Adaptive particle-aided cubature Kalman filtering for GNSS/INS-based vehicle localization

D.G.A. den Boer

Master of Science Thesis

TU Delft Delft University of Technology

Delft Center for Systems and Control

Adaptive particle-aided cubature Kalman filtering for GNSS/INS-based vehicle localization

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

D.G.A. den Boer

December 24, 2021

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

This report has been written under supervision of Royal HaskoningDHV.

Abstract

Accurate vehicle localization is considered to be a key element in future automated driving systems. A network of multiple sensors is employed to deliver information for this localization process. Loosely coupled integration of global navigation satellite systems (GNSS) and inertial navigation systems (INS) data is a common sensor-fusion method for such positioning. One of the problems of this approach, is that exact knowledge of the process- and measurement noise covariance matrices is often not available. The GNSS measurement noise uncertainties, in particular, are highly dynamic and, depending on the specific environment, might follow a non-Gaussian distribution. Since particle filter are known to be superior in non-Gaussian environments, a hybrid filtering variant is proposed: adaptive particle-aided cubature Kalman filtering. This algorithm compromises between a particle filter with kernel density estimation algorithm in periods of non-Gaussian GNSS noise, and a standard cubature Kalman filter in case of Gaussian GNSS noise. The results of GNSS/INS-based localization simulations indicate that the proposed adaptive particle-aided cubature Kalman filter outperforms traditional filtering methods in terms of minimal localization errors.

Table of Contents

Acknowledgements

During my research, I have had lots of support from various persons. First of all, I want to thank my supervisors. Peter, thank you for introducing me to Royal HaskoningDHV and enthusing me for the subject. The research took a somewhat different direction than we envisioned at the start, but you have always searched for interfaces between the companies expertises and my research. Evert, thanks for the unconditional support, passion and excitement that you have always shown. You have helped me when some tough decisions had to be made. I would like to thank Bart for always expecting high-quality, grammatically and mathematically correct documents, and for showing me the tricks of the trade of the academic world. You have taught me to prepare, host and get the most out of our efficient meetings - something that I will hopefully bring along in my future career.

Furthermore, I would like to thank my colleagues at Royal HaskoningDHV for including me in their after-work activities. I have surely enjoyed the Friday afternoon drinks at the Rotterdam office. For my fellow students at the 3ME study landscape; thanks for the numerous coffee breaks and conversations on our happy moments and our struggles.

I would like to thank my housemates Oyono, Laurens, Mark and Tijmen, as they kept asking me how my days were. We shared the burden of doing a thesis, research or residency in times all the COVID-19 despair. I could have wished for no other comrades during this period. At last, I would like to thank my parents, Gerard and Bernie, and my brothers Gijs and Rob for their support during my studies. I am proud of how we get along so good as a family.

Delft, University of Technology — D.G.A. den Boer
December 24, 2021

Glossary

List of Acronyms

GNSS	global navigation satellite system
INS	inertial navigation system
KDE	kernel density estimation
RMSE	root-mean-square error
CKF	cubature Kalman filter
PF	particle filter
CKFAPF	cubature Kalman filter-aided particle filter / cubature particle filter
IBAE-UKF	innovation-based adaptive estimation unscented Kalman filter
PACKF	adaptive particle-aided cubature Kalman filter

List of Symbols

$\boldsymbol{\mu}$	Vector with 'Gaussianity' switching parameters
$\boldsymbol{\nabla}^{\mathrm{b}}$	Accelerometer bias
$\boldsymbol{\omega}$	Rotation velocity vector
$\boldsymbol{\phi}$	Orientation error of INS
$\boldsymbol{\tau}$	Time constant
$\boldsymbol{\varepsilon}$	Gyroscope bias
$\boldsymbol{p}$	Position error of INS
$\delta\boldsymbol{\omega}$	Rotation velocity error vector
δV	Velocity error of INS
η	Tuning parameter for the instrumental matrix $\tilde{\mathbf{Q}}$
γ	Safety factor for μ
$\hat{N}_{\mathrm{eff}}$	Effective sample size
$\{\xi_i, \omega_i\}$	Cubature point
$\tilde{\mathbf{Q}}$	Instrumental matrix for process noise
$\tilde{\mathbf{R}}$	Instrumental matrix for measurement noise
$\tilde{\mathbf{y}}_{i,\mathrm{norm}}$	Normalized innovation
$\mathbf{B}$	Input transition matrix
$\mathbf{C}_{\mathrm{b}}^{\mathrm{n}}$	Rotation matrix from b-frame to n-frame
$\mathbf{C}_{sz}$	Sigma point cross covariance matrix
$\mathbf{F}$	Transition matrix
$\mathbf{f}$	Accelerometer specific force measurement
$\mathbf{H}$	Measurement matrix
$\mathbf{I}_n$	Identity matrix of size n-by-n
$\mathbf{K}$	Kalman gain
$\mathbf{P}$	Covariance matrix of states
$\mathbf{P}_{xz}$	Cross covariance matrix of states
$\mathbf{P}_{zz}$	Innovation covariance matrix
$\mathbf{Q}$	Process noise covariance matrix
$\mathbf{R}$	Measurement noise covariance matrix
$\mathbf{S}$	Square-root of covariance matrix
$\mathbf{u}$	Input vector
$\mathbf{v}$	Measurement noise vector
$\mathbf{w}$	Process noise vector
$\mathbf{x}$	Vehicle state vector at time k, determined up until time k
$\mathbf{x}^i$	State vector of the i^{th} particle
$\mathbf{X}_i$	Cubature state
$\mathbf{X}_i^*$	Cubature state propagated through transition function

