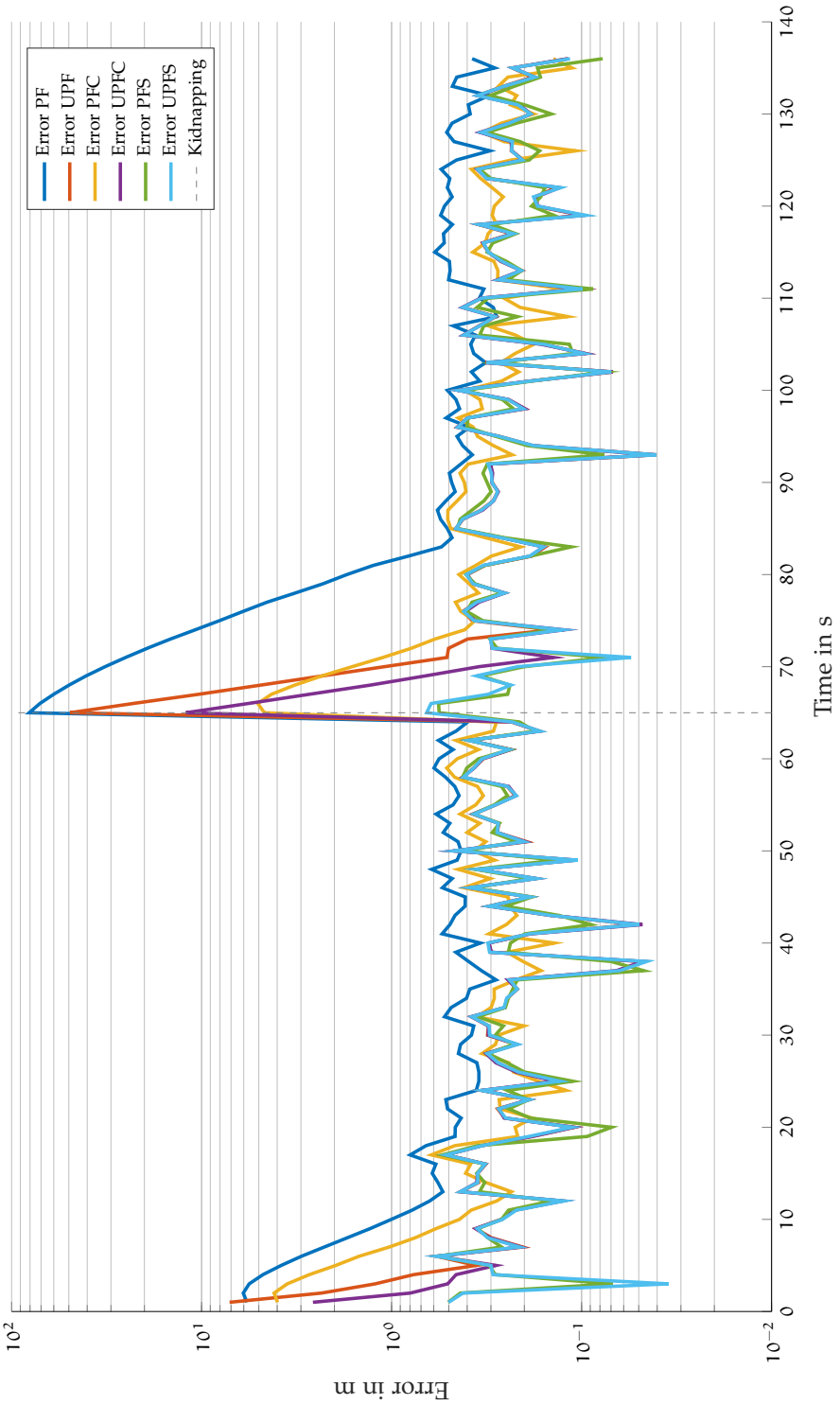


Figure 45: Mean estimation error over time in Scenario 2 with kidnapping (logarithmic scale). PF, PFC, PFS: 10000 particles. UPF, UPFC, UPFS: 100 particles.

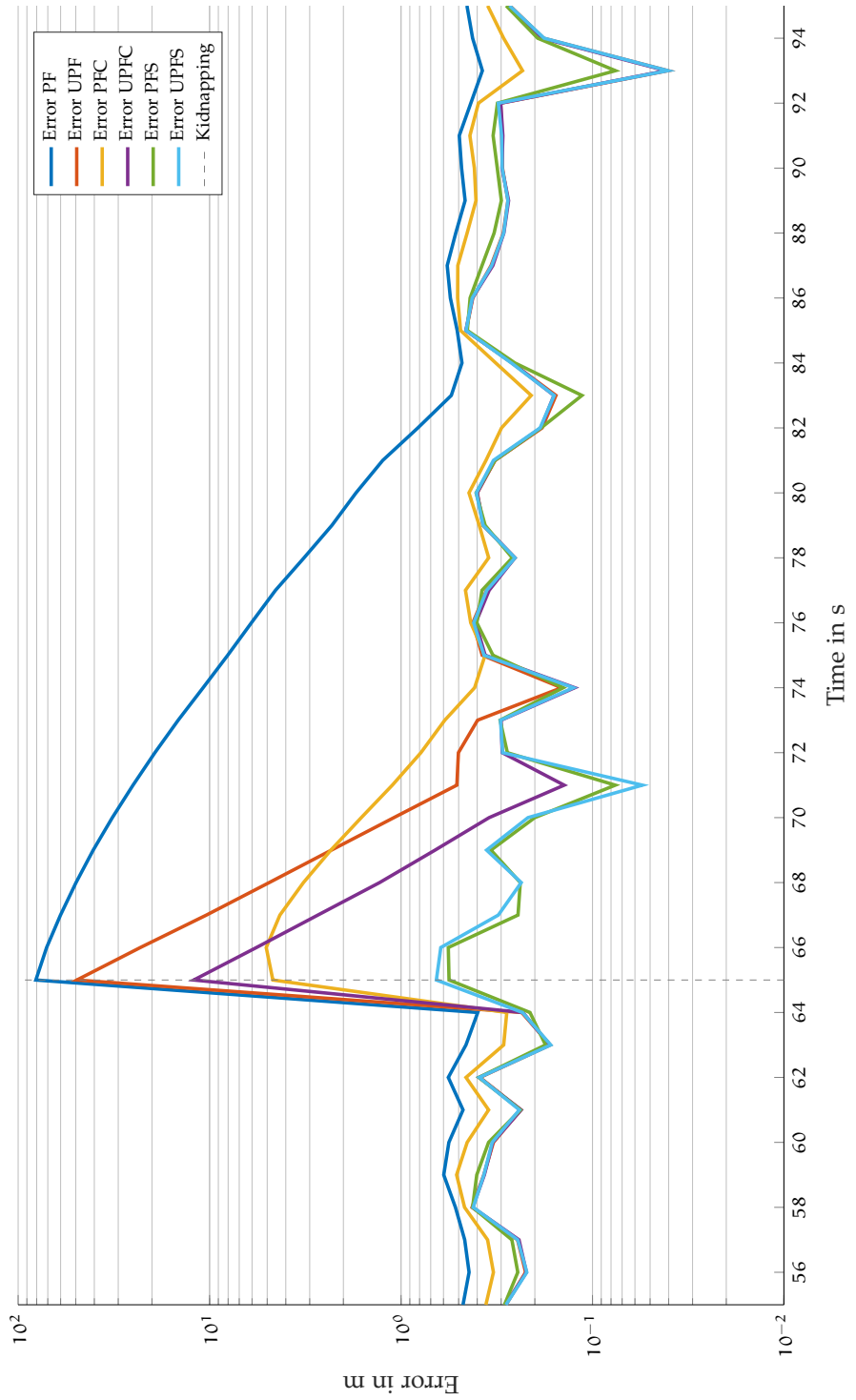


Figure 46: First 30 seconds of the mean estimation error over time after kidnapping in Scenario 2 (logarithmic scale). `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

4.4.5 *Results with 9 Landmarks*

Figures 47 - 49 depicting the estimation results for Scenario 3.

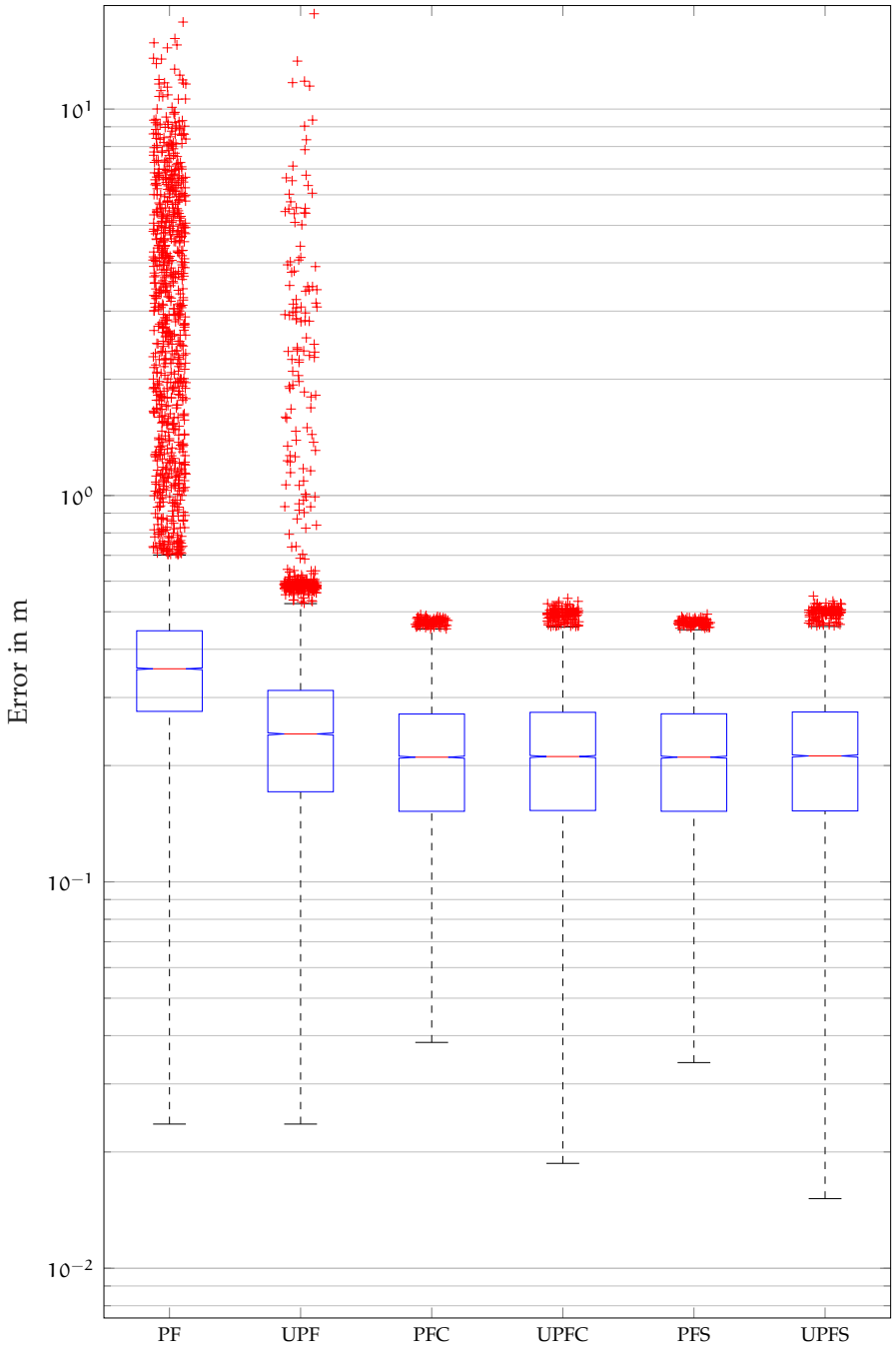


Figure 47: Box plot of the estimation errors in Scenario 3 (logarithmic scale). `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

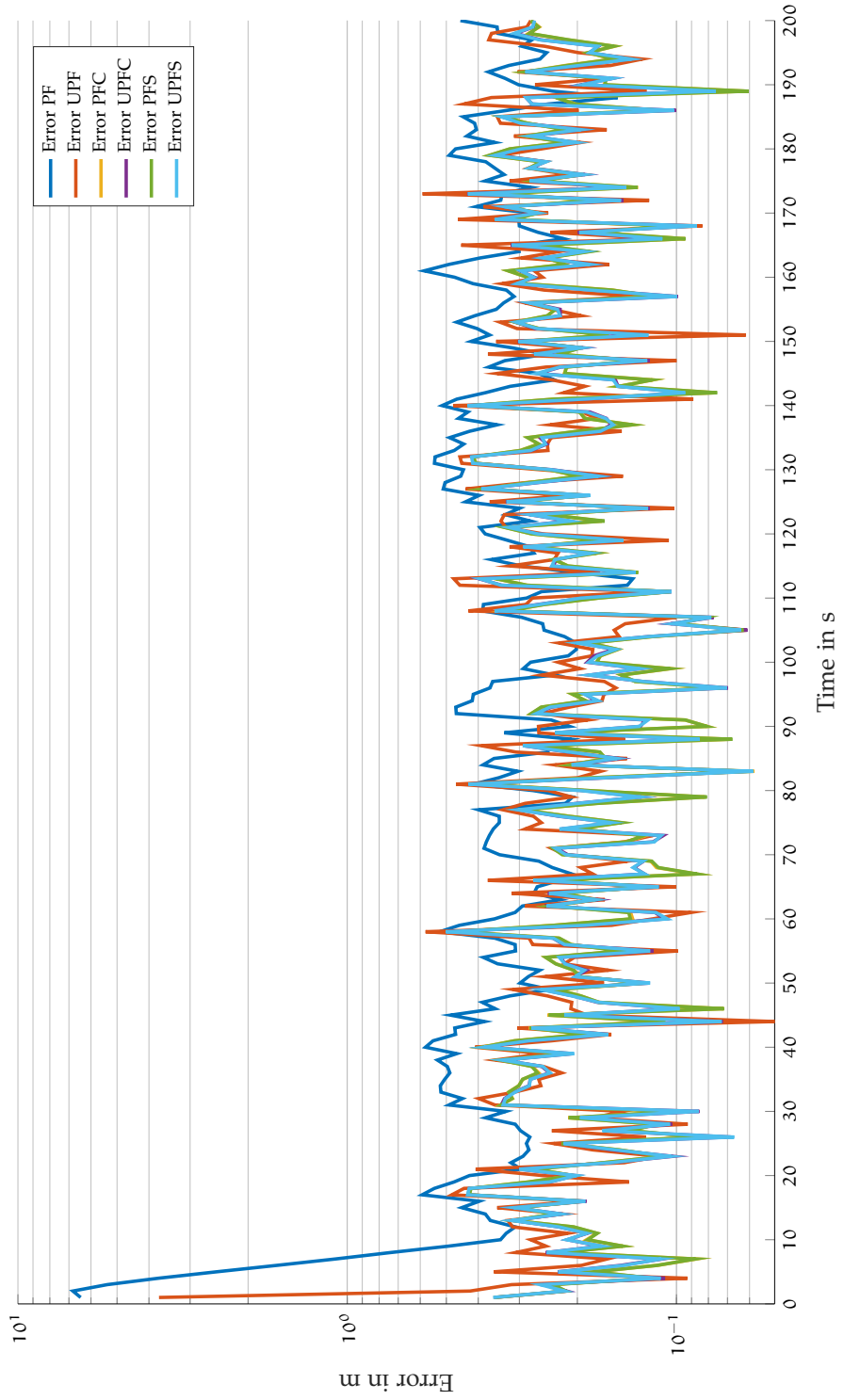


Figure 48: Mean estimation error over time in Scenario 3 (logarithmic scale). `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

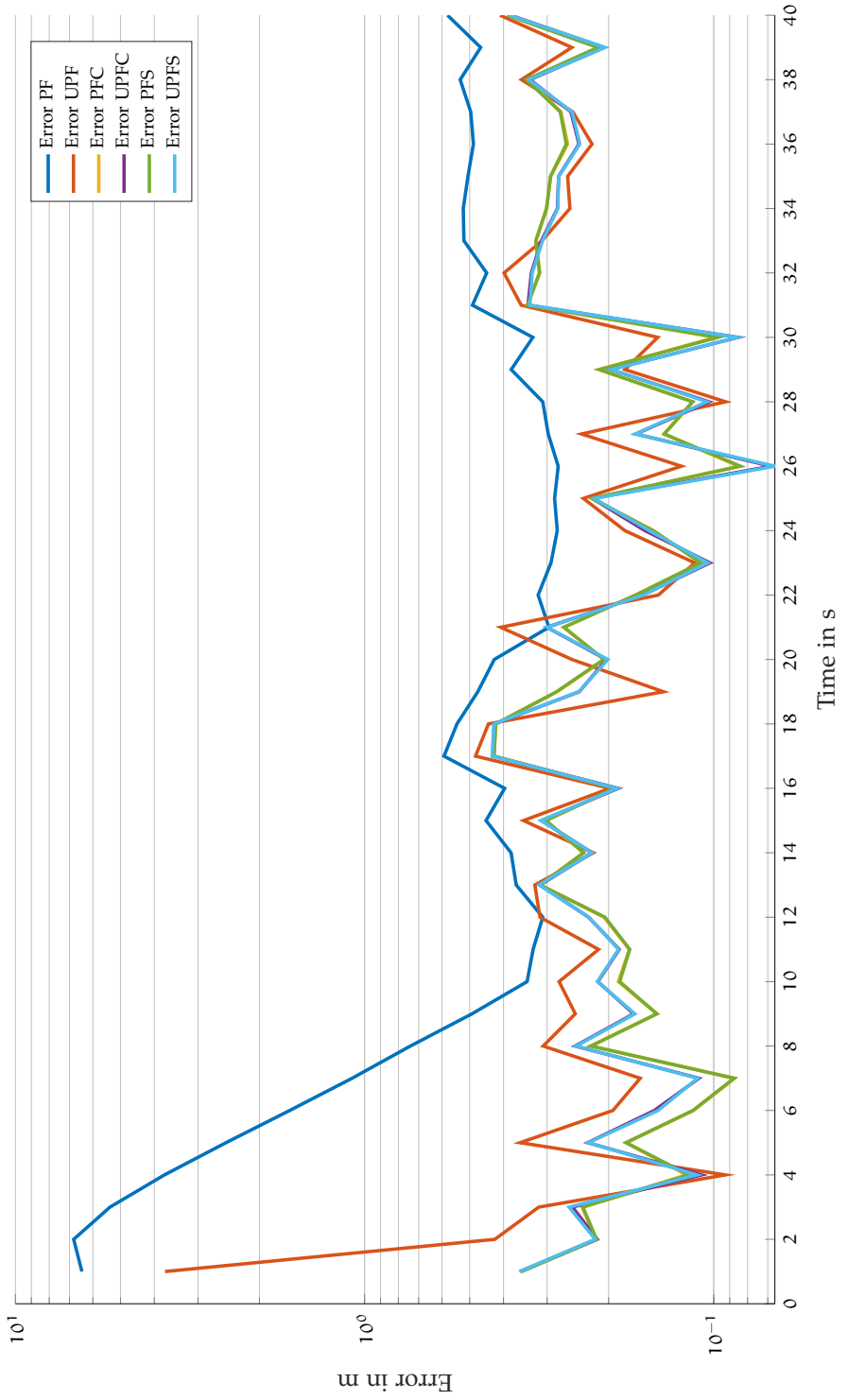


Figure 49: First 40 seconds of the mean estimation error over time in Scenario 3 (logarithmic scale). `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

4.4.6 *Results with 9 Landmarks and Kidnapping*

Figures 50 - 52 depicting the estimation results for Scenario 3 with kidnapping.

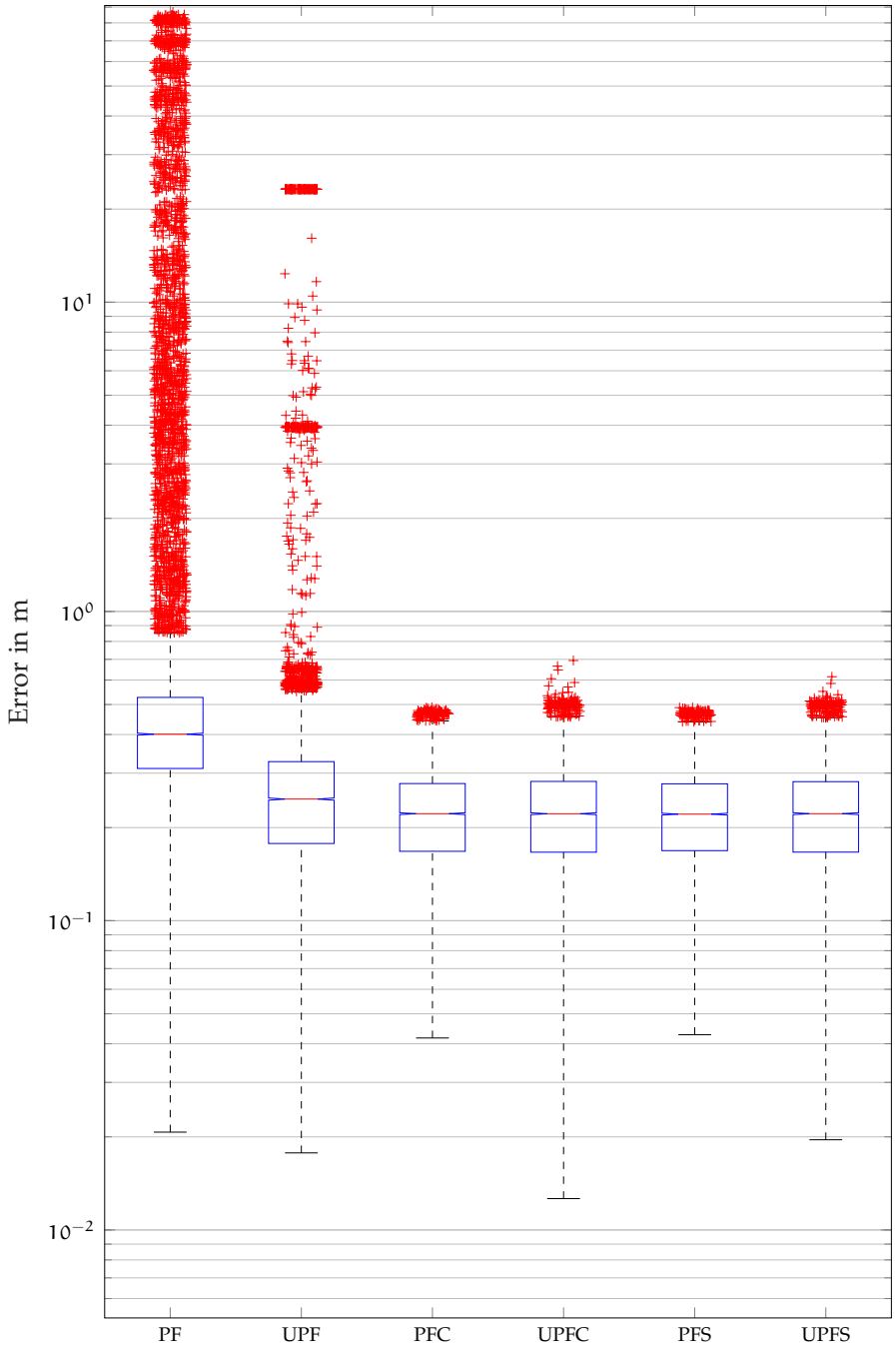


Figure 50: Box plot of the estimation errors in Scenario 3 with kidnapping (logarithmic scale). `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

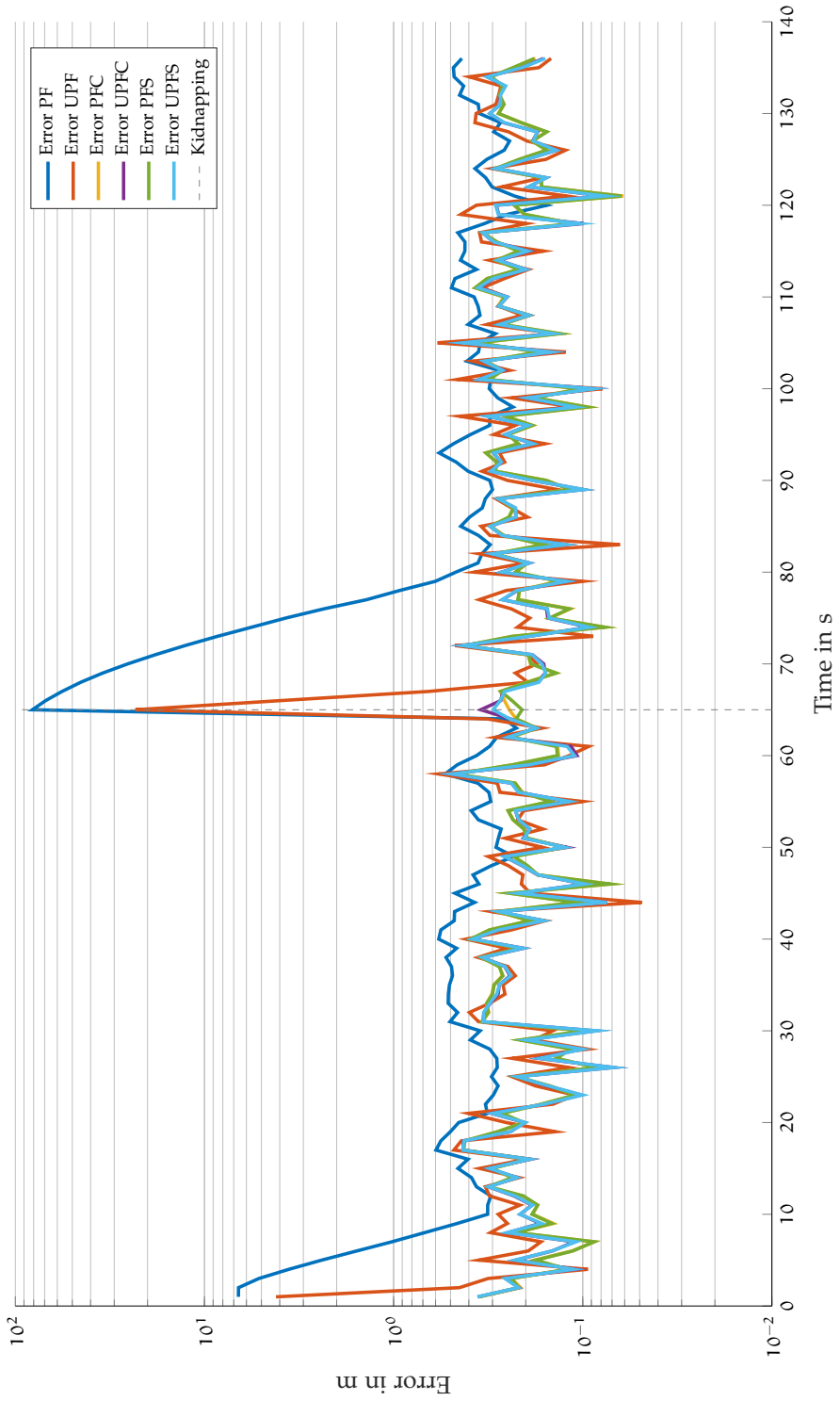


Figure 51: Mean estimation error over time in Scenario 3 with kidnapping (logarithmic scale). `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

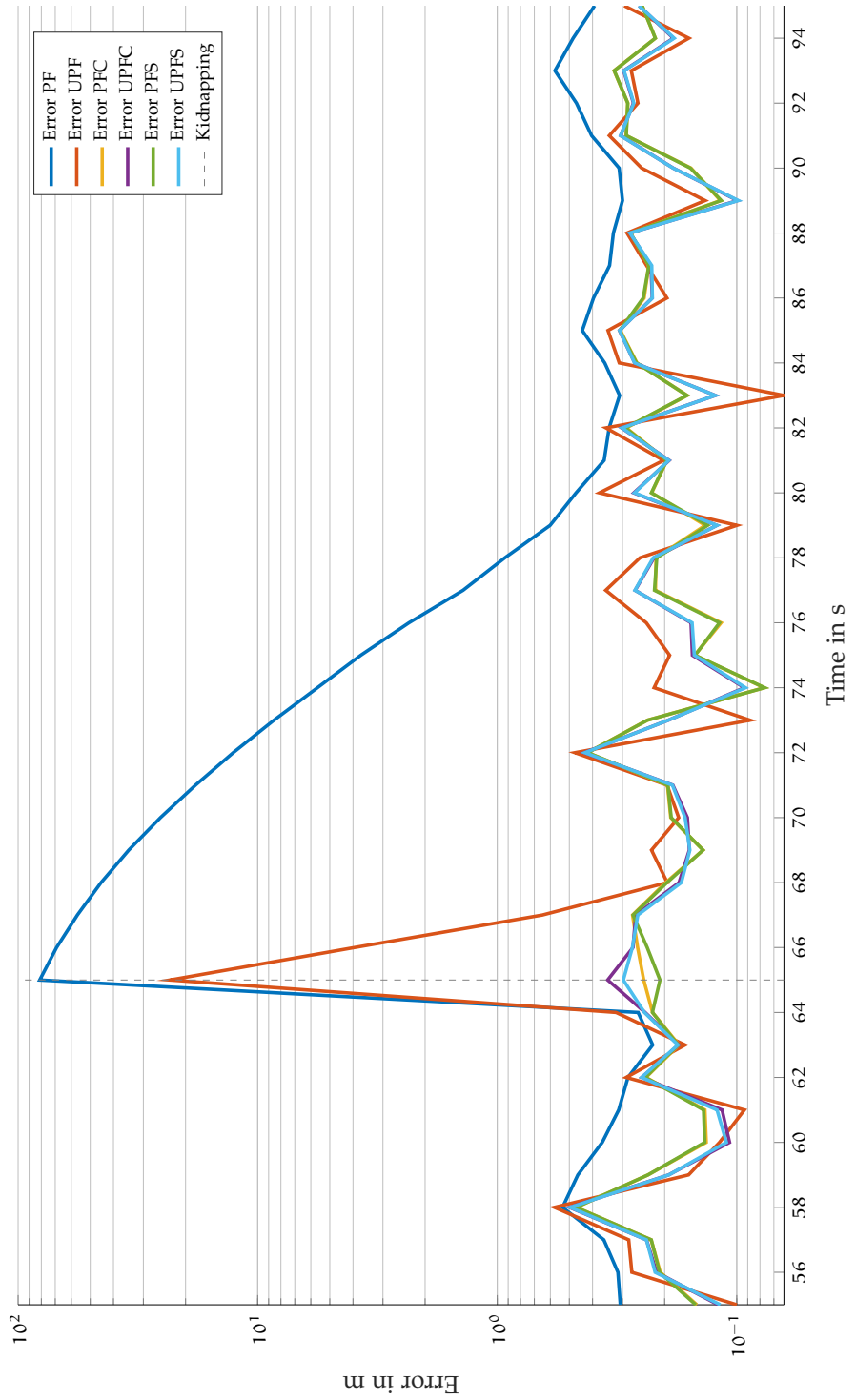


Figure 52: First 30 seconds of the mean estimation error over time after kidnapping in Scenario 3 (logarithmic scale). `PF`, `PFC`, `PFS`: 10000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

	Wake-up r. p.			Kidnapped r. p.		
	PF	PFC	PFS	PF	PFC	PFS
Median error in m	18.6	9.9	5.6	79.6	20.0	9.8
Improvement in %	–	47	70	–	75	88

Table 5: Median errors for the bootstrap filters in Scenario 1.

4.5 DISCUSSION

Given the results presented in the previous section, we will now discuss them in detail, first comparing the new against the respective conventional localisation algorithms in the following two sections and then comparing the bootstrap particle filters against the unscented particle filters in Section 4.5.3. Finally, we will discuss the differences between the results of the filters using the HC4 contractor and those using SIVIA.

4.5.1 *Constraint vs. Unconstraint Bootstrap Filters*

First, the constraint bootstrap filters (PFC, PFS) and the unconstrained bootstrap filter (PF) are contrasted in the respective localisation scenarios. Comparing the median estimation error of the bootstrap particle filter with HC4 contractor and with SIVIA, respectively, against the median error of the conventional bootstrap filter, one can see that both new filter algorithms consistently outperform the conventional one in each of the experiments, while the median error of the PFS is smaller or equal to that of the PFC (cf. box plots in Figures 35, 38, 41, 44, 47, 50). However, the difference in median errors is less pronounced for an increased number of landmarks, as can be seen in the box plots for the scenarios with four and nine landmarks in Figures 41, 44, 47, 50. In these scenarios, in turn, the initial estimation error and the error right after kidnapping is reduced significantly (cf. mean error plots in Figures 43, 46, 49, 52). It should be noted that in these figures the higher error of the conventional bootstrap filter after convergence is due to different tuning of the filters, which takes into account the confined search space in case of the hybrid filter algorithms.

In Scenario 1, with a median error of 9.9 metres for the PFC and 18.6 metres for the conventional bootstrap filter, this represents an improvement of 47 percent. The PFS achieves a median error of less than 5.6 metres, which makes up for an improvement of 70 percent (cf. box plot in Figure 35). When kidnapping the robot in Scenario 1, the respective improvements are 75 percent for the PFC and 88 percent for the PFS, given median errors of 79.6 metres (PF), 20 metres (PFC),

	Wake-up r. p.			Kidnapped r. p.		
	PF	PFC	PFS	PF	PFC	PFS
Mean initial error in m	5.8	5.1	0.5	80.8	4.7	0.6
Improvement in %	–	13	91	–	94	99

Table 6: Mean initial errors and errors after kidnapping for the bootstrap filters in Scenario 2.

and 9.8 metres (PFS), which can also be seen in the box plot in Figure 38. The median errors and the respective improvement in percent is contrasted in Table 5.

In the presence of four landmarks, the mean initial estimation error is reduced by 13 percent for the PFC and by 91 percent for the PFS, given the mean initial errors of 5.8 metres (PF), 5.1 metres (PFC), and 0.5 metres (PFS), which can be seen in the mean error plot in Figure 43. The major difference in error between the PFC and the PFS will be discussed in Section 4.5.4. The mean first error after kidnapping is reduced by 94 percent for the PFC and by 99 percent for the PFS, given the mean initial errors of 80.8 metres (PF), 4.7 metres (PFC), and 0.6 metres (PFS), which are depicted in the mean error plot in Figures 46. The mean initial errors and the respective improvement in percent are contrasted in Table 6. The lower initial errors and the faster convergence of the two hybrid bootstrap filters are also reflected in the number of outliers in the box plots depicted in Figures 41 and 44.

In Scenario 3, the error of the PFC and PFS is smaller than 60 centimetres throughout the whole estimation process. That is, both hybrid bootstrap filters converge instantly, as can be seen in the mean error plots in Figure 49 without kidnapping and in Figure 52 with kidnapping. In contrast, due to the high initial errors, the conventional bootstrap filter only achieves an upper error bound of 16.8 metres without kidnapping (cf. box plots in Figures 47) and of 86.7 metres with kidnapping (cf. box plot in Figures 50). The major improvement is due to the bounded-error estimation that is carried out when performing global localisation in the beginning and the reliable detection of kidnapping, which restarts the global localisation process likewise. The individual mean initial errors and the respective improvement in percent are contrasted in Table 7 (cf. also mean error plots in Figures 49, 52).

	Wake-up r. p.			Kidnapped r. p.		
	PF	PFC	PFS	PF	PFC	PFS
Mean initial error in m	6.5	0.36	0.36	81.2	0.24	0.21
Improvement in %	–	94	94	–	99	99

Table 7: Mean initial errors and errors after kidnapping for the bootstrap filters in Scenario 3.

4.5.2 *Constraint vs. Unconstraint Unscented Particle Filters*

In this section, the constraint unscented particle filters (UPFC, UPFS) are compared against the unconstrained unscented particle filter (UPF). With two available landmarks, the median estimation error of the constraint unscented particle filters is up to 30 metres smaller than that of the conventional unscented particle filters (cf. box plot in Figure 35), while both UPFC and UPFS have similar initial estimation errors before they slightly drift apart, as can be seen in Figure 36. This can be explained by the probabilistic nature of the filters. Looking at the results of the individual runs in the GitHub repository [102], one can observe the high error variance of the UPFS.

When four or nine landmarks are available, the median errors of both the UPFC and the UPFS are similar to that of the UPF (cf. box plots in Figures 41, 44, 47, 50). However, in these scenarios the mean initial estimation error as well as the mean first error after kidnapping is significantly reduced by the bounded-error state estimation (cf. mean error plots in Figures 43, 46, 49, 52). Both the lower initial errors and the faster convergence of the two hybrid unscented particle filters is also reflected in the number of outliers in the box plots depicted in Figures 41, 44, 47, 50.

In the presence of four landmarks, the mean initial estimation error is reduced by 64 percent for the UPFC and by 92 percent for the UPFS, given the mean initial errors of 6.4 metres (UPF), 2.3 metres (UPFC), and 0.5 metres (UPFS), which can be seen in the mean error plot in Figure 43. The mean first error after kidnapping is reduced by 76 percent for the UPFC and by 98 percent for the UPFS, given the mean initial errors of 49.4 metres (UPF), 12.1 metres (UPFC), and 0.7 metres (UPFS), which are depicted in the mean error plot in Figures 46. The mean initial errors and the respective improvement in percent are contrasted in Table 8.

In Scenario 3, the error of the UPFC and UPFS is always smaller than 70 cm (Figures 47, 50), whereas the upper error bound remains high for the conventional unscented particle filter at 23.3 metres with kidnapping and 17.6 metres without kidnapping. Again, this is due

	Wake-up r. p.			Kidnapped r. p.		
	UPF	UPFC	UPFS	UPF	UPFC	UPFS
Mean initial error in m	6.4	2.3	0.5	49.4	12.1	0.7
Improvement in %	–	64	92	–	76	98

Table 8: Mean initial errors and errors after kidnapping for the unscented particle filters in Scenario 2.