$\mathbf{y}$	Output vector
$\mathbf{z}$	Observation vector
$\mathbf{Z}_i$	Cubature state propagated through measurement function
$\mathcal{X}*$	Prediction centered matrix
$\mathcal{Z}$	Measurement centered matrix
ω_z	Yaw rate
ω_z^b	Bias yaw rate
ϕ_z	yaw angle
ρ	Tuning parameter for the instrumental matrix $\tilde{\mathbf{Q}}$
σ	Standard error
$\tilde{\mathbf{y}}$	Innovation vector
a_y	Lateral acceleration
a_y^b	Bias lateral acceleration
a_x	Longitudinal acceleration
D_k	Kolmogorov-Smirnov statistic
f	Transition function
h	Height / bandwidth of kernel density algorithm
h	Measurement function
K	An arbitrary kernel
n	Window size / number of states
N_p	Number of particles
N_T	Threshold on effective sample size
$p(A \mid B)$	Probability of event A occurring, given event B is true
q	Proposal distribution particle filter
R_n	Radius
v_x	Longitudinal velocity
v_y	Lateral velocity
w^i	Weight of the i^{th} particle
x	x-position
y	y-position
μ	'Gaussianity' switching parameter for a single state

Chapter 1

Introduction

The first experiments with automated vehicles originate from the 1920s. Though a low level of autonomy was already achieved at that time, the phenomenon of fully self-driving vehicles was put away as science-fiction. As for the late 2010s, a hype burst, placing autonomous vehicles on top of most-discussed topics in technology, transport and urban planning. Carmakers were spending billions on the development of automated features in their vehicles, and the world was convinced fully self-driving cars on the roads were not that far away.

As of today, a road full of autonomous vehicles is considered to be decades away [36]. Experts do not know how fast this advanced vehicle technology will develop. There is, however, a general consensus about the tremendous potential of autonomous driving. About 94% of all serious crashes is caused by human errors [1], indicating that the number of crashes could be reduced significantly when automating driving tasks. Besides, it is expected that a road of fully automated vehicles could reduce greenhouse gas emissions of road transport by nearly 35% [53]. Considering these statistics, there sure is a lot at stake in the coming decades.

1-1 Relevance of vehicle localization

One key element keeps coming back when advancing vehicle technology; the need for accurate vehicle state estimation [63]. The states of a vehicle concern its kinematics: position and orientation, as well as their time derivatives in both linear and angular form. For higher levels of autonomy, a vehicle must be able to know its most important states, i.e. the ones that describe its location, up to decimeter level [18]. The ultimate goal of the developments in vehicle localization is crystal clear: providing a real-time, accurate location at any time, at any place and under any condition.

Ideally, all incoming information for automated vehicles is reliable and robust. In reality, complex and degraded working conditions could drastically degrade the reliability and robustness of sensor information [27]. Satellite signals can be disturbed by large objects in the surroundings of a vehicle that occlude the signal, as depicted in Fig. 1-1. Cameras do not

perform properly when the visibility is reduced due to, for example, adverse weather conditions [61]. Too dark or bright lighting conditions reduce vision-based sensor performance and snow could cover the lane markings, all preventing vehicle localization algorithms from working properly.

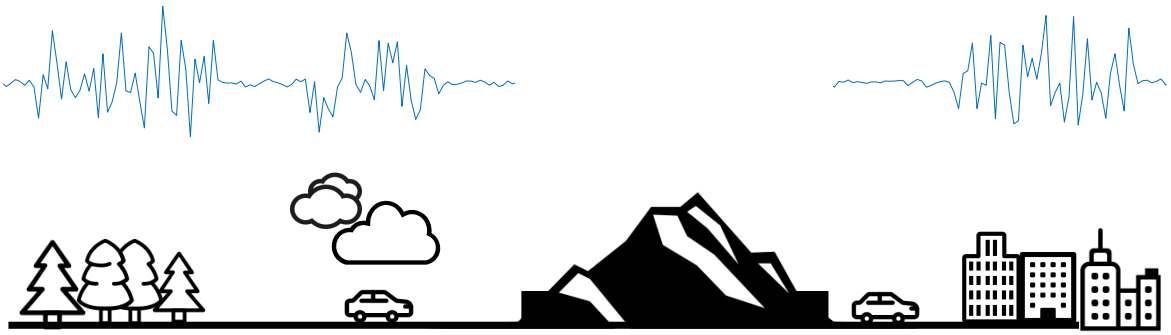

Figure 1-1: The satellite signal (blue line) of a receiver mounted on a vehicle can be disturbed by environmental factors such as trees, clouds, tunnels and urban areas. In tunnels, there might even be no reception at all. Adopted from [25].

Though some types of information sources outperform others, there is a broad consent on the fact that any stand-alone hardware system is not fit for vehicle localization on its own [38]. Each source of information has its strengths and weaknesses, which are mostly linked to accuracy, the detection range and the robustness in specific environments. Current localization techniques therefore exploit multiple sensors that complement each other [40]. Integration of global navigation satellite system (GNSS) data and inertial navigation system (INS) measurements is an example of such information fusion. These two sensors are almost exclusively present in any vehicle localization architecture.

1-2 GNSS/INS-based vehicle localization

GNSS provides autonomous, geospatial positioning on a global scale. GNSS is a generic term for satellite navigation, not to be confused with global positioning systems (GPS), which is the United States' version. Electronic receivers embedded in vehicles determine the longitude, latitude and altitude of the receiver. The receivers use transmitted time signals that are sent along a direct line by a radio on a satellite. The great advantage of GNSS signals is their global coverage. Unfortunately, GNSS signal might be lost in areas where the direct line to satellites is obstructed; think of tunnels, valleys, forests and urban areas (Fig. 1-1). Besides, for localization purposes, low-end models are often employed, which have an accuracy of approximately 5m for the cheapest models [41], or 2.5m for slightly better models [39].