	Wake-up r. p.			Kidnapped r. p.		
	UPF	UPFC	UPFS	UPF	UPFC	UPFS
Mean initial error in m	3.7	0.4	0.4	23.2	0.3	0.3
Improvement in %	–	89	89	–	99	99

Table 9: Mean initial errors and errors after kidnapping for the unscented particle filters in Scenario 3.

to the reduction in the size of the initial search space achieved by the bounded-error estimation. The mean initial errors and the respective improvement in percent are contrasted in Table 9.

4.5.3 *Bootstrap Filters vs. Unscented Particle Filters*

When comparing the bootstrap particle filters (PF, PFC, PFS) against the unscented particle filters (UPF, UPFC, UPFS), the results confirm the theory. If the likelihood is very peaked, given very accurate measurements and an unambiguous scenario like Scenario 2 or 3, the unscented particle filter and its constrained counterparts make better use of the existing particles by incorporating the latest measurement into the proposal distribution. As can be seen in the mean error plots in Figures 43, 46, 49, 52, the mean estimation error of the unscented particle filters decreases faster than that of the bootstrap particle filters, since particles are actively moved towards the peak of the likelihood function. Especially when using a contractor, whose resulting box in Scenario 2 has still a considerable volume, the unscented particle filter can improve the estimation accuracy in the beginning of the global localisation process. Figure 43 shows how the mean initial error is 2.3 metres for the UPFC in contrast to 5 metres for the PFC, since the UPFC incorporates the first measurement when propagating the initial particle distribution. In addition to that, the UPFC increases the speed

of convergence. The mean estimation error falls below 1 metre in 7 seconds for the PFC, whereas the UPFC only takes 1 second.

When nine landmarks are visible, the constrained unscented particle filters do not noticeably improve the estimation accuracy when compared to the constrained bootstrap filters. The mean estimation errors of the PFC, UPFC, PFS, and UPFS are consistently below 70 centimetres, as can be seen in the box plots in Figures 47, 50 and the mean error plots in Figures 48, 51). This can be explained by the very narrow box that confines the initial set of uniformly spread particles. Given such narrow region, particles are already close to the peak of the likelihood function and the overlap of the proposal distribution and the posterior distribution is already large, due to the bounded-error estimation. The lower bound in the median error of approximately 20 centimetres is to be explained by the finitely accurate measurements. It may be possible to lower it by more accurate measurements, but using an increased number of particles or more elaborate filtering techniques will presumably not improve the estimation accuracy further, which for many practical applications appears to be sufficient.

When only two landmarks are available, that is the measurements are ambiguous and a large region of the state space has a high observation likelihood, the three unscented particle filters fail to converge quickly (cf. mean error plots in Figures 36, 39). As opposed to the bootstrap particle filters, which merely move the particles according to the system model, the unscented particle filters actively move particles to regions of high likelihood. However, if the likelihood is not concentrated in certain regions of the search space, this movement may in fact be counterproductive. In those unambiguous scenarios, the bootstrap filter indeed benefits from the fact that it only moves the particles according to the system model, so that physical movement of the robot adds information when the measurements themselves do not provide sufficient information to unambiguously localise the robot. In other words, when one does not know precisely enough where to move particles to, regarding the observations, one should merely move according to the system model.

In summary, the more information is available to move particles to regions of high likelihood, the more pronounced the improvement of an unscented particle filter will be. However, with a box estimate that has sufficiently small volume already and with an accurate system model, the effect may be insignificant.

4.5.4 *HC4 Contractor vs. SIVIA*

The SIVIA algorithm may find a better approximation of the solution set described by the constraints than the contractor, especially when its shape strongly deviates from that of a cuboid in three-dimensions or an n_x-orthotope for higher dimensional state spaces. However,

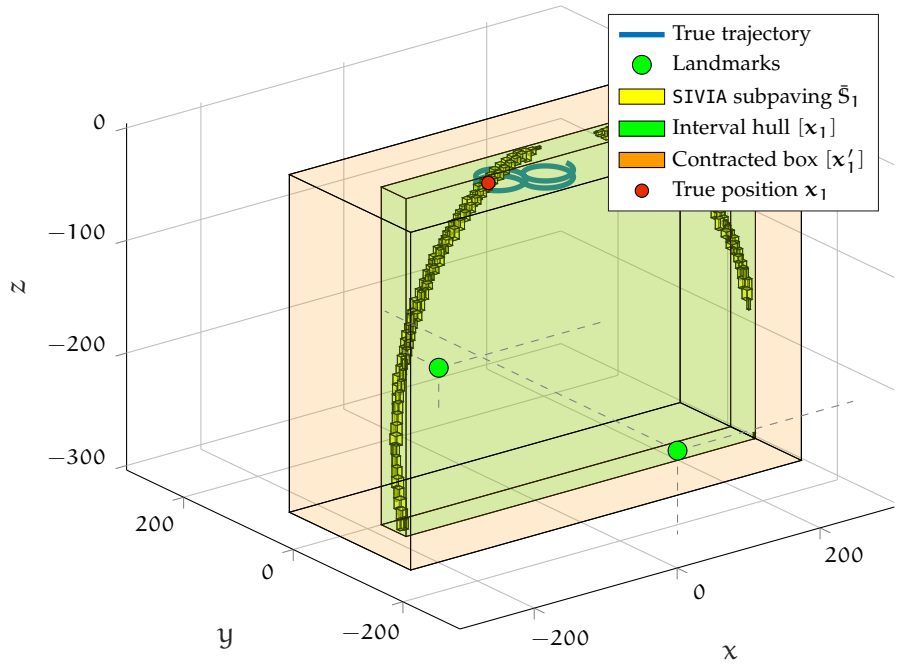


Figure 53: Comparison of the SIVIA subpaving $\bar{\mathbb{S}}_1$ its interval hull $[\mathbf{x}_1]$ and the contracted box $[\mathbf{x}'_1]$ in Scenario 1 with two landmarks.

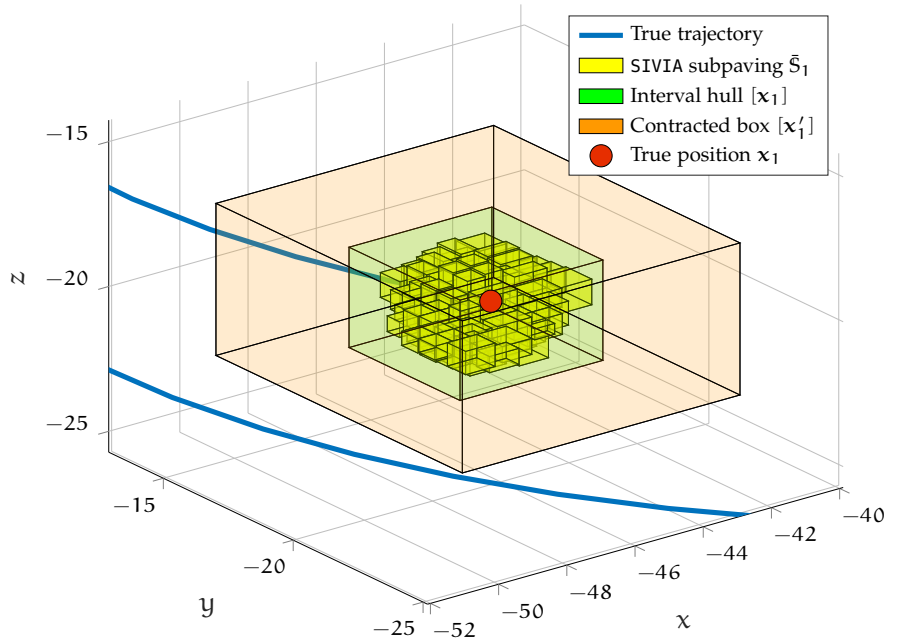


Figure 54: Comparison of the SIVIA subpaving $\bar{\mathbb{S}}_1$ its interval hull $[\mathbf{x}_1]$ and the contracted box $[\mathbf{x}'_1]$ in Scenario 3 with nine landmarks.

since we use the hull of the SIVIA subpaving as a confinement for the spreading of particles, in practice the difference may be minor, especially for an increased number of landmarks. This is depicted in Figures 53 and 54, which compare the interval hull $[\mathbf{x}_1]$ of the SIVIA subpaving with the contracted box, here denoted with $[\mathbf{x}'_1]$, both based on the first measurement in Scenario 1 and 3, respectively. The results show that the more information in terms of landmarks is available, the less marked the difference in volume and therefore the impact on the increase in estimation accuracy will be.

The box plots in Figures 47 and 50 show very similar errors for both the contractor and SIVIA in Scenario 3, while the box plots in Figure 41 and 44 indicate a difference in the error distribution for the contractor and SIVIA in Scenario 2. In particular, the number of outliers is reduced, as the initial errors are lower and therefore convergence takes place more rapidly. The volume of the contracted box is usually larger than that of the hull, but relating the difference in size to the vastly larger computational demands of SIVIA, for a practical application the difference may be insignificant and remains to be tested experimentally in a specific localisation scenario.

5 CONCLUSION AND FUTURE WORK

After an introduction into the subject matter, we have elaborated upon the theoretical foundations of probabilistic filtering and interval analysis. Subsequently, we presented a detailed description of the newly proposed filter algorithms. On the basis of the experiments and their results in the previous chapter, in this chapter we shall now conclude and present possible future work in Section 5.2.

5.1 CONCLUSION

Localisation is a fundamental problem in mobile robotics. When creating mobile robots that operate autonomously, possibly without any human supervision, accurate self-localisation is one of the fundamental abilities such robots should possess. At the same time, computational cost is a critical feature for many practical applications, which causes a natural dilemma between the localisation accuracy and the number of particles in real-life Monte Carlo localisation. Thus, it is desirable to make good use of particles by moving them to regions of the search space that are associated with a high observation likelihood.

Motivated by the above dilemma, we presented four new hybrid localisation algorithms based on non-linear bounded-error state estimation and probabilistic filtering. The rationale behind these new algorithms is to only perform Monte Carlo localisation over a limited region of the search space. Both a bootstrap filter and an unscented particle filter was combined with two bounded-error estimators, namely the forward-backward contractor and the Set Inverter via Interval Analysis, respectively.

As opposed to existing hybrid localisation approaches based on Monte Carlo simulation and interval analysis, in the new algorithms the bounded-error state estimate is not maintained throughout the whole estimation process, so as to save computational cost. Instead, additionally available information in the form of constraints based on geometrical considerations of the environment were incorporated in the Bayesian filters in order to improve the accuracy throughout the estimation process and to detect kidnapping. The bounded-error estimate is only computed in the beginning when solving the wake-up robot problem or after kidnapping. In comparison with the hybrid approaches in [27] and [28], the additional computational cost for the bounded-error state estimation is reduced drastically by not maintaining the box estimate throughout the entire estimation process, while the benefits in terms of an increase in the localisation accuracy

are preserved. Given a sequence of 200 measurements, like in the experiments above, when computing position estimates for the entire sequence of measurements, the bounded-error estimation is carried out once instead of 200 times, which demonstrates the potential for saving computational cost. When solving the kidnapped-robot problem, the reduction depends on the number of kidnappings. In the simulated scenarios, with one kidnapping per trajectory, the bounded-error position estimate is computed twice instead of 200 times.

Evaluating the performance of the new algorithms in various simulated underwater robot localisation scenarios, we have shown that the hypotheses are true and that the newly proposed algorithms are generally able to solve the wake-up robot problem as well as the kidnapped robot problem more accurately, when compared to conventional unconstrained probabilistic filtering, with an improvement of up to 88 percent in the median estimation error when compared to conventional filtering methods. In addition to that, the mean initial estimation error is significantly reduced by up to 94 percent and the mean error after kidnapping is reduced by up to 99 percent. The improvement is particularly pronounced when four or nine landmarks are available.

When the observation likelihood is very peaked, due to very accurate measurements, and when a sufficiently large number of landmarks is available to unambiguously determine the robot's position, the hybrid unscented particle filter performs as well as the hybrid bootstrap filter or better. Especially when the resulting box of the bounded-error localisation has a relatively large volume, the constrained unscented particle filters improve the estimation accuracy in the beginning and increase the speed of convergence, when compared to the three bootstrap filters.

With four or more landmarks, the bootstrap filter with `SIVIA` and the unscented particle filter with `SIVIA` deliver a very accurate position estimate with errors always smaller than 70 centimetres. If computational cost is a critical feature and a slightly higher initial error that decreases fast is tolerable in a certain application scenario, the two filters with the forward-backward contractor (`PFC`, `UPFC`) are the filters of choice, as the contractor is computationally less demanding than `SIVIA`. All four new algorithms detected kidnapping reliably in all the experiments so that the respective improvement after kidnapping is roughly according to that when solving the wake-up robot problem.

The four new localisation algorithms are not limited to underwater robot localisation but instead are applicable to any landmark-based localisation scenario. Of course, an estimation of the robot's pose or other state variables that are related to the position and orientation, such as the velocity, can easily be integrated in the algorithms as well. We shall now identify possible future work in order to further improve the proposed localisation algorithms.

5.2 FUTURE WORK

If the confined region obtained by bounded-error state estimation is narrow enough, the problem may be regarded as a tracking problem. In the scenario that contained nine landmarks, the volume of the contracted box or subpaving suggests that a uni-modal distribution may be a sufficiently accurate approximation of the true posterior probability distribution. Possible alterations of the proposed algorithms are using an unscented Kalman filter together with interval analysis in order to reduce computational cost. After a bounded-error state estimation algorithm has narrowed down the search space to a smaller region, a truncated unscented Kalman filter [107] could be utilised to obtain a trimmed Gaussian posterior probability distribution over the state. These modifications require one subpaving with a volume smaller than some predefined threshold.

If this condition is not fulfilled the proposed particle filter can be utilised until the particle variance is lower than some threshold. One may conclude that the filter has converged and that tracking is appropriate. Then, the particle filter could be replaced dynamically by a parameterised filter such as the unscented Kalman filter. All four hybrid localisation algorithms are designed so that any filter that matches the Bayesian framework can be plugged in easily to replace the respective particle filter.

Furthermore, it would be desirable to extend the new localisation algorithms so that they work with indistinguishable landmarks. Instead of a fixed number of landmarks, using a varying number at each time step would increase the freedom of application to a wider class of localisation problems and would map the inherently limited range of distance measurements in practice.

Regarding the bounded-error state estimator, an algorithm based on relaxed intersections, such as the Robust `SIVIA` algorithm [108], may be used in real-life scenarios where the measurement data can contain outliers that would cause an empty solution set with the conventional `SIVIA` algorithm.

In order to avoid the slow convergence of the UPF when only two landmarks are available, the influence of the latest measurement on the proposal distribution can be influenced by varying the measurement noise covariance matrix depending on the number of landmarks present. Then, when a sufficient number of landmarks is available but the volume of a contracted box or `SIVIA` hull is still large, the benefits of the unscented particle filter would manifest, whereas when the likelihood is not very peaked, a high measurement covariance value will dampen the impact of the unscented Kalman filter in the generation of the proposal distribution, so that the filter behaves like a bootstrap particle filter in such scenario.

APPENDIX

Photo of the Simulated Underwater Robot

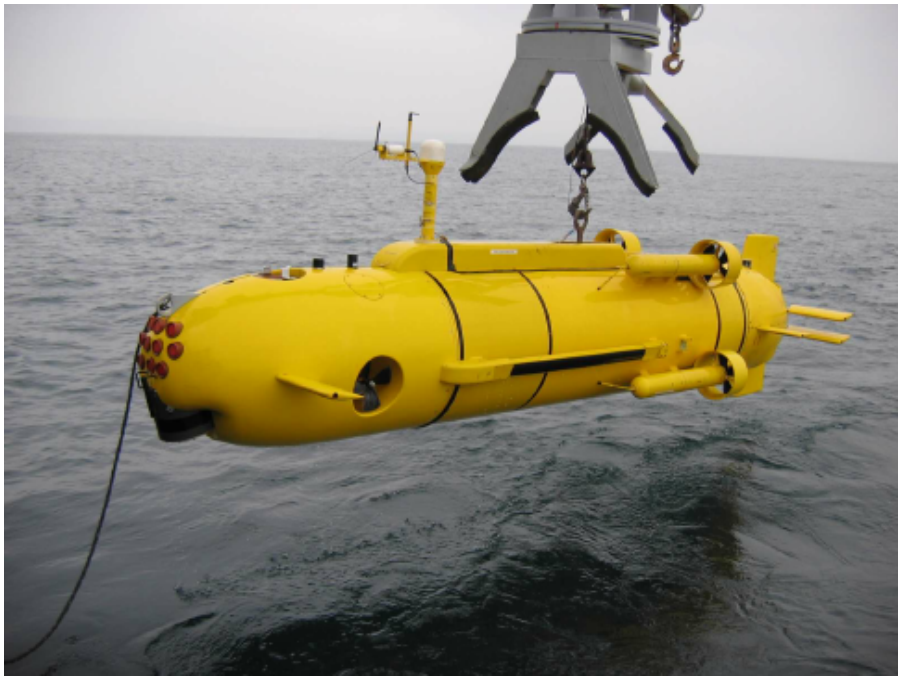


Figure 55: The autonomous underwater vehicle *Redermor*, developed by GESMA (Groupe d'Etude Sous-Marine de l'Atlantique).

Additional Results in Scenario 1 with 2 Landmarks

Table 10 and 11 list the figures depicting the estimation results for Scenario 1, with and without kidnapping, respectively.

Additional Results in Scenario 2 with 4 Landmarks

Table 12 and 13 list the figures depicting the estimation results for Scenario 2, with and without kidnapping, respectively.

Additional Results in Scenario 3 with 9 Landmarks

Table 14 and 15 list the figures depicting the estimation results for Scenario 3, with and without kidnapping, respectively.

Box plots	Mean errors	Number of particles	
		PF / PFC / PFS	UPF / UPFC / UPFS
35	36, 37	10000	1000
56	57, 58	1000	100

Table 10: List of figures depicting the estimation results in Scenario 1.

Box plots	Mean errors	Number of particles	
		PF / PFC / PFS	UPF / UPFC / UPFS
38	39, 40	10000	1000
59	60, 61	1000	100

Table 11: List of figures depicting the estimation results in Scenario 1 with kidnapping.

Box plots	Mean errors	Number of particles	
		PF / PFC / PFS	UPF / UPFC / UPFS
41	42, 43	10000	100
62	63, 64	1000	10

Table 12: List of figures depicting the estimation results in Scenario 2.

Box plots	Mean errors	Number of particles	
		PF / PFC / PFS	UPF / UPFC / UPFS
44	45, 46	10000	100
65	66, 67	1000	10

Table 13: List of figures depicting the estimation results in Scenario 2 with kidnapping.