An INS can determine the position of a vehicle at the hand of a set of motion sensors, rotation sensors, wheel encoders and a computer. The data comes from an inertial measurement unit (IMU), which often consists of an accelerometer, gyroscope and possibly a magnetometer. Since the motion sensors capture the relative motion with respect to the previous measurement, a so-called dead-reckoning method is used for determining a location. Dead-reckoning

uses speed and heading estimates as well as time measurements to determine a current position. Odometry is a sub-category of dead-reckoning that is used in a context with wheels; the change in rotation of the wheels is used to determine the velocity. A standard INS does not operate with odometry, but with accelerometer and gyroscope measurements.

Data from GNSS and INS is considered to be a golden combination for sensor fusion, since the properties of these data sources are opposing in some sense. INS location estimates encounter problems of accumulating errors over longer distances, thereby failing in long term navigation operations [37]. The integration of GNSS can correct this error accumulation in intervals, thereby limiting the localization errors. Besides, the smooth INS location estimates are often used instead of the noisy GNSS signal. It results in both smooth and accurate location estimates.

1-3 Filtering methods

GNSS/INS-based vehicle localization falls in the category of stochastic filtering problems. Such problems consider the case of state estimation based on a set of inaccurate and noisy observations or measurements. In other words, filtering methods attempt to find the best estimate for the current state, given observations up to that time. The development of the states is, in case of GNSS/INS-based vehicle localization, often represented by a nonlinear discrete state-space model:

$$\begin{aligned} \mathbf{x}_k &= f\left(\mathbf{x}_{k-1}, \mathbf{u}_k, \mathbf{w}_k\right) \\ \mathbf{y}_k &= h\left(\mathbf{x}_k, \mathbf{v}_k\right) \end{aligned} \tag{1-1}$$

In this equation, subscript k is a time indicator, $\mathbf{x}$ denotes the state vector, $\mathbf{u}$ is the input vector, $\mathbf{y}$ is the output vector and $\mathbf{w}_k$ and $\mathbf{v}_k$ are noise sequences with unknown statistics. The first equation indicates the state transition by function f. The probability that a current state takes the value $\mathbf{x}_k$, given the previous state $\mathbf{x}_{k-1}$ and input $\mathbf{u}_k$, is denoted as $p\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k\right)$. The second equation of (1-1) is the measurement function h. In the context of filtering, an observation of an output $\mathbf{y}_k$ at time k is denoted as $\mathbf{z}_k$. The measurement function h describes the probability $p\left(\mathbf{z}_k \mid \mathbf{x}_k\right)$ that a measurement takes value $\mathbf{z}_k$ given state $\mathbf{x}_k$.

The objective of filtering is to find the optimal state, i.e. the best estimate of $\mathbf{x}$ at time k. This comes down to, given initial probability density $p\left(\mathbf{x}_0\right)$, measurement likelihood $p\left(\mathbf{z}_k \mid \mathbf{x}_k\right)$ and prior state transition probability density $p\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k\right)$, finding the posterior density $p\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{z}_k\right)$.

1-3-1 Kalman filtering

Rudolf Kalman proposed a new type of linear filtering, called Kalman filtering, in the 1960s [34]. Kalman filters are based on the recursive Bayesian prediction-update framework [10]. Fig. 1-2 provides an example of the phases of this framework for Gaussian distributed process- and measurement noise. Given a previous estimate and knowledge about the modelling uncertainties, the prediction phase produces a predicted distribution, as indicated by the red Gaussian of Fig. 1-2. Then, a filter uses a measurement and, with some a priori knowledge of the uncertainty of this measurement, constructs a distribution around this measurement.

This step is indicated by the purple Gaussian of the center graph of Fig. 1-2. At last, a weighted average is taken over the two distributions, in which the weights are determined by the uncertainty of the two quantities. This results in the orange distribution of the right graph of Fig. 1-2.

A Kalman filter calculates an optimal estimate when the system is linear, and the noise distributions are Gaussian. Unfortunately, many systems in real-life are nonlinear, and suffer from non-Gaussian process- and measurement noise. To solve the problem of nonlinearity, some nonlinear variants of the Kalman filter have been developed. The most common ones are; i) the extended Kalman filter [64], ii) the unscented Kalman filter ([33], [73]) and iii) the cubature Kalman filter [8]. Extended Kalman filters linearize a system around the current estimates of the mean and covariance, after which they proceed as a standard Kalman filter. An unscented Kalman filter uses a deterministic set of sigma points around the mean. The points have the property that if they are propagated through a function, the mean and covariance of these sigma points accurately approximates the true mean and covariance after propagation. Cubature Kalman filters work similarly, but the set of points (cubature points) is determined differently, and there is one sample less needed compared to unscented Kalman filtering.