Box plots	Mean errors	Number of particles	
		PF / PFC / PFS	UPF / UPFC / UPFS
47	48, 49	10000	100
68	69, 70	1000	10

Table 14: List of figures depicting the estimation results in Scenario 3.

Box plots	Mean errors	Number of particles	
		PF / PFC / PFS	UPF / UPFC / UPFS
50	51, 52	10000	100
71	72, 73	1000	10

Table 15: List of figures depicting the estimation results in Scenario 3 with kidnapping.

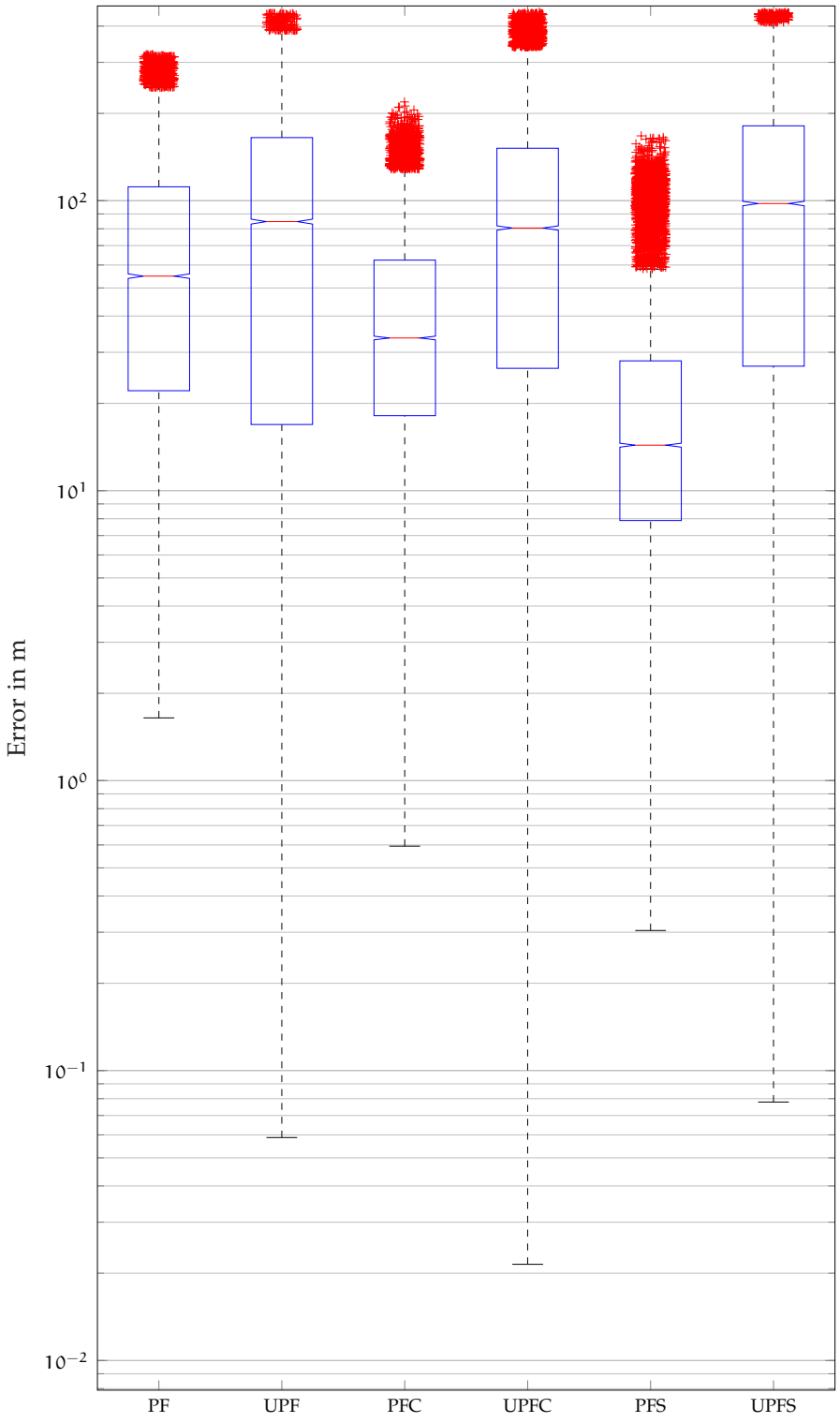


Figure 56: Box plot of the estimation errors in Scenario 1 (logarithmic scale). `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

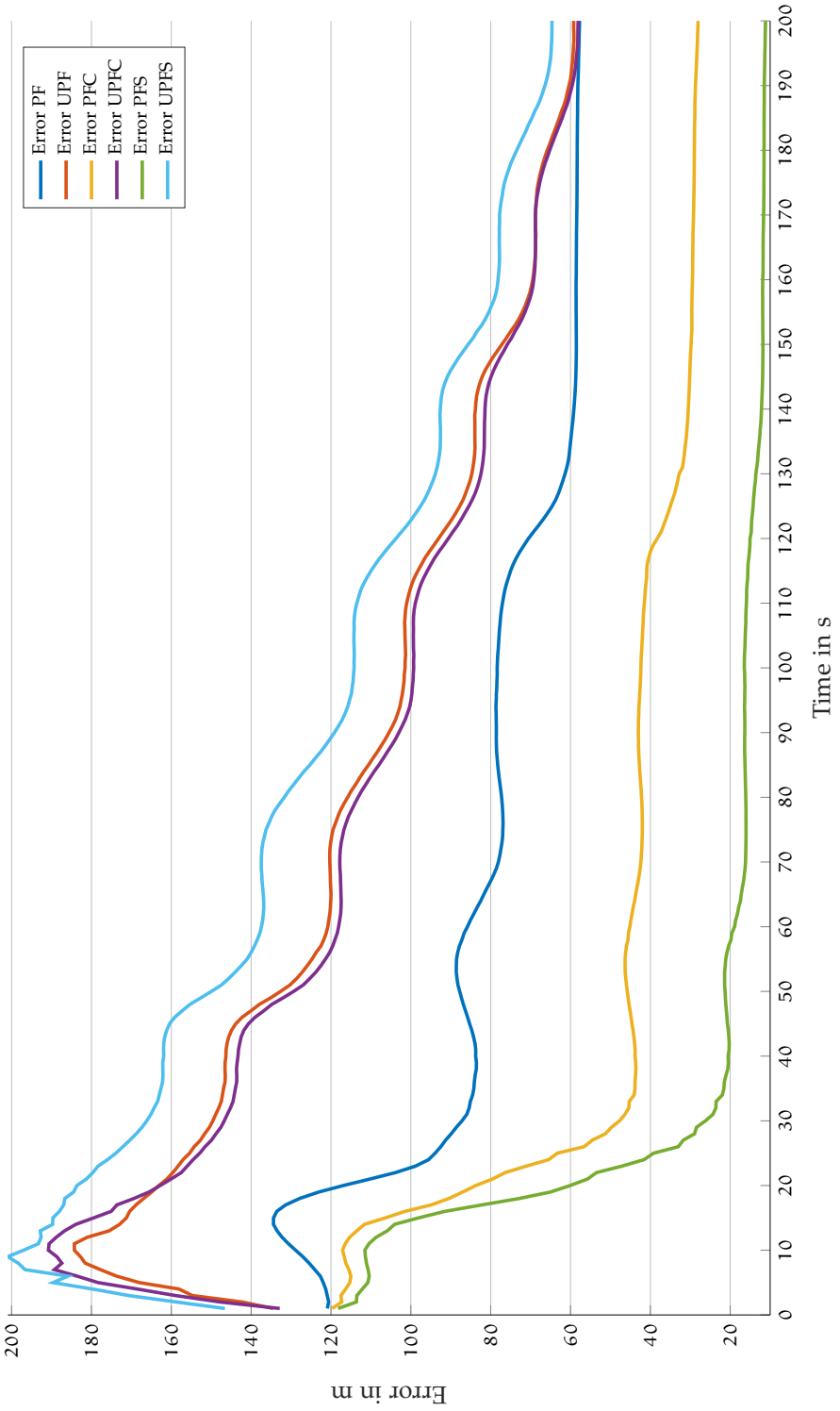


Figure 57: Mean estimation error over time in Scenario 1. `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

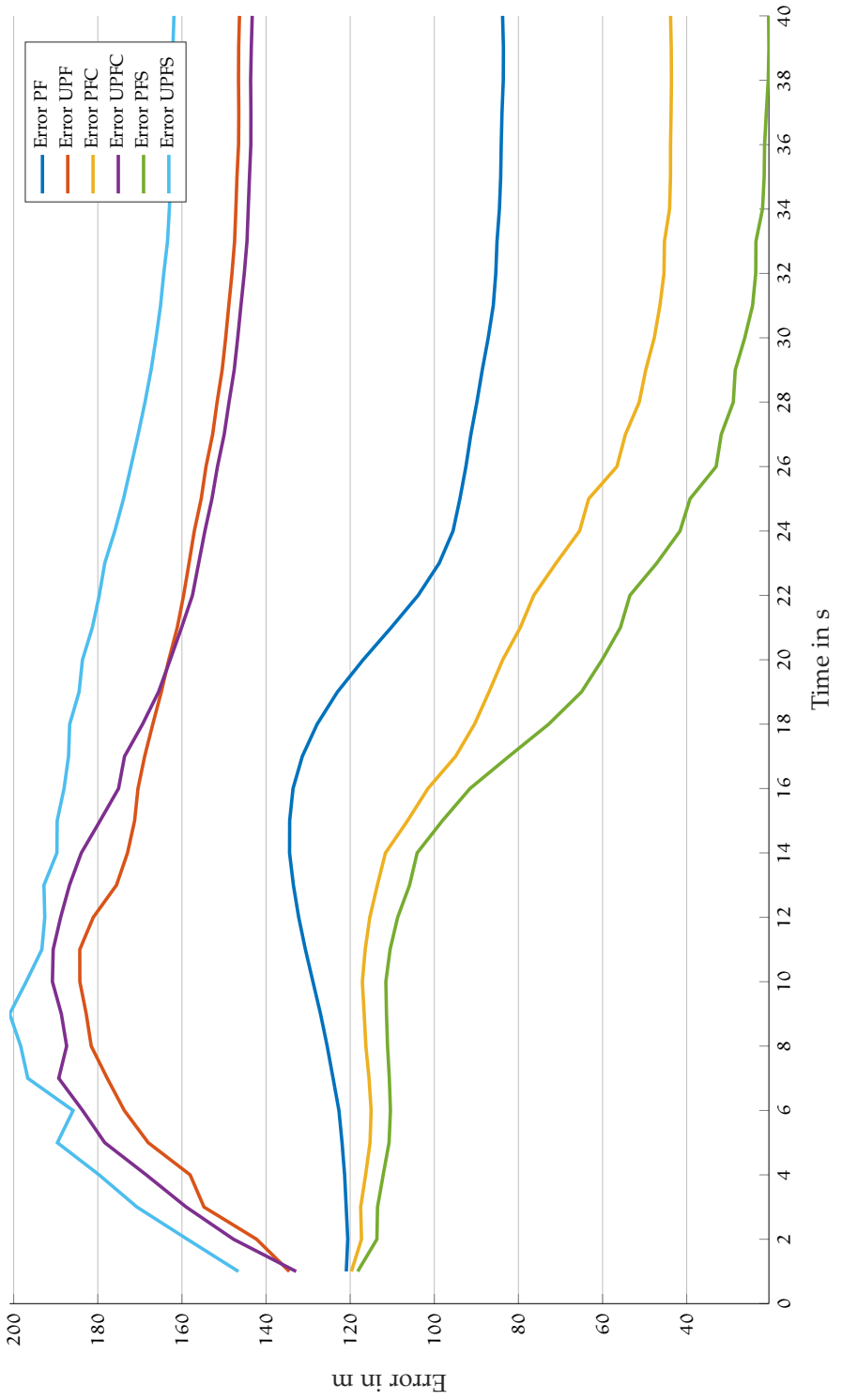


Figure 58: First 40 seconds of the mean estimation error over time in Scenario 1. `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

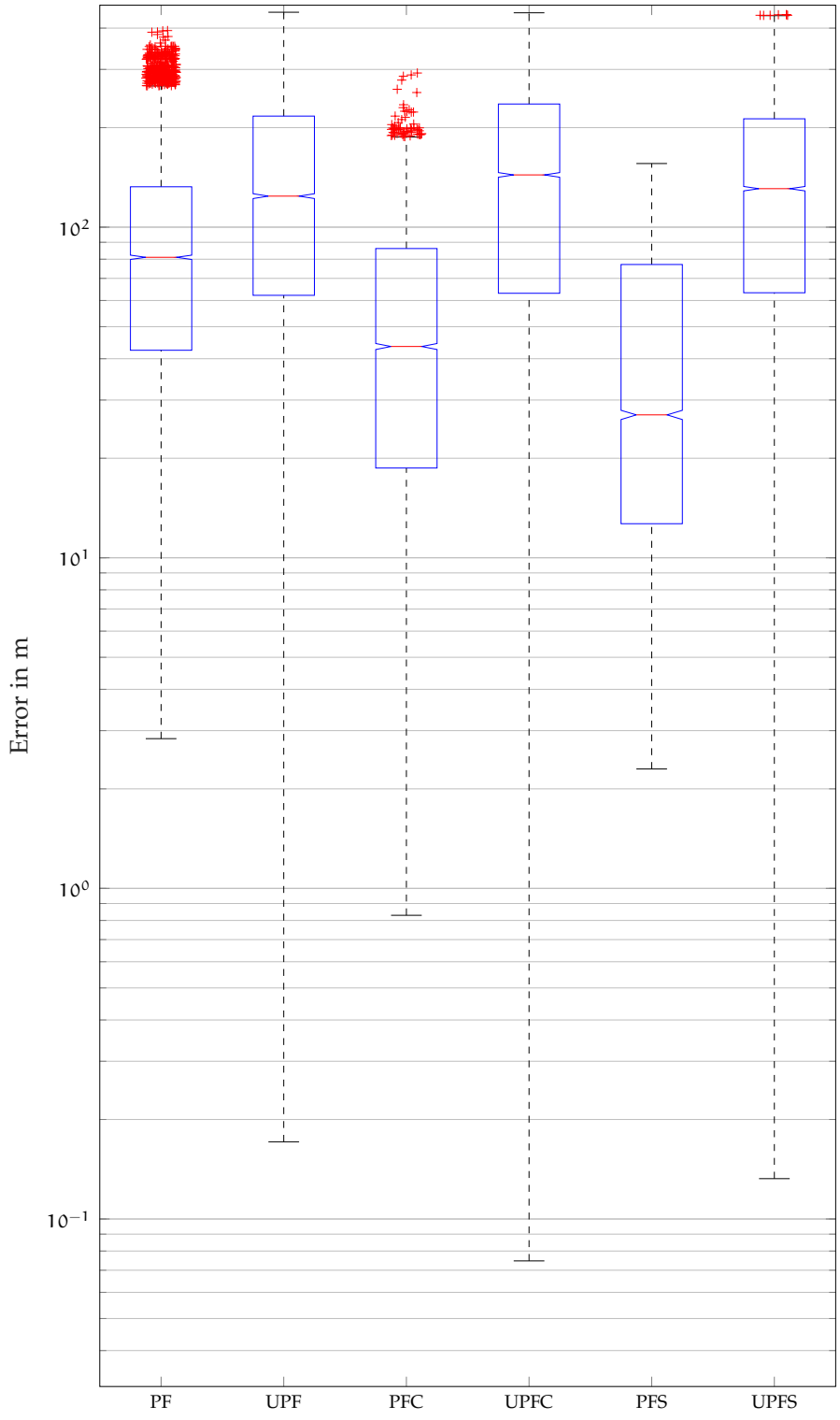


Figure 59: Box plot of the estimation errors in Scenario 1 with kidnapping (logarithmic scale). `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

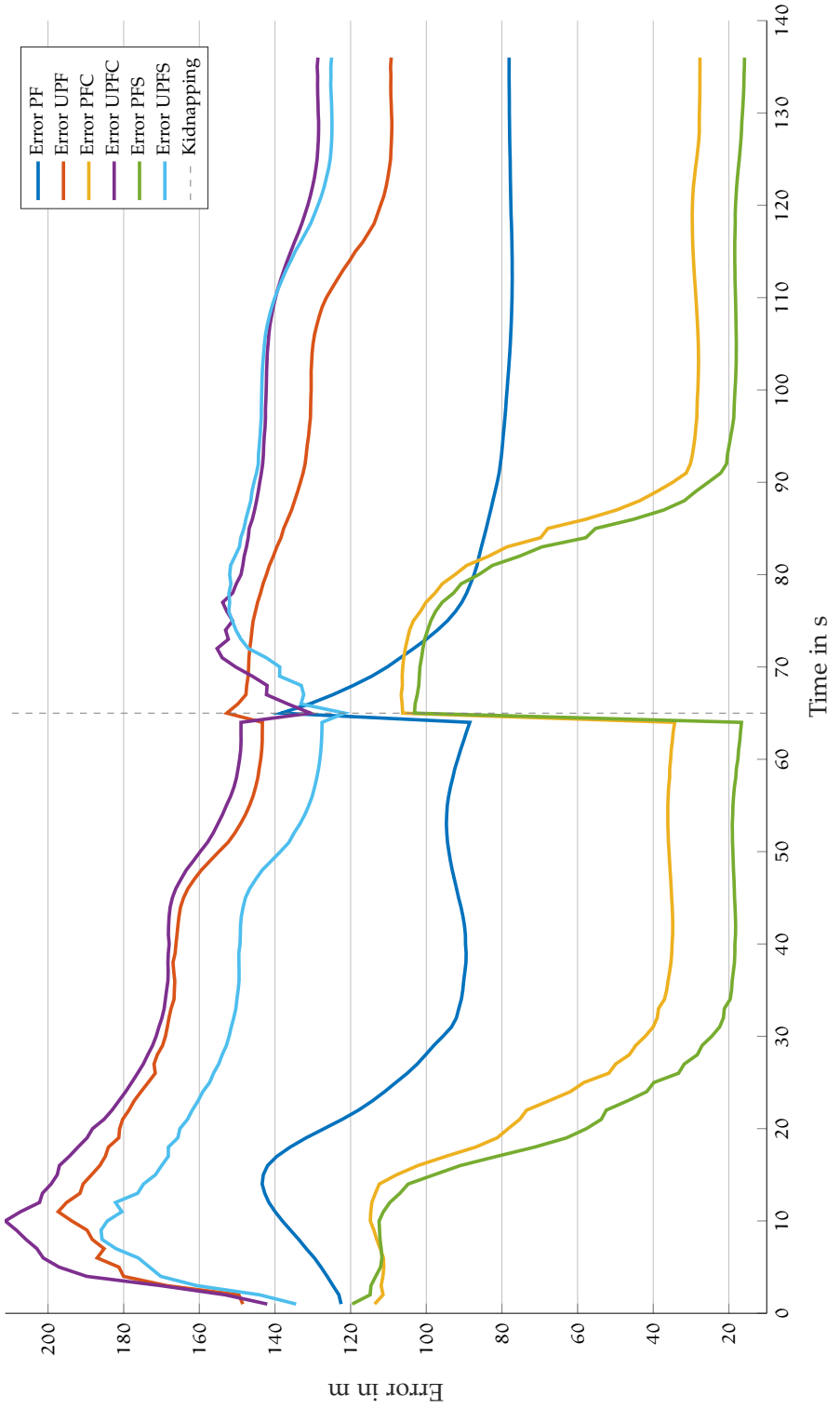


Figure 60: Mean estimation error over time in Scenario 1 with kidnapping. `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 10 particles.