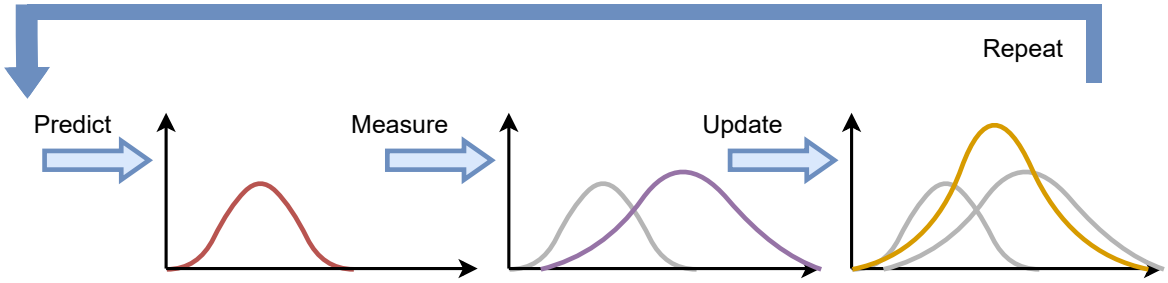

Figure 1-2: The recursive Bayesian prediction-update framework for Gaussian process- and measurement noise. A weighted average between the prediction (red) and measurement (purple) results in a filtered estimate (orange). Adopted from [2].

1-3-2 Particle filtering

Particle filtering [26] is fundamentally different from any Kalman filtering approach. Particle filters are based on sequential Monte Carlo approximations. Monte Carlo sampling is an efficient sampling technique for state estimation.

Initially, the space of possible states is represented by N_p particles, denoted as $\mathbf{x}_k^i$. The space is filled at the hand of some probability measure; high particle density around some states indicates a high probability of that state being the true one. Consecutively, one lets each particle develop through time according to the corresponding dynamic state-space. The posterior distribution is then estimated through a weighted average of all particles.

After some iteration, it can happen that all the weight is put on a few particles. Large disparities in the weights can eventually lead to all but one weights being zero. As a consequence, the algorithm degenerates and the works only for a few iterations [11]. To overcome this problem, particles with very low weights are replaced by ones that lay close to the particles with higher weights after each iteration. Various methods exist for this resampling step, more

details on this can be found in [42]. For more general information on the current generation of filtering methods, the interested reader could refer to an excellent survey on recent trends in Gaussian filtering [5].

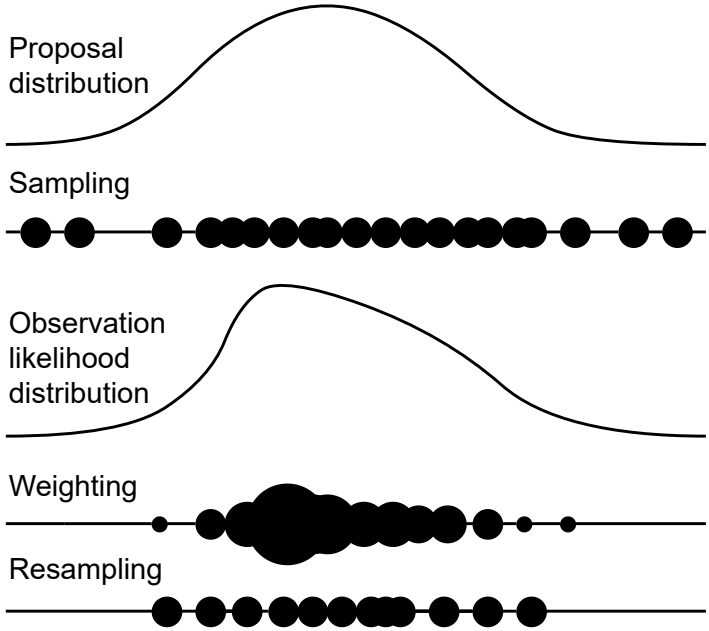

Figure 1-3: Sampling-importance-resampling process (SIR). From the initial (proposal) distribution, particles (the black dots), are sampled. They are weighted according to their probability w.r.t. the measurement likelihood. At last, particles with low weights are resampled from the proposal distribution. Adopted from [4].

1-4 Summary and structure of this thesis

Vehicle localization is an essential prerequisite for almost all automated features in intelligent vehicles. Therefore, the new generations of vehicles are equipped with many sensors that can provide real-time positioning data. To guarantee an accurate position estimate at all times, at any place and under any condition, information from multiple sensors is necessary. The fused location estimate of GNSS and INS is considered to be an essential contributor to current localization methods. This fusion step is often performed by filtering methods, such as Kalman filtering algorithms and particle filters. This thesis features a proposition for a new hybrid filter for GNSS/INS-based vehicle localization.

Chapter 2 will discuss common integration strategies for GNSS and INS, and provides a thorough mathematical description of the exploited system architecture. Besides, the current state of the art will be examined. Then, Chapter 3 contains a proposition for a new type of hybrid filtering. The proposed algorithm will be evaluated in Chapter 4, where the performance for GNSS/INS-based localization simulations will be compared to existing filtering methods. Chapter 5 contains the concluding remarks of this thesis.

Chapter 2

Modelling the localization process

There are many possible configurations for modelling the motions of a vehicle. These can be categorized based on the assumptions and simplifications that are made. As such, a model could assume constant velocity, constant acceleration, constant turn rate, constant curvature, and so on [60]. More advanced models take additional factors into account, such as dynamics (vertical forces, tire slip) of each individual wheel [62]. Luckily, fusion of global navigation satellite system (GNSS) and inertial navigation system (INS) does not require such advanced vehicle modelling. The main challenge is to transform the accelerometer and gyroscope measurements into valid location estimates.

Three different models types will be used in this chapter; i) a model for generating ground-truth data (true location data), which will be discussed in Section 2-1, ii) GNSS- and INS sensor error models for generating sensor measurement data (Section 2-2), and iii) a model for GNSS/INS fusion, as will be explained in Section 2-3. With the ground-truth and sensor measurement data, we aim to realistically validate filtering-based localization methods in later chapters.

For all the models, simulations and results of this work, the MATLAB R2021a software environment will be used. The simulations are performed on an HP ZBook Fury 15 G7 with an Intel core I7 vPRO processor.