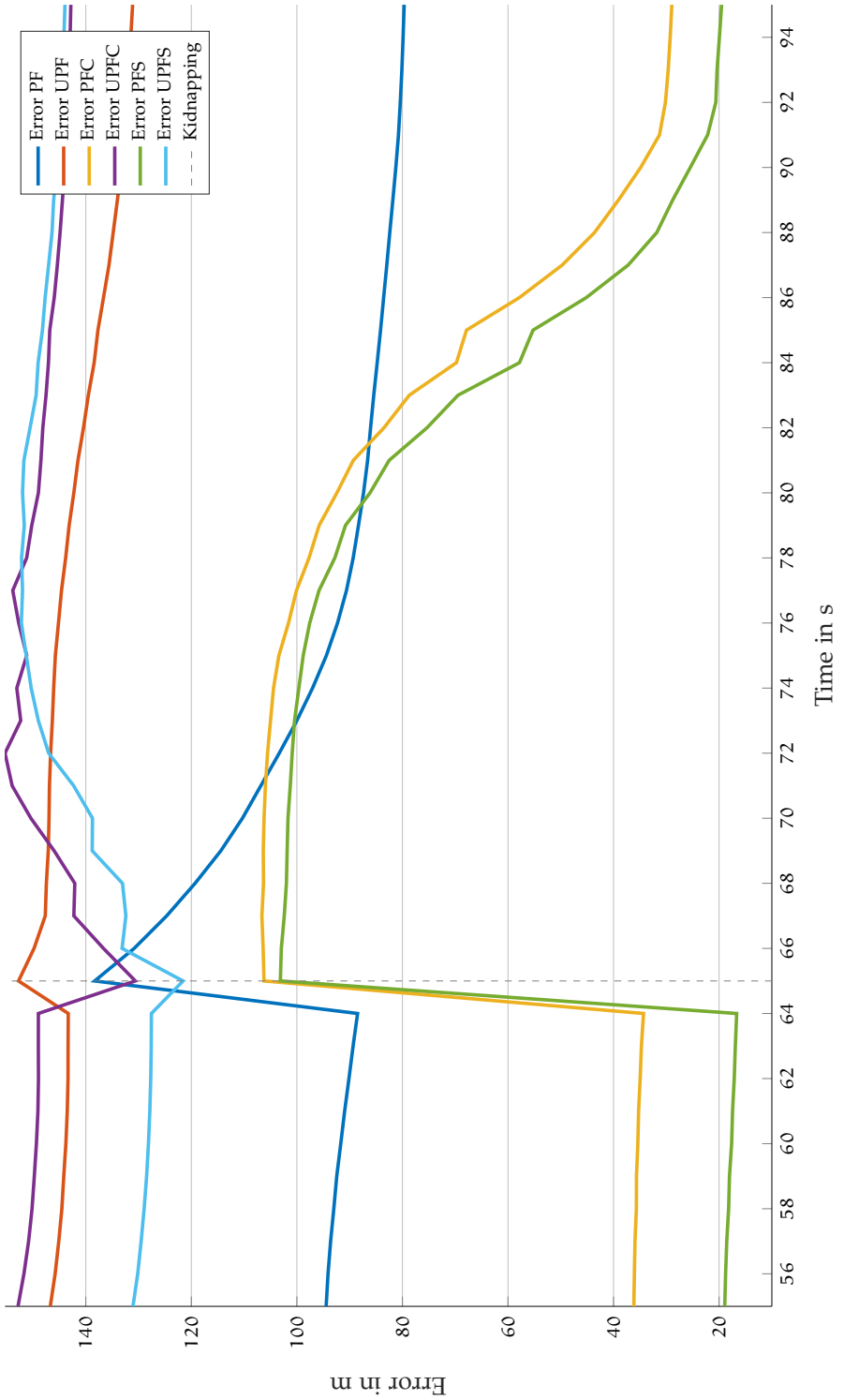


Figure 61: First 30 seconds of the mean estimation error over time after kidnapping in Scenario 1. `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 100 particles.

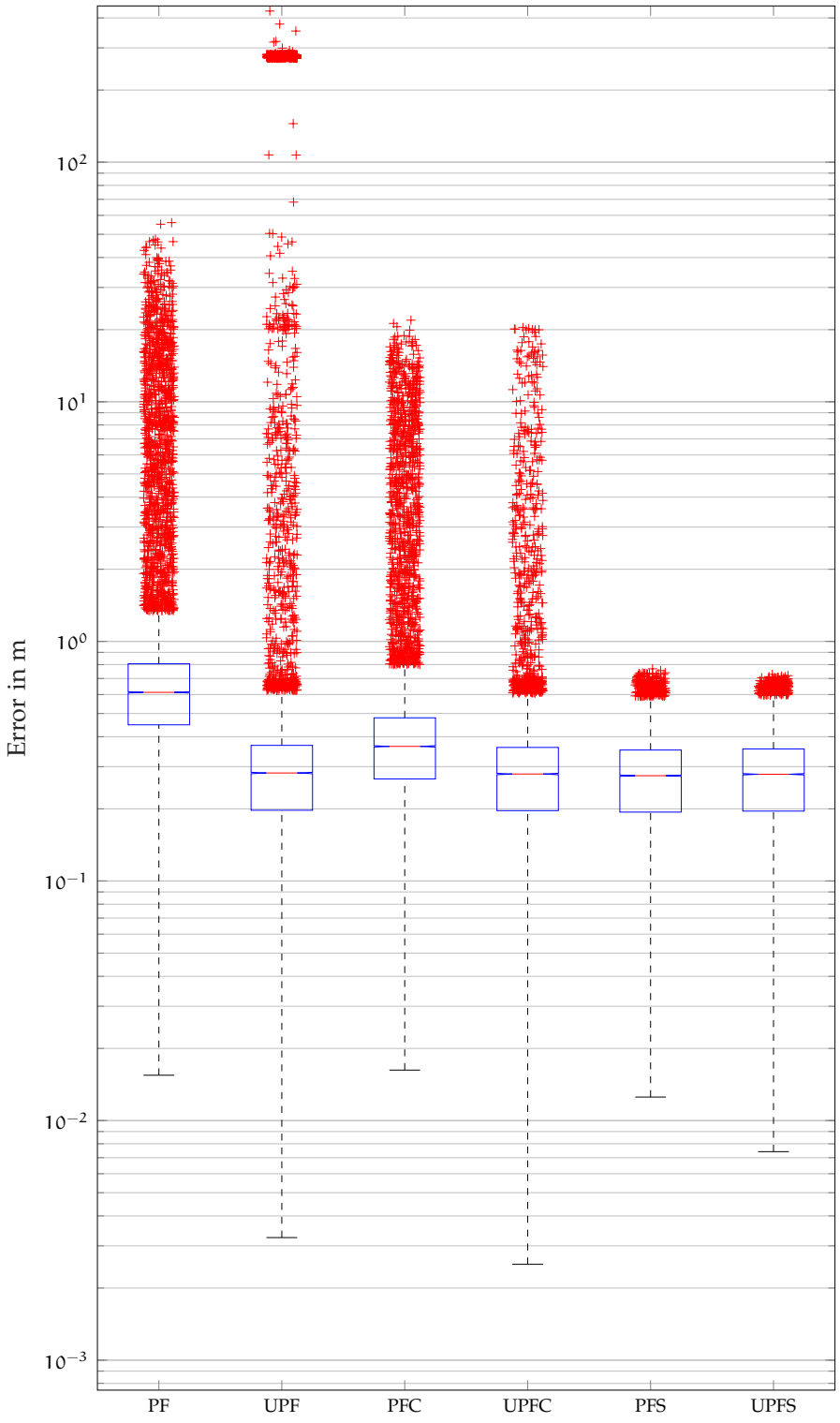


Figure 62: Box plot of the estimation errors in Scenario 2 (logarithmic scale). `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 10 particles.

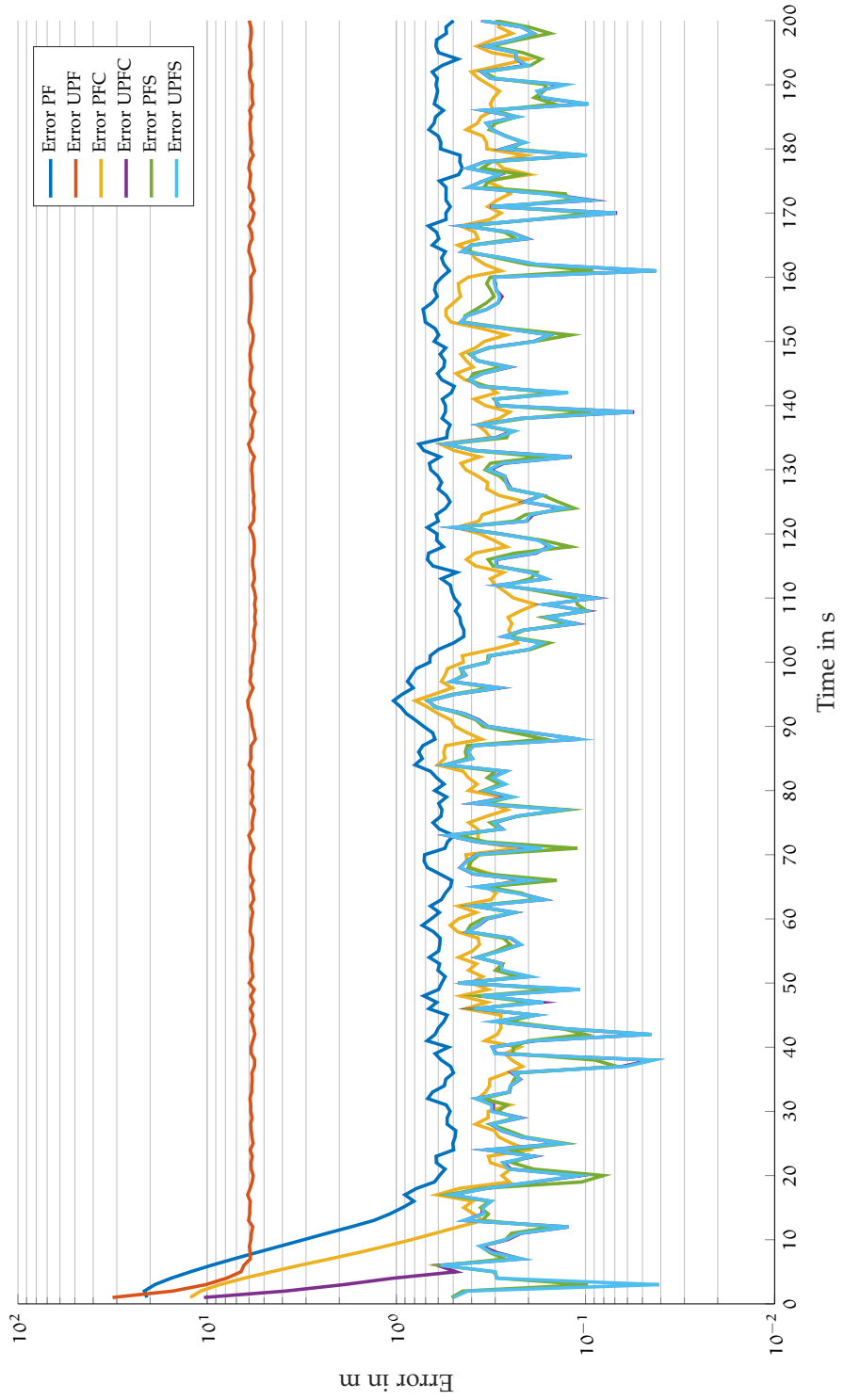


Figure 63: Mean estimation error over time in Scenario 2 (logarithmic scale). `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 10 particles.

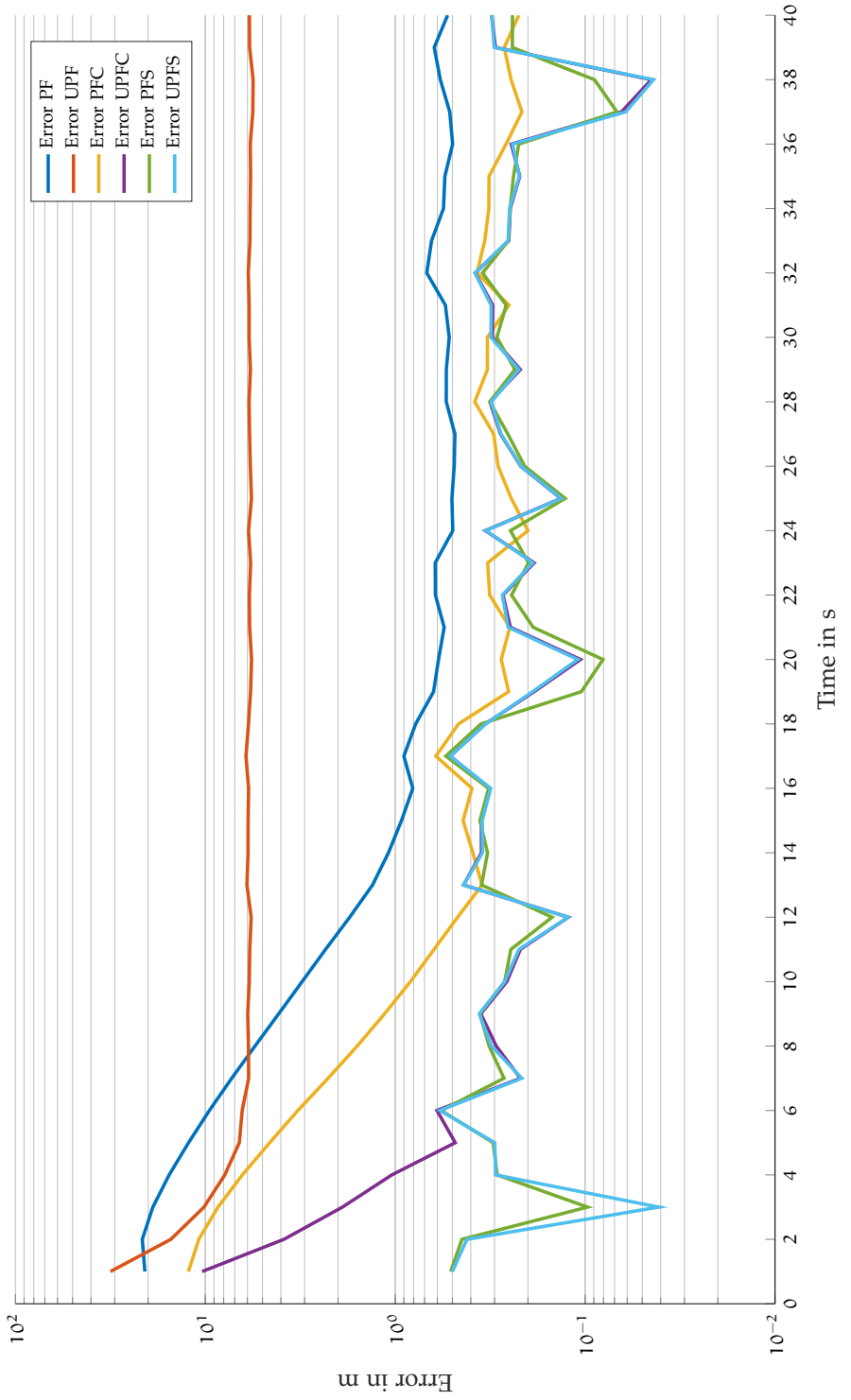


Figure 64: First 40 seconds of the mean estimation error over time in Scenario 2 (logarithmic scale). PF, PFC, PFS: 1000 particles. UPF, UPFC, UPFS: 10 particles.

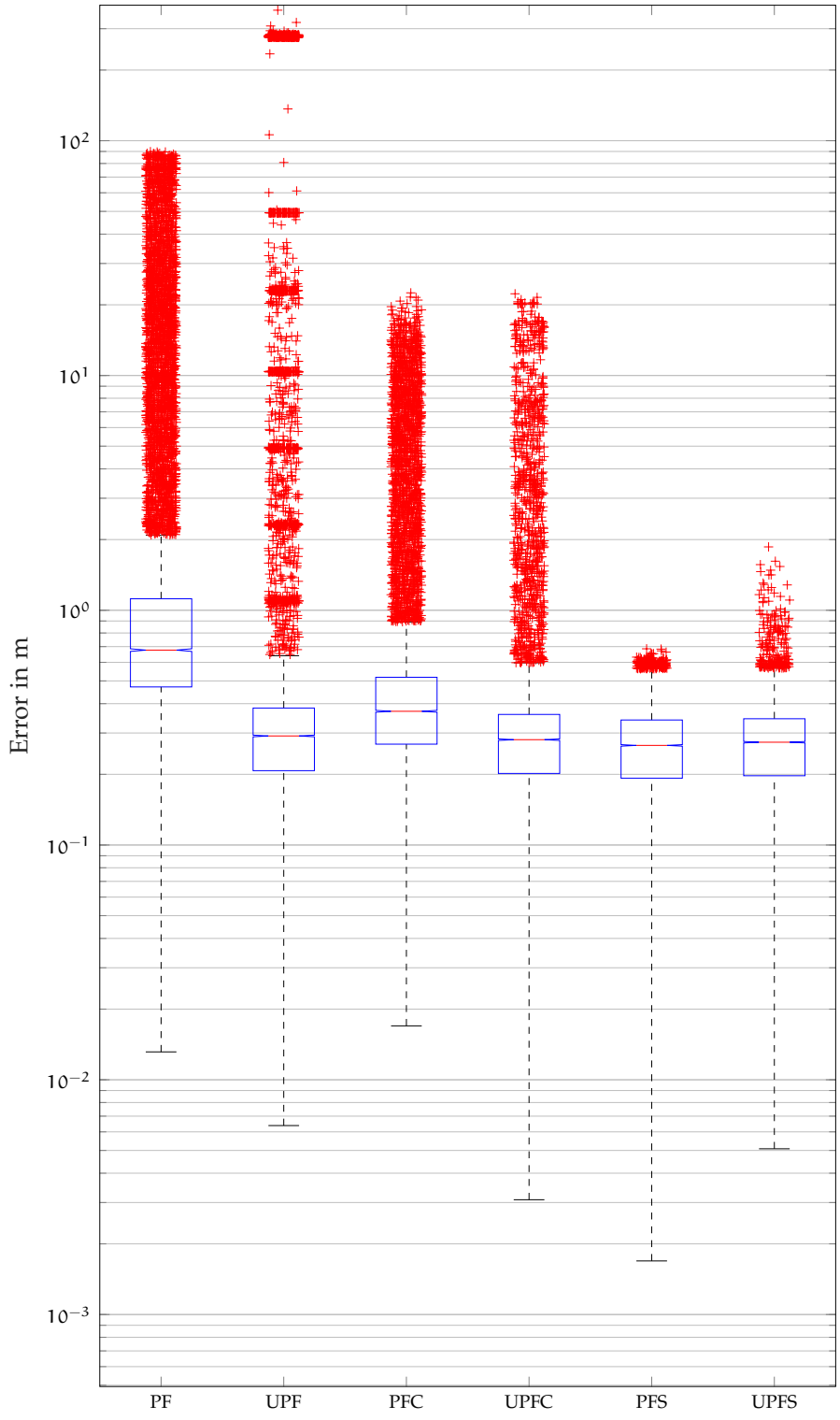


Figure 65: Box plot of the estimation errors in Scenario 2 with kidnapping (logarithmic scale). `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 10 particles.