2-1 Vehicle model for ground-truth data acquisition

In the state-space model for ground-truth data acquisition, the state vector $\mathbf{x}$ is defined as follows:

$$\mathbf{x} = \begin{bmatrix} x & y & \phi_z & v_x & v_y & \omega_z & a_y & \omega_{z,b} & a_{y,b} \end{bmatrix}^T \tag{2-1}$$

In this vector, x indicates the longitudinal position in global coordinates, y represents the lateral position in the global coordinate frame, ϕ_z is the yaw of the vehicle, v_x is the longitudinal velocity, v_y is the lateral velocity, a_y is the lateral acceleration, ω_z is the yaw rate, a_y^b is the

bias of the acceleration measurements and w_z^b is the bias of the yaw rate measurements. The model uses the input $\mathbf{u}_k = \begin{bmatrix} \delta_k & a_{x,k} \end{bmatrix}^T$ at time step k, in which δ is the steering angle and $a_{x,k}$ is the longitudinal acceleration. Apart from x, y, and ϕ_z, all states of the state vector are given in the body frame of the given vehicle. These axis configurations will be discussed in more detail later on in Subsection 2-3-1.

For the acquisition of ground-truth data, we use an Euler approximation of a model adopted from [30]. This state transition model takes the following form:

$$\mathbf{x}_{k+1} = \begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \phi_{k+1} \\ v_{x,k+1} \\ v_{y,k+1} \\ \omega_{z,k+1} \\ a_{x,k+1} \\ a_{y,k+1} \\ \omega_{b,k+1} \\ a_{y,b,k+1} \end{bmatrix} = \begin{bmatrix} x_k + v_{x,k} \cdot \Delta t \cdot \cos(\phi_k) - v_{y,k} \cdot \Delta t \cdot \sin(\phi_k) \\ y_k + v_{x,k} \cdot \Delta t \cdot \sin(\phi_k) + v_{y,k} \cdot \Delta t \cdot \cos(\phi_{z,k}) \\ \phi_{z,k} + \Delta t \cdot (\omega_{z,k} - \omega_{b,k}) \\ v_{x,k} + \Delta t \cdot (a_{x,k}) \\ v_{y,k} + \Delta t \cdot (a_{y,k} - a_{y,b,k}) \\ \omega_{z,k} + \Delta t \cdot (\frac{\beta_4}{v_{x,k}} v_{y,k} + (\frac{\beta_5}{v_{x,k}}) \omega_{z,k} + \beta_6 \delta) \\ \frac{\beta_1}{v_{x,k}} v_{y,k} + (\frac{\beta_2}{v_{x,k}} - v_{x,k}) \omega_{z,k} + \beta_3 \delta \\ -\frac{1}{\tau_\omega}\, \omega_{b,k} \\ -\frac{1}{\tau_a}\, a_{y,b,k} \end{bmatrix} + \mathbf{w}_k \tag{2-2}$$

In the last two equations, τ_ω and τ_a are the correlation time constants of the gyroscope and accelerometer biases, respectively. The parameters β_i of the process model can be calculated with:

$$\beta_i \begin{cases} \beta_1 = \frac{-2(\mathrm{C}_{\alpha \mathrm{f}} + \mathrm{C}_{\alpha \mathrm{r}})}{m}, & \beta_2 = \frac{2(-\mathrm{C}_{\alpha \mathrm{f}} l_\mathrm{f} + \mathrm{C}_{\alpha \mathrm{r}} l_\mathrm{r})}{m} \\ \beta_3 = \frac{2\mathrm{C}_{\alpha \mathrm{f}}}{m}, & \beta_4 = \frac{2(-\mathrm{C}_{\alpha \mathrm{f}} l_\mathrm{f} + \mathrm{C}_{\alpha \mathrm{r}} l_\mathrm{r})}{I_\mathrm{f}} \\ \beta_5 = \frac{-2\left(\mathrm{C}_{\alpha \mathrm{f}} l_\mathrm{f}^2 + \mathrm{C}_{\alpha \mathrm{r}} l_\mathrm{r}^2\right)}{m}, & \beta_6 = \frac{2\mathrm{C}_{\alpha \mathrm{f}} l_\mathrm{f}}{I_z} \end{cases} \tag{2-3}$$

in which m indicates the vehicle mass, l_f and l_r indicate the distance (front and rear) of the wheel axis to the center of gravity of the vehicle. $\mathrm{C}_{\alpha \mathrm{f}}$ and $\mathrm{C}_{\alpha \mathrm{r}}$ indicate the cornering stiffness of the front- and rear axis.

Note that this model does not consider any vertical motions of the vehicle. This means that the vehicle moves on a flat surface, and that roll and pitch motions are neglected. Such simplifications make sure we avoid needles complexity; adding more degrees of freedom does not contribute to a clear demonstration of the proposed hybrid filter.

Different sets of inputs $\mathbf{u}$ will be used to generate ground-truth data. This data is generated without any process or measurement noise.

2-2 Sensor error modelling for acquisition of measurement data

Sensors attempt to capture the true states of a system. Unfortunately, errors are introduced in this process. Modelling sensors comes down to modelling the errors that arise when using these sensors. These errors are known as measurement noise. In our work, it is assumed that these errors are additive.

The GNSS measurement data is obtained by artificially adding measurement noise to the ground-truth data of the previous section. A second simulation with model (2-2) will be performed to obtain INS measurements. Logically, the process noise $\mathbf{w}_k$ is now nonzero. From this second simulation, at each time step, the first 8 states of (2-2) are extracted. These will be used as INS measurement data. This process of noise injection is a widely adopted tool for simulating noise in order to validate localization algorithms [20, 48, 55].