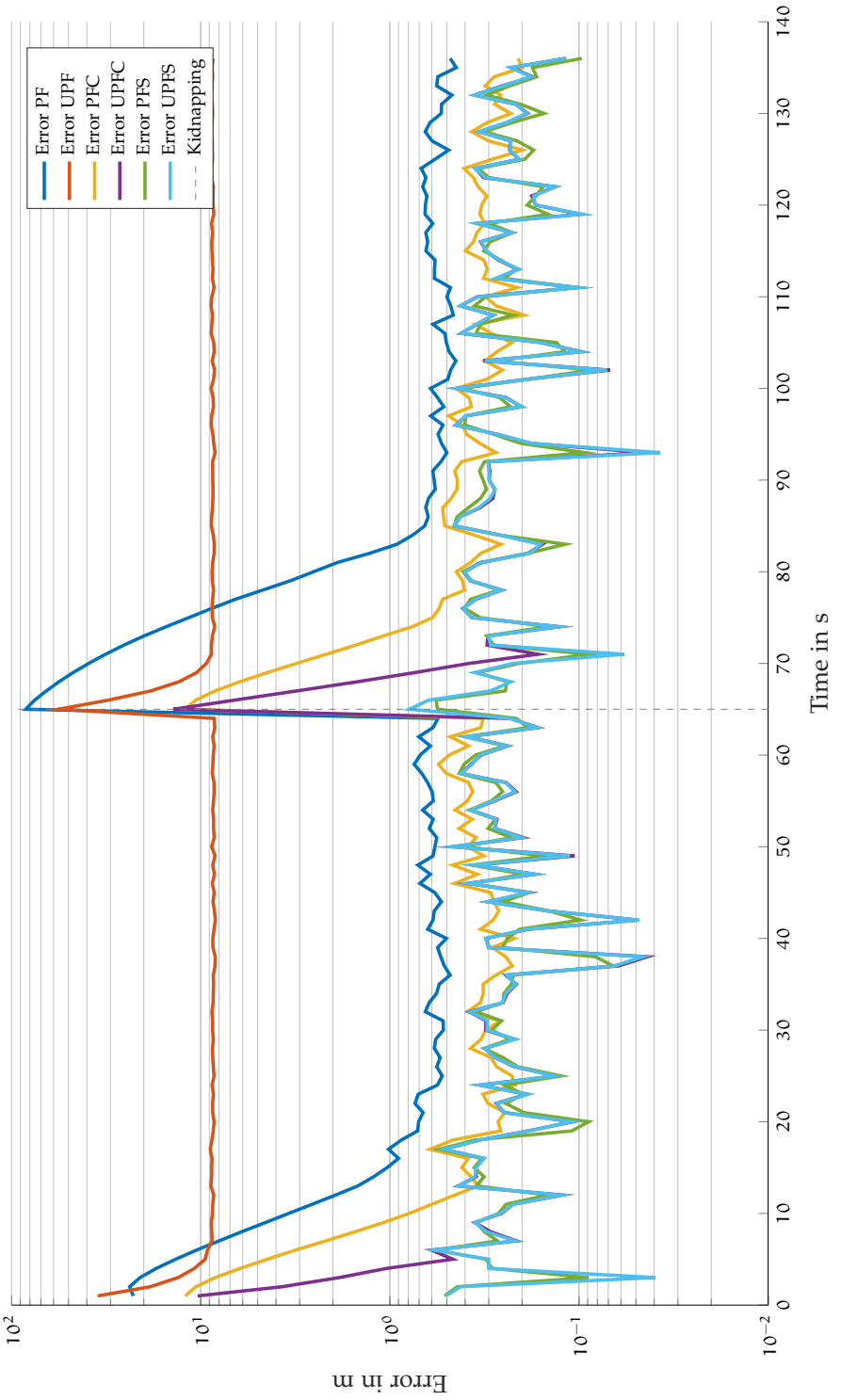


Figure 66: Mean estimation error over time in Scenario 2 with kidnapping (logarithmic scale). PF, PFC, PFS: 1000 particles. UPF, UPFC, UPFS: 10 particles.

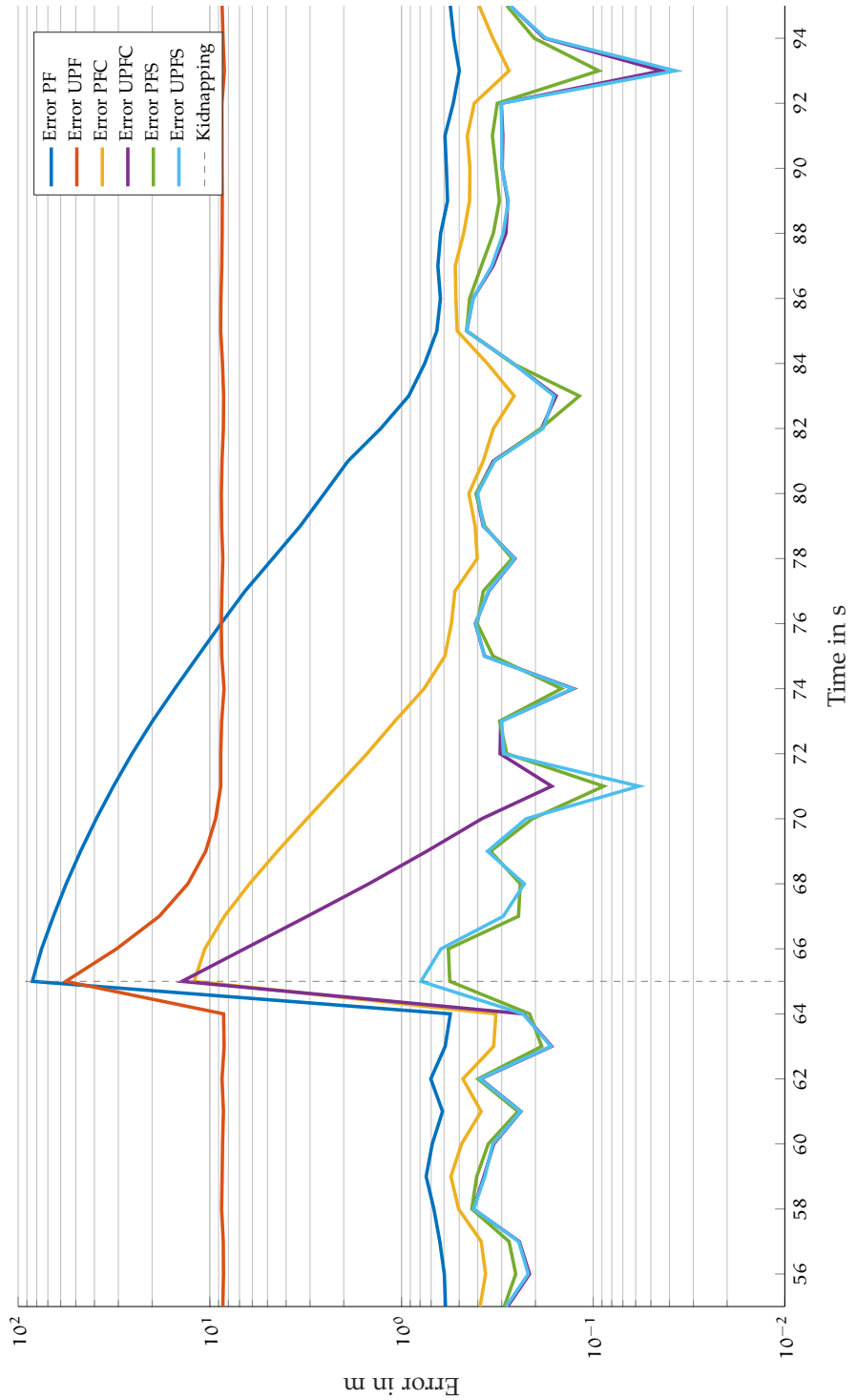


Figure 67: First 30 seconds of the mean estimation error over time after kidnapping in Scenario 2 (logarithmic scale). PF, PFC, PFS: 1000 particles. UPF, UPFC, UPFS: 10 particles.

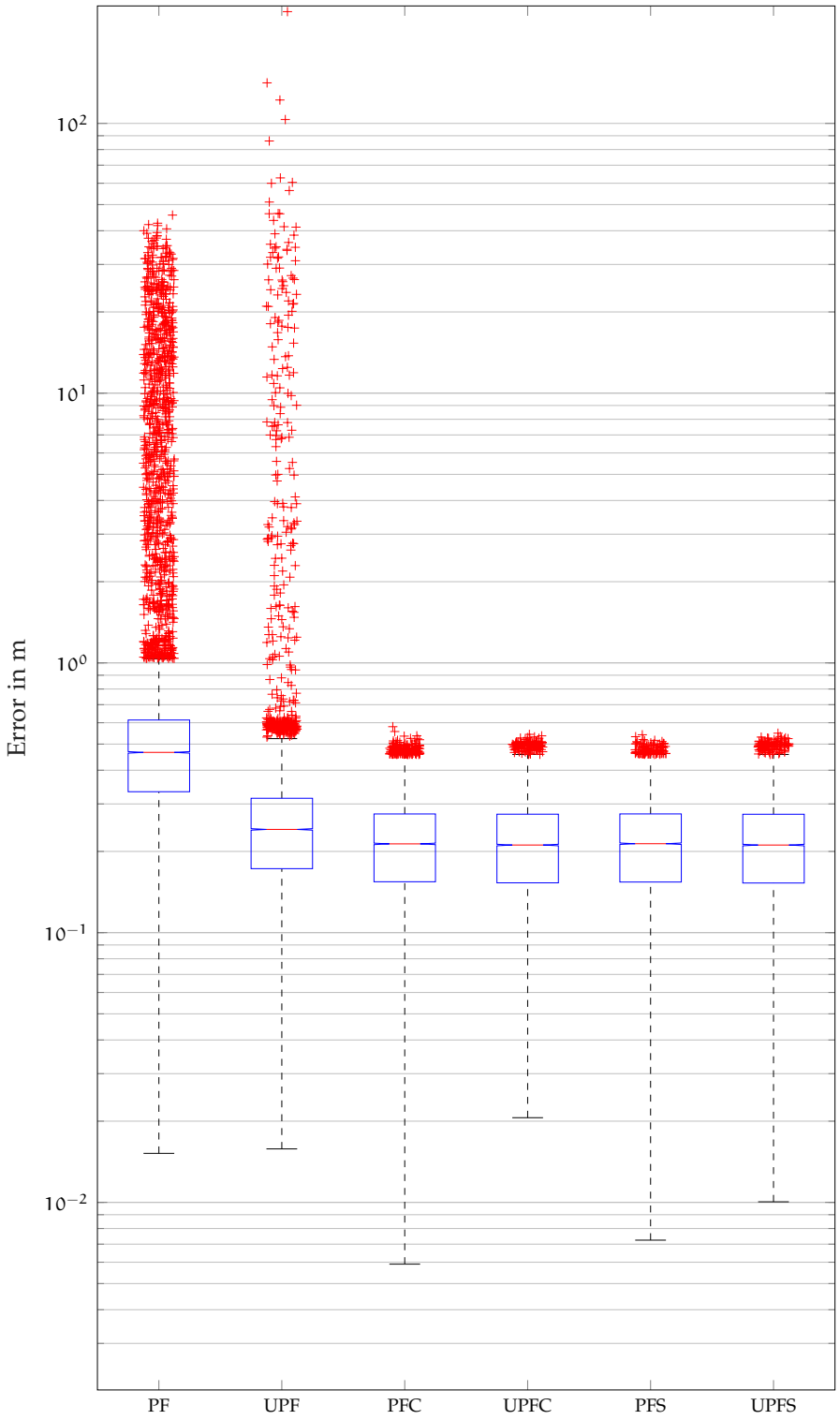


Figure 68: Box plot of the estimation errors in Scenario 3 (logarithmic scale). `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 10 particles.

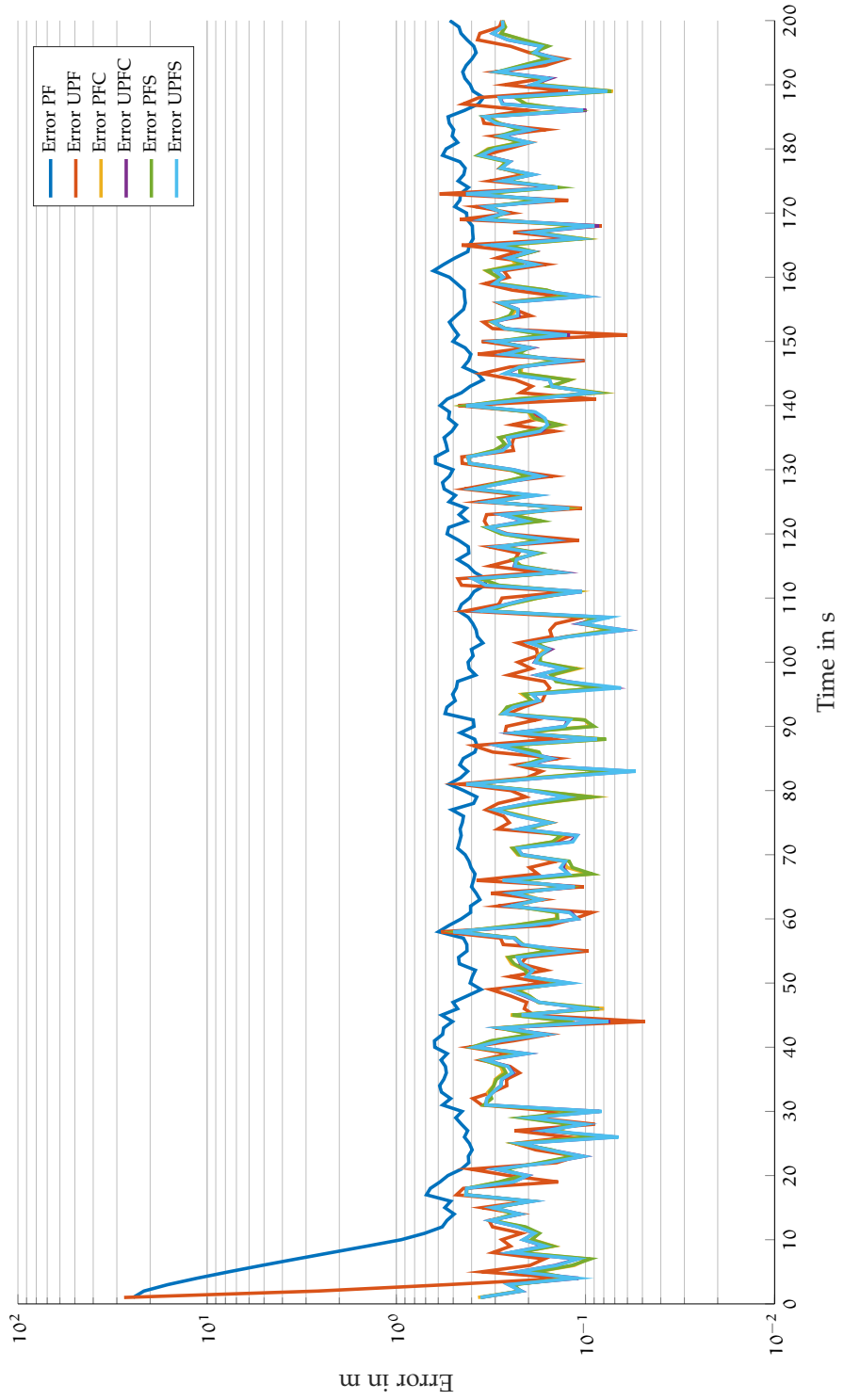


Figure 69: Mean estimation error over time in Scenario 3 (logarithmic scale). `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 10 particles.

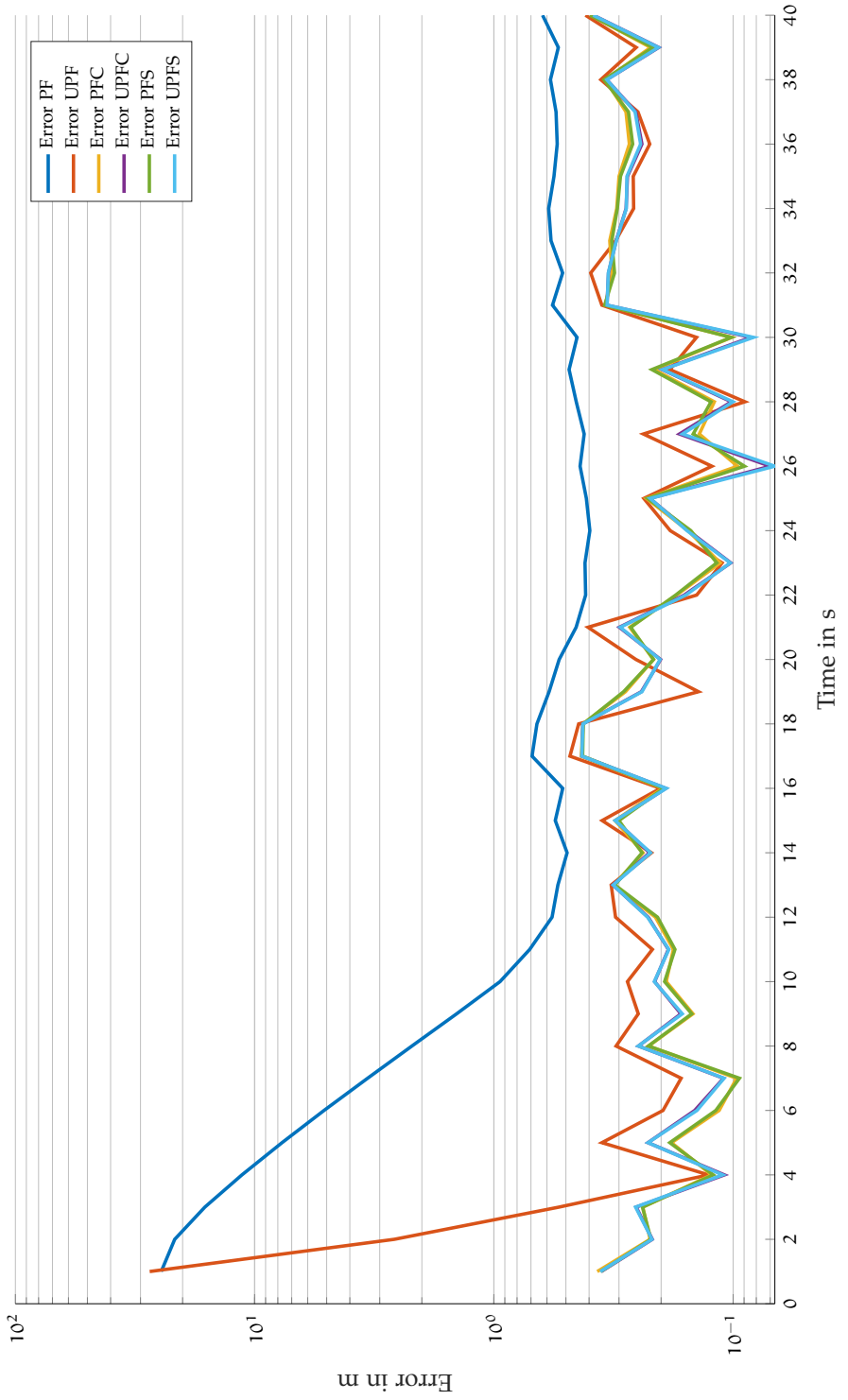


Figure 70: First 40 seconds of the mean estimation error over time in Scenario 3 (logarithmic scale). `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 10 particles.