2-2-1 GNSS measurement data

In our case, GNSS measurement are obtained by contaminating the corresponding ground-truth states with an additive measurement noise sequence $\mathbf{v}_k$. Following [30], it is assumed that the GNSS device can obtain both position and velocity measurements. The GNSS observations are then generated as:

$$\mathbf{z}_k^{\text{GPS}} = \begin{bmatrix} x_k^{\text{GPS}} \\ y_k^{\text{GPS}} \\ v_{x,k}^{\text{GPS}} \\ v_{y,k}^{\text{GPS}} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \\ v_{x,k} \\ v_{y,k} \end{bmatrix} + \mathbf{v}_k \tag{2-4}$$

2-2-2 INS measurement data

Errors in the INS are modelled as either standard additive noise or bias. The former is often modelled as Gaussian white noise. The latter occurs when the accelerometer and gyroscope outputs are integrated over time to produce velocity and yaw estimates, respectively. In this process, the velocity and yaw errors will increase proportionally with time.

INS data is obtained differently than the GNSS data of the previous subsection, since a random error build up affects the modelled dynamics in the INS along the way. The simulation has to be performed again, now with nonzero process noise $\mathbf{w}$. Due to the augmentation with the bias states $\omega_{b,k}$ and $a_{y,b,k}$, one can model such errors that build up over time.

Let it be noted that the longitudinal acceleration $a_{x,k}$ is an input that we have full knowledge of. A sequence of INS measurements can then be obtained:

$$\mathbf{z}_k^{\text{INS}} = \begin{bmatrix} x_k^{\text{INS}} \\ y_k^{\text{INS}} \\ \phi_k^{\text{INS}} \\ v_{x,k}^{\text{INS}} \\ v_{y,k}^{\text{INS}} \\ \omega_{z,k}^{\text{INS}} \\ a_{x,k}^{\text{INS}} \\ a_{y,k}^{\text{INS}} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \\ \phi_k \\ v_{x,k} \\ v_{y,k} \\ \omega_{z,k} \\ a_{x,k} \\ a_{y,k} \end{bmatrix} + \mathbf{w}_k \tag{2-5}$$

2-3 Loosely coupled GNSS/INS integration

When combining information from a GNSS and an INS, there are four common strategies: uncoupled [69], loosely coupled [65], tightly coupled [43] and ultra tightly coupled [9]. In uncoupled integration, the INS location estimate is reset when there is a GNSS signal available. There is no significant performance enhancement when using this technique, except for an increased robustness. Loosely coupled methods use the GNSS speed- and position measurements as observations of a Kalman filter that models INS error dynamics. Similarly, tightly coupled integration methods use pseudorange and Doppler measurements (i.e. raw GNSS signals) in the prediciton step of a Kalman filter. The observations come from the INS. In ultra tightly coupled methods, the signal from the central GNSS processor is used to control the code and carrier signals for each satellite, by means of digital tracking loops. In tightly coupled methods, this tracking loop is performed separately for each satellite, leading to higher vulnerability to jamming and unknown vehicle dynamics [23].

Loosely coupled integration strategies are considered to be the most intuitive. Besides, the GNSS data can be obtained easily compared to tightly coupled methods. The observations vector of (2-4) can simply be used. Therefore, the remainder of this report proceeds on a loosely coupled strategy. Filters of the Kalman family operate with a prediction step and an update step. The predictions are provided by the process model, which is based on INS data. The update step is based on GNSS measurements.

2-3-1 Process model

In literature, the process model for loosely coupled integration is implemented in various ways, with different notations and assumptions. The model in this subsection contains elements from [6], [28], [49] and [66].

As stated earlier, it is assumed there is no change in the z-coordinate of the vehicle. These states are still included in the process, but are set to zero after simulating the process model.

Several axis-frames are used in the localization process. The local navigation frame is denoted as the n-frame, with ENU (East-North-Up) directions. The global frames are the geocentric inertial frame (i-frame) and earth-centered earth-fixed frame (e-frame), for which more information can be found in [15]. The body frame (b-frame) is defined at the INS center, and is given in RFU (Right-Front-Up) convention.

The state vector is denoted as follows:

$$\mathbf{x} = \begin{bmatrix} \delta\mathbf{p}^T & \delta\mathbf{V}^T & \boldsymbol{\phi}^T & \boldsymbol{\varepsilon}^{\mathrm{b}T} & \boldsymbol{\nabla}^{\mathrm{b}T} \end{bmatrix}^T \tag{2-6}$$

in which $\delta\boldsymbol{p} = \begin{bmatrix} \delta L & \delta\lambda & \delta h \end{bmatrix}^T$ is the INS position error in latitude L, longitude λ and height h, $\delta V^{\mathrm{n}} = \begin{bmatrix} V_{\mathrm{E}} & \delta V_{\mathrm{N}} & \delta V_{\mathrm{U}} \end{bmatrix}^T$ is the velocity error in the n-frame and $\boldsymbol{\phi} = \begin{bmatrix} \phi_{\mathrm{E}} & \phi_{\mathrm{N}} & \phi_{\mathrm{U}} \end{bmatrix}^T$ denotes the orientation error in the n-frame. The state $\boldsymbol{\varepsilon}^{\mathrm{b}} = \begin{bmatrix} \varepsilon_x & \varepsilon_y & \varepsilon_z \end{bmatrix}^T$ is the bias of the gyroscope, given in the b-frame. The accelerometer specific force measurements in the