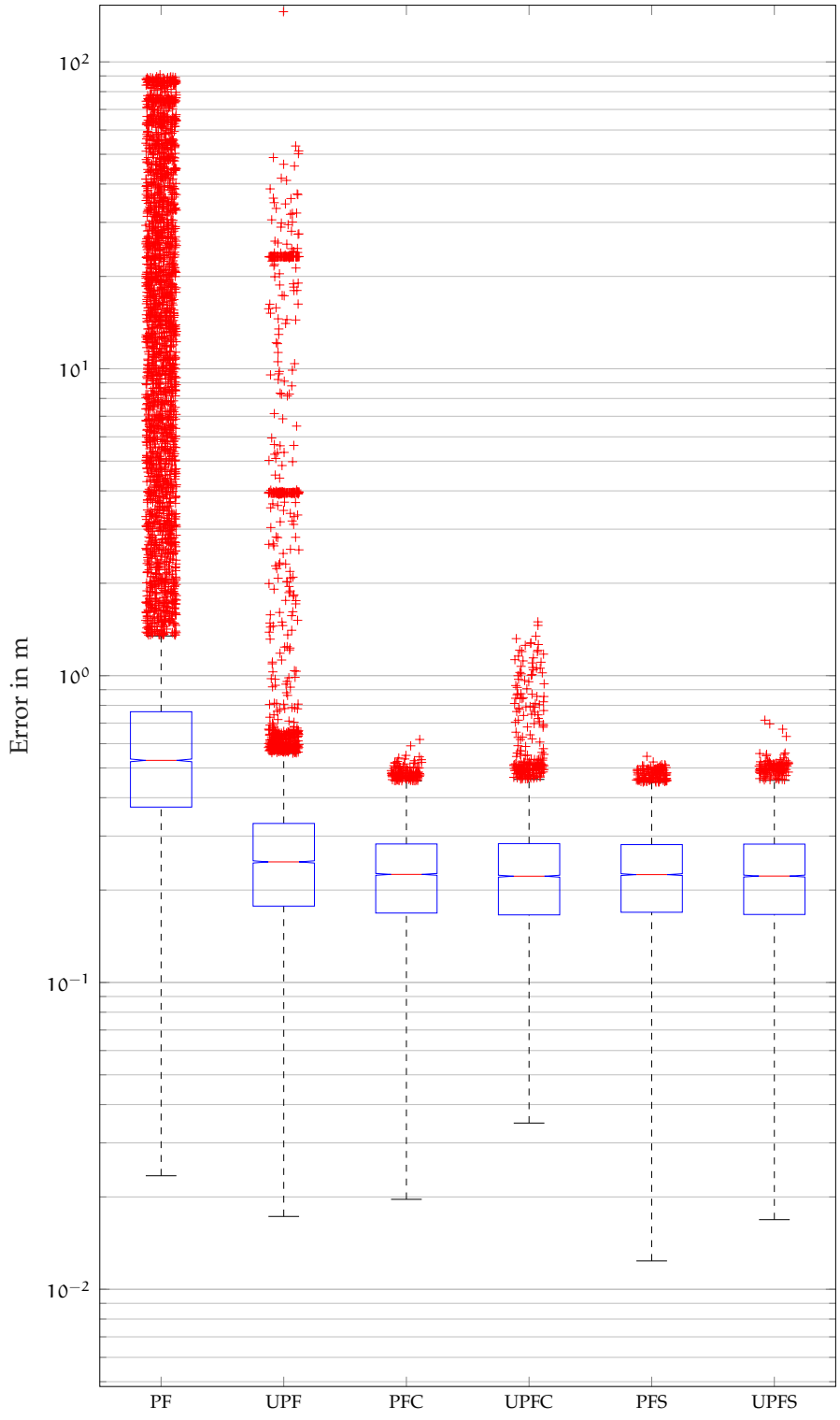


Figure 71: Box plot of the estimation errors in Scenario 3 with kidnapping (logarithmic scale). `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 10 particles.

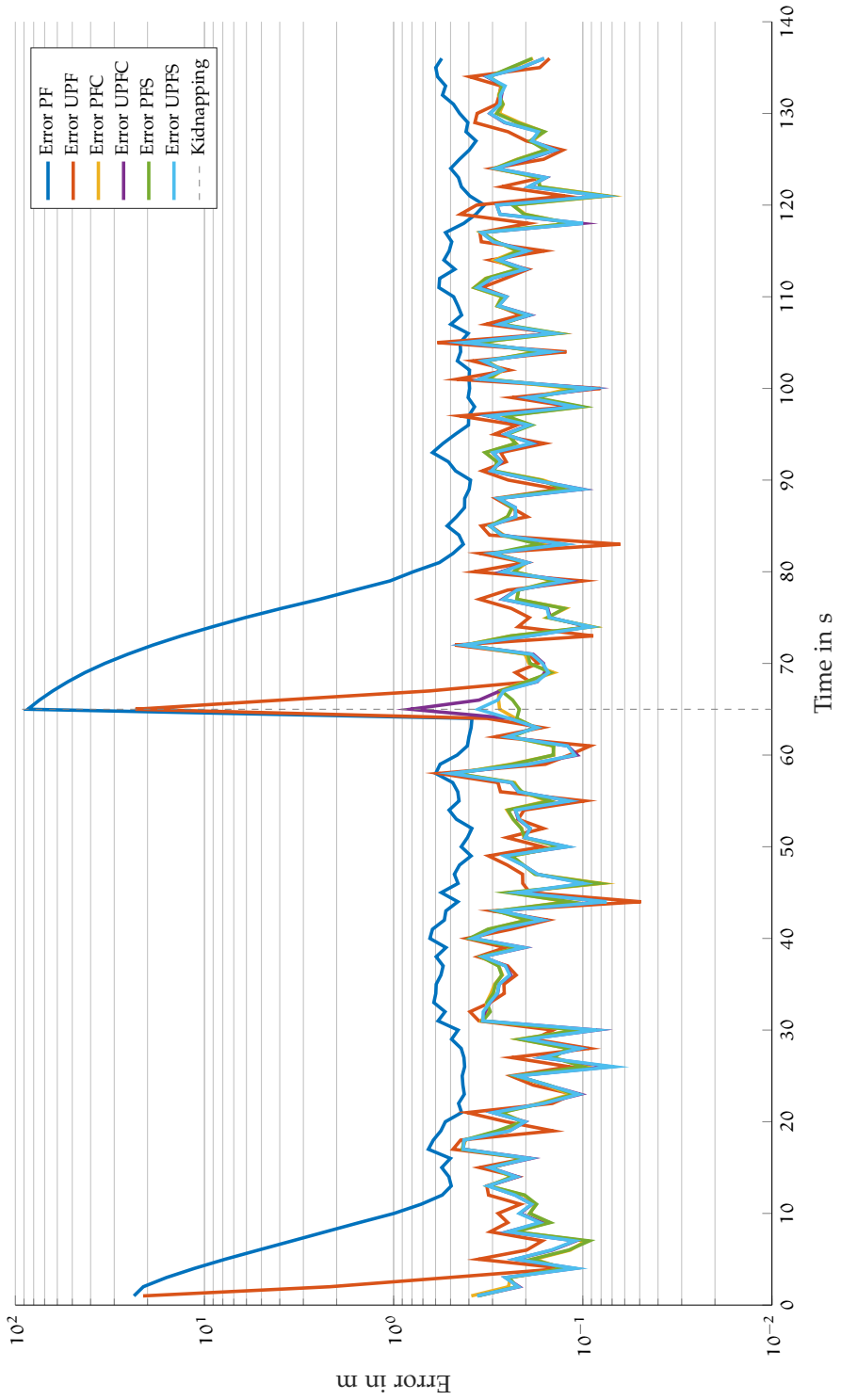


Figure 72: Mean estimation error over time in Scenario 3 with kidnapping (logarithmic scale). PF, PFC, PFS: 1000 particles. UPF, UPFC, UPFS: 10 particles.

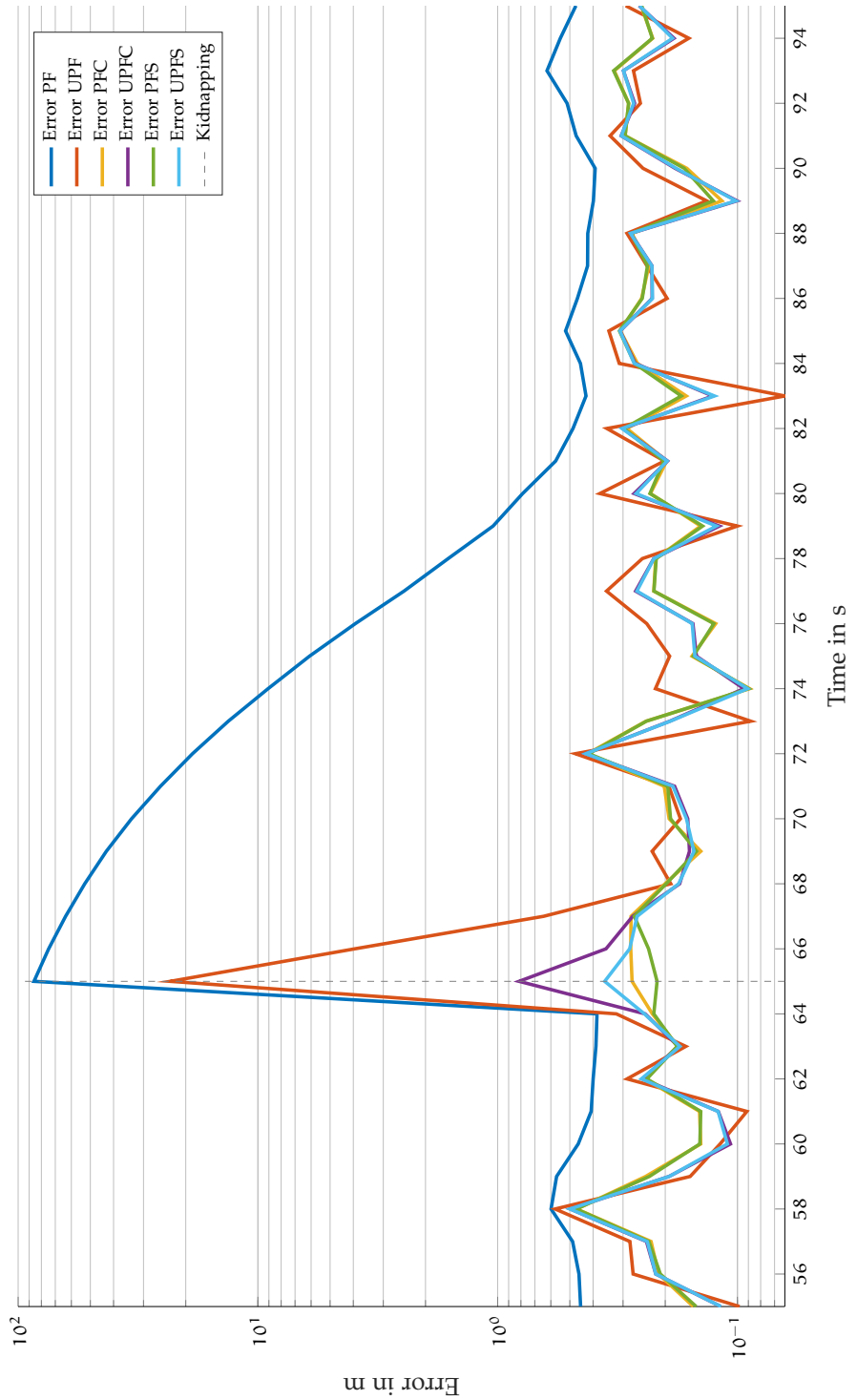


Figure 73: First 30 seconds of the mean estimation error over time after kidnapping in Scenario 3 (logarithmic scale). `PF`, `PFC`, `PFS`: 1000 particles. `UPF`, `UPFC`, `UPFS`: 10 particles.

BIBLIOGRAPHY

[1] Howie Choset, Seth Hutchinson, Kevin M. Lynch, George Kantor, Wolfram Burgard, Lydia E. Kavraki and Sebastian Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, 2005.

[2] Ulrich Nehmzow. „Mobile Robotics: Research, Applications and Challenges". In: *Future Trends in Robotics*. Institution of Mechanical Engineers, 2011.

[3] Jitendra R. Raol. *Multi-Sensor Data Fusion with MATLAB®*. CRC Press, 2009. ISBN: 9781439800058.

[4] Jingchao Zhao, Junyao Gao, Fangzhou Zhao and Yi Liu. „A Search-and-Rescue Robot System for Remotely Sensing the Underground Coal Mine Environment". In: *Sensors* 17.10 (2017). ISSN: 1424-8220. DOI: 10.3390/s17102426.

[5] A. Matos, A. Martins, A. Dias, B. Ferreira, J. M. Almeida, H. Ferreira, G. Amaral, A. Figueiredo, R. Almeida and F. Silva. „Multiple Robot Operations for Maritime Search and Rescue in euRathlon 2015 Competition". In: *OCEANS 2016 - Shanghai*. 2016, pp. 1–7. DOI: 10.1109/OCEANSAP.2016.7485707.

[6] Zoltán Beck, Luke Teacy, Alex Rogers and Nicholas R. Jennings. „Online Planning for Collaborative Search and Rescue by Heterogeneous Robot Teams". In: *International Conference on Autonomous Agents & Multiagent Systems*. AAMAS '16. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2016, pp. 1024–1033. ISBN: 978-1-4503-4239-1.

[7] Richard Voyles and Howie Choset. „Search and Rescue Robots". In: *Journal of Field Robotics* 25.1-2 (2008), pp. 1–2.

[8] Binoy Shah and Howie Choset. „Survey on Urban Search and Rescue Robots". In: *Robotics Society of Japan* 22.5 (2004), pp. 582–586.

[9] Matthew Dunbabin and Lino Marques. „Robots for Environmental Monitoring: Significant Advancements and Applications". In: *IEEE Robotics & Automation Magazine* 19.1 (2012), pp. 24–39.

[10] Guillaume Bresson, Zayed Alsayed, Li Yu and Sébastien Glaser. „Simultaneous Localization and Mapping: A Survey of Current Trends in Autonomous Driving". In: *IEEE Transactions on Intelligent Vehicles* 2.3 (2017), pp. 194–220.

[11] Rafael Vivacqua, Raquel Vassallo and Felipe Martins. „A Low Cost Sensors Approach for Accurate Vehicle Localization and Autonomous Driving Application“. In: *Sensors* 17.10 (2017), p. 2359.

[12] Boris Crnokić, Miroslav Grubisic and Tomislav Volaric. „Different Applications of Mobile Robots in Education“. In: *International Journal on Integrating Technology in Education* 6 (Sept. 2017), pp. 15–28.

[13] Nak Yong Ko, Tae Gyun Kim and Yong Seon Moon. „Particle Filter Approach for Localization of an Underwater Robot Using Time Difference of Arrival“. In: *Oceans*. IEEE. 2012, pp. 1–7.

[14] Sebastian Thrun, Dieter Fox, Wolfram Burgard and Frank Dellaert. „Robust Monte Carlo Localization for Mobile Robots“. In: *Artificial Intelligence* 128.1 (2001), pp. 99 –141. ISSN: 0004-3702. DOI: 10.1016/S0004-3702(01)00069-8.

[15] Stergios I. Roumeliotis and George A. Bekey. „Bayesian Estimation and Kalman Filtering: a Unified Framework for Mobile Robot Localization“. In: *International Conference on Robotics and Automation*. Vol. 3. 2000, 2985–2992 vol.3. DOI: 10.1109/ROBOT.2000.846481.

[16] Yong Rui and Yunqiang Chen. „Better Proposal Distributions: Object Tracking Using Unscented Particle Filter“. In: *Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 2. 2001, pp. 786–793. DOI: 10.1109/CVPR.2001.991045.

[17] Patric Jensfelt and Steen Kristensen. „Active Global Localization for a Mobile Robot Using Multiple Hypothesis Tracking“. In: *Transactions on Robotics and Automation* 17.5 (2001), pp. 748–760. ISSN: 1042-296X. DOI: 10.1109/70.964673.

[18] Fabrice Le Bars, Jan Sliwka, Luc Jaulin and Olivier Reynet. „Set-membership State Estimation With Fleeting Data“. In: *Automatica* 48.2 (2012), pp. 381–387.

[19] Dominique Meizel, Olivier Lévêque, Luc Jaulin and Eric Walter. „Initial Localization by Set Inversion“. In: *Transactions on Robotics and Automation* 18.6 (2002), pp. 966–971.

[20] Emmanuel Seignez and Michel Lambert. „Complexity Study of Guaranteed State Estimation Applied to Robot Localization“. In: *International Conference on Control, Automation, Robotics and Vision*. 2008, pp. 398–405. DOI: 10.1109/ICARCV.2008.4795552.

[21] Michel Kieffer, Luc Jaulin, Éric Walter and Dominique Meizel. „Robust Autonomous Robot Localization Using Interval Analysis“. In: *Reliable Computing* 6.3 (2000), pp. 337–362. ISSN: 1573-1340. DOI: 10.1023/A:1009990700281.

[22] Michel Kieffer, Luc Jaulin, Eric Walter and Dominique Meizel. „Guaranteed Mobile Robot Tracking Using Interval Analysis“. In: *Workshop on Application of Interval Analysis to System and Control*. 1999, pp. 347–360.

[23] Emmanuel Seignez, Michel Kieffer, Alain Lambert, Eric Walter and Thierry Maurin. „Real-time Bounded-error State Estimation for Vehicle Tracking“. In: *The International Journal of Robotics Research* 28.1 (2009), pp. 34–48.

[24] Luc Jaulin. „A Nonlinear Set Membership Approach for the Localization and Map Building of Underwater Robots“. In: *Transactions on Robotics* 25.1 (2009), pp. 88–98. ISSN: 1552-3098. DOI: 10.1109/TRO.2008.2010358.

[25] Luc Jaulin. „Localization of an Underwater Robot Using Interval Constraint Propagation“. In: *International Conference on Principles and Practice of Constraint Programming*. Springer. 2006, pp. 244–255.

[26] A. Lambert, D. Gruyer, B. Vincke and E. Seignez. „Consistent outdoor vehicle localization by bounded-error state estimation“. In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2009, pp. 1211–1216. DOI: 10.1109/IROS.2009.5354673.

[27] Renata Neuland, Jeremy Nicola, Renan Maffei, Luc Jaulin, Edson Prestes and Mariana Kolberg. „Hybridization of Monte Carlo and Set-membership Methods for the Global Localization of Underwater Robots“. In: *International Conference on Intelligent Robots and Systems*. Sept. 2014.

[28] Renata Neuland, Renan Maffei, Luc Jaulin, Edson Prestes and Mariana Kolberg. „Improving the Precision of AUVs Localization in a Hybrid Interval-Probabilistic Approach Using a Set-Inversion Strategy“. In: *Unmanned Systems* 02 (Oct. 2014), pp. 361–375.

[29] Jérémy Nicola. „Robust, precise and reliable simultaneous localization and mapping for and underwater robot. Comparison and combination of probabilistic and set-membership methods for the SLAM problem“. PhD thesis. 2017. URL: http://www.theses.fr/2017BRES0066.

[30] Immanuel Ashokaraj, Antonios Tsourdos, Peter Silson and Brian White. „Sensor Based Robot Localisation and Navigation: Using Interval Analysis and Extended Kalman Filter“. In: *Asian Control Conference*. Vol. 2. IEEE. 2004, pp. 1086–1093. DOI: 10.1109/IROS.2004.1389321.

[31] Immanuel Ashokaraj, Antonios Tsourdos, Peter Silson and Brian White. „Sensor Based Robot Localisation and Navigation: Using Interval Analysis and Unscented Kalman Filter“. In: *International Conference on Intelligent Robots and Systems*. Vol. 2. IEEE. 2004, pp. 1086–1093. DOI: 10.1109/IROS.2004.1389321.

[32] Immanuel Ashokaraj, Antonios Tsourdos, Peter Silson, Brian White and John Economou. „A Fuzzy Logic Approach in Feature Based Robot Navigation Using Interval Analysis and UKF“. In: *Fuzzy Information Processing*. Vol. 2. IEEE. 2004, pp. 808–813.

[33] Immanuel Ashokaraj, Antonios Tsourdos, Peter Silson and Brian White. „Mobile Robot Localisation and Navigation Using Multi-sensor Fusion via Interval Analysis and UKF“. In: *Towards Auto-nomous Robotic Systems* (2004).

[34] Simon Haykin. *Adaptive Filter Theory*. Pearson Education, 2013. ISBN: 9780133080360.

[35] Simon Haykin. *Neural Networks and Learning Machines*. Prentice Hall, 2009. ISBN: 9780131471399.

[36] Sebastian Thrun, Wolfram Burgard and Dieter Fox. *Probabilistic Robotics*. Intelligent Robotics and Autonomous Agents. MIT Press, 2005. ISBN: 9780262201629.

[37] Peter S. Maybeck. *Stochastic Models, Estimation, and Control*. Mathematics in Science and Engineering. Navtech GSP Supply, 2002.

[38] Rudolph Emil Kalman. „A New Approach to Linear Filtering and Prediction Problems“. In: *Transactions of the ASME–Journal of Basic Engineering* 82.Series D (1960), pp. 35–45.

[39] Gregory F. Welch. „Kalman Filter“. In: *Computer Vision*. Ed. by Katsushi Ikeuchi. Springer US, 2014, pp. 435–437. ISBN: 978-0-387-30771-8. DOI: 10.1007/978-0-387-31439-6_716.

[40] Paul Zarchan and Howard Musoff. *Fundamentals of Kalman Filtering: A Practical Approach*. Progress in Astronautics and Aeronautics. Academic Press, 2009, pp. 34–38. ISBN: 9781600867187.

[41] Mohinder S. Grewal and Angus P. Andrews. *Kalman Filtering: Theory and Practice Using MATLAB*. Wiley, 2008. ISBN: 9780470377802.

[42] Simon J. Julier and Jeffrey K. Uhlmann. „New Extension of the Kalman Filter to Nonlinear Systems“. In: *Signal Processing Sensor Fusion and Target Recognition* 3068 (1997), pp. 3068 –3068 –12. DOI: 10.1117/12.280797.

[43] Eric A. Wan and Rudolph Van Der Merwe. „The Unscented Kalman Filter for Nonlinear Estimation“. In: *Adaptive Systems for Signal Processing, Communications, and Control*. IEEE. 2000, pp. 153–158.

[44] Fredrik Gustafsson and Gustaf Hendeby. „Some Relations Between Extended and Unscented Kalman Filters“. In: *Transactions on Signal Processing* 60.2 (2012), pp. 545–555.

[45] Eric A. Wan and Rudolph Van Der Merwe. „The Unscented Kalman Filter“. In: *Kalman Filtering and Neural Networks* 5.2007 (2001), pp. 221–280.

[46] Simon J. Julier and Jeffrey K. Uhlmann. *A General Method for Approximating Nonlinear Transformations of Probability Distributions*. Tech. rep. University of Oxford, Department of Engineering Science, 1996.

[47] Simon J. Julier. „The Scaled Unscented Transformation“. In: *American Control Conference*. Vol. 6. IEEE. 2002, pp. 4555–4559.

[48] Simon J. Julier, Jeffrey K. Uhlmann and Hugh F. Durrant-Whyte. „A New Approach for Filtering Nonlinear Systems“. In: *American Control Conference*. Vol. 3. IEEE. 1995, pp. 1628–1632.

[49] Rudolph Van Der Merwe, Arnaud Doucet, Nando De Freitas and Eric A. Wan. *The Unscented Particle Filter*. Tech. rep. Cambridge University, 2000.

[50] Rudolph Van Der Merwe, Arnaud Doucet, Nando De Freitas and Eric A. Wan. „The Unscented Particle Filter“. In: *Advances in Neural Information Processing Systems*. 2001, pp. 584–590.

[51] M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon and Tim Clapp. „A Tutorial on Particle Filters for Online Nonlinear / Non-Gaussian Bayesian Tracking“. In: *Transactions on Signal Processing* 50.2 (2002), pp. 174–188. ISSN: 1053-587X. DOI: 10.1109/78.978374.

[52] Arnaud Doucet. *On Sequential Simulation-based Methods for Bayesian Filtering*. CUED/F-INFENG/TR. University of Cambridge, Department of Engineering, 1998.

[53] John Geweke. „Bayesian Inference in Econometric Models Using Monte Carlo Integration“. In: *Econometrica: Journal of the Econometric Society* (1989), pp. 1317–1339.

[54] Arnaud Doucet, Nando De Freitas and Neil Gordon. „An Introduction to Sequential Monte Carlo Methods“. In: *Sequential Monte Carlo Methods in Practice*. Springer, 2001, pp. 3–14.

[55] Elmar Garcia. „Bayes-Filter zur Genauigkeitsverbesserung und Unsicherheitsermittlung von dynamischen Koordinatenmessungen“. PhD thesis. Friedrich–Alexander–Universität Erlangen, 2012.

[56] Augustine Kong, Jun S. Liu and Wing Hung Wong. „Sequential Imputations and Bayesian Missing Data Problems“. In: *Journal of the American Statistical Association* 89.425 (1994), pp. 278–288.

[57] Neil J. Gordon, David J. Salmond and Adrian F. Smith. „Novel Approach to Nonlinear / Non-Gaussian Bayesian State Estimation“. In: *Radar and Signal Processing*. Vol. 140. 2. IET. 1993, pp. 107–113.

[58] Zhe Chen. „Bayesian Filtering: From Kalman Filters to Particle Filters, and Beyond“. In: *Statistics* 182.1 (2003), pp. 1–69.

[59] Fred Daum. „Nonlinear filters: beyond the Kalman filter“. In: *IEEE Aerospace and Electronic Systems Magazine* 20.8 (2005), pp. 57–69.

[60] Arnaud Doucet, Neil J. Gordon and Vikram Krishnamurthy. „Particle Filters for State Estimation of Jump Markov Linear Systems“. In: *Transactions on Signal Processing* 49.3 (2001), pp. 613–624.

[61] Fredrik Gustafsson. „Particle Filter Theory and Practice with Positioning Applications“. In: *Aerospace and Electronic Systems* 25.7 (2010), pp. 53–82. ISSN: 0885-8985. DOI: 10.1109/MAES.2010.5546308.

[62] David Salmond and Neil J. Gordon. *An Introduction to Particle Filters*. Tech. rep. University of Melbourne, 2006.

[63] Michael K. Pitt and Neil Shephard. „Filtering via Simulation: Auxiliary Particle Filters“. In: *Journal of the American Statistical Association* 94.446 (1999), pp. 590–599. ISSN: 01621459.

[64] Arnaud Doucet, Simon Godsill and Christophe Andrieu. „On Sequential Monte Carlo Sampling methods for Bayesian Filtering“. In: *Statistics and Computing* 10.3 (2000), pp. 197–208.

[65] Jun S. Liu and Rong Chen. „Sequential Monte Carlo Methods for Dynamic Systems“. In: *Journal of the American Statistical Association* 93.443 (1998), pp. 1032–1044.

[66] Scott Lenser and Manuela Veloso. „Sensor Resetting Localization for Poorly Modelled Mobile Robots“. In: *International Conference on Robotics and Automation*. Vol. 2. IEEE. 2000, pp. 1225–1232.

[67] D. Avitzour. „Stochastic Simulation Bayesian Approach to Multitarget Tracking“. In: *Radar, Sonar and Navigation* 142.2 (1995), pp. 41–44.

[68] Edward R. Beadle and Petar M. Djuric. „A Fast-weighted Bayesian Bootstrap Filter for Nonlinear Model State Estimation“. In: *Aerospace and Electronic Systems* 33.1 (1997), pp. 338–343.

[69] Michael Isard and Andrew Blake. „Contour Tracking by Stochastic Propagation of Conditional Density“. In: *European Conference on Computer Vision*. Springer. 1996, pp. 343–356.

[70] Genshiro Kitagawa. „Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models“. In: *Journal of Computational and Graphical Statistics* 5.1 (1996), pp. 1–25.

[71] Carlo Berzuini, Nicola G. Best, Walter R. Gilks and Cristiana Larizza. „Dynamic Conditional Independence Models and Markov Chain Monte Carlo Methods“. In: *Journal of the American Statistical Association* 92.440 (1997), pp. 1403–1412.

[72] Jon G. Rokne. „Interval Arithmetic and Interval Analysis: An Introduction“. In: *Granular Computing: An Emerging Paradigm*. Ed. by Witold Pedrycz. Heidelberg: Physica-Verlag HD, 2001, pp. 1–22. ISBN: 978-3-7908-1823-9. DOI: 10.1007/978-3-7908-1823-9_1.

[73] Luc Jaulin, Michel Kieffer, Olivier Didrit and Eric Walter. *Applied Interval Analysis: With Examples in Parameter and State Estimation, Robust Control and Robotics*. Vol. 1. Springer Science & Business Media, 2001.

[74] Eldon Hansen and G. William Walster. *Global Optimization Using Interval Analysis: Revised And Expanded*. Monographs and Textbooks in Pure and Applied Mathematics. CRC Press, 2003. ISBN: 9780203026922.

[75] Ramon E. Moore, R. Baker Kearfott and Michael J. Cloud. *Introduction to Interval Analysis*. Vol. 110. Siam, 2009. DOI: 10.1137/1.9780898717716.

[76] C. Lottaz, D. Sam-Haroud, B. Faltings and I. Smith. „Constraint Techniques for Collaborative Design“. In: *International Conference on Tools with Artificial Intelligence*. 1998, pp. 34–41. DOI: 10.1109/TAI.1998.744754.

[77] D. Sam-Haroud and B. Faltings. „Consistency Techniques for Continuous Constraints“. In: *Constraints* 1.1 (1996), pp. 85–118. ISSN: 1572-9354. DOI: 10.1007/BF00143879.

[78] Jamila Sam. „Constraint Consistency Techniques for Continuous Domains“. PhD thesis. Lausanne: IIF, 1995, p. 200.

[79] F. Rossi, P. van Beek and T. Walsh. *Handbook of Constraint Programming*. Foundations of Artificial Intelligence. Elsevier Science, 2006. ISBN: 9780080463803.

[80] Frédéric Benhamou, Frédéric Goualard, Laurent Granvilliers and Jean-François Puget. „Revising Hull and Box Consistency“. In: *Logic Programming*. MIT press, 1999, pp. 230–244.

[81] Luc Jaulin. *Interval Robotics*. Tech. rep. 2012.

[82] Elif Garajová and Martin Meciar. „Solving and Visualizing Nonlinear Set Inversion Problems“. In: *Reliable Computing* 22 (2016), pp. 104–115.

[83] Pau Herrero, Pantelis Georgiou, Christofer Toumazou, Benoît Delaunay and Luc Jaulin. „An Efficient Implementation of SIVIA Algorithm in a High-Level Numerical Programming Language“. In: *Reliable Computing* (Oct. 2012), pp. 239–251.

[84] Luc Jaulin and Eric Walter. „Set Inversion Via Interval Analysis for Nonlinear Bounded-error Estimation“. In: *Automatica* 29.4 (1993), pp. 1053–1064.

[85] Juansempere. *Wikimedia Commons File: Taitbrianzyx.svg*. [Accessed 28 July, 2018]. URL: http://commons.wikimedia.org/wiki/File:Taitbrianzyx.svg.

[86] James Diebel. *Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors*. Tech. rep. Stanford University, 2006, pp. 5–6.

[87] Blender.org. *Blender Game Engine*. [Accessed 21 August, 2018]. URL: https://docs.blender.org/manual/en/dev/game_engine/index.html.

[88] Derek Rowell. *State-Space Representation of LTI Systems*. Tech. rep. 2002.

[89] Frank L Lewis. *Applied Optimal Control & Estimation: Digital Design & Implementation*. Prentice Hall Englewood Cliffs, NJ, 1992.

[90] Emmanuel Seignez and Alain Lambert. „Complexity Study of Guaranteed State Estimation Applied to Robot Localization“. In: *International Conference on Control, Automation, Robotics and Vision* (2008), pp. 398–405.

[91] Xinguang Shao, Biao Huang and Jong Min Lee. „Constrained Bayesian State Estimation – A Comparative Study and a New Particle Filter Based Approach“. In: *Journal of Process Control* 20.2 (2010), pp. 143 –157. ISSN: 0959-1524. DOI: 10.1016/j.jprocont.2009.11.002.

[92] Y. T. Chiang, L. S. Wang, F. R. Chang and H. M. Peng. „Constrained Filtering Method for Attitude Determination Using GPS and Gyro“. In: *Radar, Sonar and Navigation* 149.5 (2002), pp. 258–264.

[93] Ondřej Straka, Jindřich Duník and Miroslav Šimandl. „Truncation Nonlinear Filters for State Estimation with Nonlinear Inequality Constraints“. In: *Automatica* 48.2 (2012), pp. 273–286.

[94] D. Simon. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Wiley, 2006. ISBN: 9780470045336.

[95] Rambabu Kandepu, Bjarne Foss and Lars Imsland. „Applying the Unscented Kalman Filter for Nonlinear State Estimation“. In: *Journal of Process Control* 18.7 (2008), pp. 753 –768. ISSN: 0959-1524. DOI: 10.1016/j.jprocont.2007.11.004.

[96] Lixin Lang, Wen-Shiang Chen, Bhavik R. Bakshi, Prem K. Goel and Sridhar Ungarala. „Bayesian Estimation via Sequential Monte Carlo Sampling – Constrained Dynamic Systems“. In: *Automatica* 43.9 (2007), pp. 1615–1622.

[97] Bruno O. S. Teixeira, Leonardo A. B. Tôrres, Luis A. Aguirre and Dennis S. Bernstein. „On Unscented Kalman filtering with State Interval Constraints“. In: *Journal of Process Control* 20.1 (2010), pp. 45–57.

[98] Xinguang Shao, Biao Huang and Jong Min Lee. „Constrained Bayesian State Estimation – A Comparative Study and a New Particle Filter Based Approach“. In: *Journal of Process Control* 20.2 (2010), pp. 143–157.

[99] „An Interval Space Reducing Method for Constrained Problems with Particle Swarm Optimization“. In: *Applied Soft Computing* 59 (2017), pp. 405 –417. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2017.05.022.

[100] S. M. Rump. „INTLAB – INTerval LABoratory“. In: *Developments in Reliable Computing*. Ed. by Tibor Csendes. http://www.ti3.tuhh.de/rump/. Dordrecht: Kluwer Academic Publishers, 1999, pp. 77–104.

[101] Mathworks. *Matlab MEX*. [Accessed 14 October, 2018]. URL: https://uk.mathworks.com/help/matlab/matlab_external/introducing-mex-files.html.

[102] GitHub. *Code and Data of the Experiments*. [Accessed 1. November, 2018]. URL: https://github.com/rob-weiss.

[103] Jeremy Nicola. *Raw MORSE Simulation Data of the Experiments*. [Accessed 23 August, 2018]. URL: https://github.com/nicolaje/reliable-slam.

[104] LAAS-CNRS. *Modular OpenRobots Simulation Engine*. [Accessed 03 August, 2018]. URL: https://www.openrobots.org/wiki/morse.

[105] Erwin Coumans. *Bullet Physics Engine*. [Accessed 27 August, 2018]. URL: www.bulletphysics.org.

[106] Luc Jaulin, Michel Legris and Frédéric Dabe. „GESMI, un logiciel pour l'aide à localisation de mines sous-marines“. In: *JIME* (2006).

[107] Á. F. Garcia-Fernandez, M. R. Morelande and J. Grajal. „Truncated Unscented Kalman Filtering“. In: *IEEE Transactions on Signal Processing* 60.7 (2012), pp. 3372–3386. ISSN: 1053-587X. DOI: 10.1109/TSP.2012.2193393.

[108] V. Drevelle and P. Bonnifait. „Robust Positioning Using Relaxed Constraint-propagation“. In: *Intelligent Robots and Systems*. 2010, pp. 4843–4848. DOI: 10.1109/IROS.2010.5649794.

ERKLÄRUNG

Ich versichere, die von mir vorgelegte Arbeit selbstständig verfasst zu haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen und ist noch nicht veröffentlicht worden. Ich bin mir bewusst, dass eine unwahre Erklärung rechtliche Folgen haben wird.

Steinfurt, 6. November 2018

Robin Weiß