b-frame are defined as $\mathbf{f}^{\mathrm{b}} = \begin{bmatrix} f_x & f_y & f_z \end{bmatrix}^T$, with its bias $\nabla^{\mathrm{b}} = \begin{bmatrix} \nabla_x^{\mathrm{b}} & \nabla_y^{\mathrm{b}} & \nabla_z^{\mathrm{b}} \end{bmatrix}^T$. The biases are both modelled as first-order Markov processes:

$$\begin{aligned} \dot{\boldsymbol{\nabla}}^{\mathrm{b}} &= -\frac{1}{\boldsymbol{\tau}_a}\boldsymbol{\nabla}^{\mathrm{b}} + \mathbf{w}_a \\ \dot{\boldsymbol{\varepsilon}}^{\mathrm{b}} &= -\frac{1}{\boldsymbol{\tau}_\omega}\boldsymbol{\varepsilon}^{\mathrm{b}} + \mathbf{w}_\omega \end{aligned} \tag{2-7}$$

Note that $\mathbf{w}$, $\boldsymbol{\tau}_a$ and $\boldsymbol{\tau}_\omega$ are similar as in the model of (2-2). The time derivative of the orientation error $\dot{\boldsymbol{\phi}}$ is determined through:

$$\dot{\boldsymbol{\phi}} = -\boldsymbol{\omega}_{\mathrm{in}}^{\mathrm{n}} \times \boldsymbol{\phi} + \delta\boldsymbol{\omega}_{\mathrm{in}}^{\mathrm{n}} - \mathbf{C}_{\mathrm{b}}^{\mathrm{n}}\boldsymbol{\varepsilon}_{\mathrm{b}}^{\mathrm{b}} \tag{2-8}$$

in which $\boldsymbol{\omega}_{\mathrm{in}}^{\mathrm{n}}$ denotes the true angle rate and $\delta\boldsymbol{\omega}_{\mathrm{in}}^{\mathrm{n}}$ is its error vector. The matrix $\mathbf{C}_{\mathrm{b}}^{\mathrm{n}}$ is the rotation matrix from b-frame to n-frame, which is described by:

$$\mathbf{C}_{\mathrm{b}}^{\mathrm{n}} = \begin{bmatrix} \cos\phi_z & \sin\phi_z & 0 \\ -\sin\phi_z & \cos\phi_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2-9}$$

Note that a simplification is made with respect to the rotation matrix in for example [79], since roll and pitch motions are neglected in our case. Logically, this holds for both the b-frame and n-frame, and it leads to $\sin\phi_x = 1$, $\cos\phi_y = 1$, $\sin\phi_x = 0$ and $\sin\phi_y = 0$.

The change in the velocity error in the navigation frame can be calculated as:

$$\delta\dot{\mathbf{V}}^{\mathrm{n}} = \mathbf{C}_{\mathrm{b}}^{\mathrm{n}}\mathbf{f}^{\mathrm{b}} \times \boldsymbol{\phi} - (2\boldsymbol{\omega}_{\mathrm{ie}}^{\mathrm{n}} + \boldsymbol{\omega}_{\mathrm{en}}^{\mathrm{n}}) \times \delta\boldsymbol{V}^{\mathrm{n}} - (2\delta\boldsymbol{\omega}_{\mathrm{ie}}^{\mathrm{n}} + \delta\boldsymbol{\omega}_{\mathrm{en}}^{\mathrm{n}}) \times \mathbf{V}^{\mathrm{n}} + \mathbf{C}_{\mathrm{b}}^{\mathrm{n}}\nabla^{\mathrm{b}} \tag{2-10}$$

In this equation, $\boldsymbol{\omega}_{\mathrm{en}}^{\mathrm{n}}$ is the rotation velocity vector from e-frame to n-frame and $\boldsymbol{\omega}_{\mathrm{ie}}^{\mathrm{n}}$ is the rotation of the earth in the n-frame, with $\delta\boldsymbol{\omega}_{\mathrm{en}}^{\mathrm{n}}$ and $\delta\boldsymbol{\omega}_{\mathrm{ie}}^{\mathrm{n}}$ as the corresponding error vectors. The time derivative of the position error can be obtained as follows:

$$\delta\dot{\mathbf{p}} = \begin{bmatrix} \delta\dot{L} \\ \delta\dot{\lambda} \\ \delta\dot{h} \end{bmatrix} = \begin{bmatrix} \frac{\delta V_{\mathrm{N}}}{R_{\mathrm{n}}+h} - \delta h\frac{V_{\mathrm{N}}}{(R_{\mathrm{n}}+h)^2} \\ \frac{\delta V_{\mathrm{E}}}{R_{\mathrm{e}}+h}\sec L + \delta L\frac{V_{\mathrm{E}}}{R_{\mathrm{e}}+h}\tan L\sec L - \delta h\frac{V_{\mathrm{E}}\sec L}{(R_{\mathrm{e}}+h)^2} \\ \delta V_{\mathrm{U}} \end{bmatrix} \tag{2-11}$$

In this set of equations, R_{n} and R_{e} are the radii of the curvatures in the meridian and prime vertical axis of the earth, respectively. Following [66], $\boldsymbol{\omega}_{\mathrm{in}}^{\mathrm{n}}$ and $\delta\boldsymbol{\omega}_{\mathrm{in}}^{\mathrm{n}}$ can now be calculated:

$$\begin{aligned} \boldsymbol{\omega}_{\mathrm{in}}^{\mathrm{n}} &= \begin{bmatrix} -\frac{\delta V_{\mathrm{N}}}{R_{\mathrm{n}}+h} & \omega_{\mathrm{ie}}\cos L + \frac{V_{\mathrm{E}}}{R_{\mathrm{e}}+h} & \omega_{\mathrm{ie}}\sin L + \frac{V_{\mathrm{E}}}{R_{\mathrm{e}}+h}\tan L \end{bmatrix}^T \\ \delta\boldsymbol{\omega}_{\mathrm{in}}^{\mathrm{n}} &= \begin{bmatrix} -\frac{\delta V_{\mathrm{N}}}{R_{\mathrm{n}}+h} + \delta h\frac{V_{\mathrm{N}}}{(R_{\mathrm{n}}+h)^2} \\ -\delta L\omega_{\mathrm{ie}}\sin L + \frac{\delta V_{\mathrm{E}}}{R_{\mathrm{e}}+h} - \delta h\frac{V_{\mathrm{E}}}{(R_{\mathrm{e}}+h)^2} \\ \delta L\omega_{\mathrm{ie}}\cos L + \frac{\delta V_{\mathrm{E}}}{R_{\mathrm{e}}+h}\tan L + \delta L\frac{V_{\mathrm{E}}}{R_{\mathrm{e}}+h}\sec^2 L - \delta h\frac{V_{\mathrm{E}}\tan L}{(R_{\mathrm{e}}+h)^2} \end{bmatrix} = \mathbf{F}_{\mathrm{av}}\delta\mathbf{V}^{\mathrm{n}} + \mathbf{F}_{\mathrm{ap}}\delta\mathbf{p} \end{aligned} \tag{2-12}$$

Substituting (2-12) into (2-10) leads to the following:

$$\delta\dot{\mathbf{V}}^{\mathrm{n}} = \boldsymbol{C}_{\mathrm{b}}^{\mathrm{n}}\mathbf{f}^{\mathrm{b}} \times \boldsymbol{\phi} - (2\boldsymbol{\omega}_{\mathrm{ie}}^{\mathrm{n}} + \boldsymbol{\omega}_{\mathrm{en}}^{\mathrm{n}}) \times \delta\mathbf{V}^{\mathrm{n}} - (\delta\boldsymbol{\omega}_{\mathrm{ie}}^{\mathrm{n}} + \mathbf{F}_{\mathrm{av}}\delta\mathbf{V}^{\mathrm{n}} + \mathbf{F}_{\mathrm{ap}}\delta\mathbf{p}) \times \mathbf{V}^{\mathrm{n}} + \mathbf{C}_{\mathrm{b}}^{\mathrm{n}}\nabla^{\mathrm{b}} \quad (2\text{-}13)$$

For each of the states in the vector $\mathbf{x}$ of (2-6), we have now described the continuous dynamics. With the forward Euler discretization method [22], the set of equations is discretized. The full, discrete state transition equation then takes the following form:

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{I}_3 + \mathbf{F}_{11}\Delta t & \mathbf{F}_{12}\Delta t & 0_3 & 0_3 & 0_3 \\ \mathbf{F}_{21}\Delta t & \mathbf{I}_3 + \mathbf{F}_{22}\Delta t & \mathbf{F}_{23}\Delta t & \mathbf{C}_{\mathrm{b},k}^{\mathrm{n}}\Delta t & 0_3 \\ \mathbf{F}_{31}\Delta t & \mathbf{F}_{32}\Delta t & \mathbf{I}_3 + \mathbf{F}_{33}\Delta t & 0_3 & \mathbf{C}_{\mathrm{b},k}^{\mathrm{n}}\Delta t \\ 0_3 & 0_3 & 0_3 & \mathbf{I}_3 - \frac{1}{\boldsymbol{\tau}_a}\Delta t & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & \mathbf{I}_3 - \frac{1}{\boldsymbol{\tau}_\omega}\Delta t \end{bmatrix}_{k-1} \mathbf{x}_{k-1} + \mathbf{w}_k \quad (2\text{-}14)$$

From now on, we will proceed with this discrete version. The $\mathbf{F}_{ij}$ matrices can be found in Appendix A.

The covariance matrix $\mathbf{Q}$ of the process noise can be estimated at the hand of sensor specifications. It is time-varying because of the rotation matrix $\mathbf{C}_{\mathrm{b}}^{\mathrm{n}}$, that depends on the yaw angle $\phi_{z,k}$ at time k. Both the accelerometer and the gyroscope experience noise- and bias errors, which are included as standard errors $\boldsymbol{\sigma}_{\mathrm{a}}$, $\boldsymbol{\sigma}_{\omega}$, $\boldsymbol{\sigma}_{\mathrm{b,a}}$ and $\boldsymbol{\sigma}_{\mathrm{b},\omega}$, respectively. The process noise then takes the following form:

$$\mathbf{w}_k \sim N(0, \mathbf{Q}_k), \quad \mathbf{Q}_k = \begin{bmatrix} 0_3 & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & \mathbf{C}_{\mathrm{b},k}^{\mathrm{n}} & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & \mathbf{C}_{\mathrm{b},k}^{\mathrm{n}} & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & \mathbf{I}_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} 0 \\ \boldsymbol{\sigma}_{\mathrm{a}}^2 \\ \boldsymbol{\sigma}_{\omega}^2 \\ \boldsymbol{\sigma}_{\mathrm{b,a}}^2 \\ \boldsymbol{\sigma}_{\mathrm{b},\omega}^2 \end{bmatrix} \quad (2\text{-}15)$$

2-3-2 Measurement model

Remember how the INS observations were defined in (2-5). The positions are given in the local n-frame. However, the input vector $\mathbf{u}_k = \begin{bmatrix} \delta_k & a_{x,k} \end{bmatrix}^T$ provides the steering angle δ and longitudinal acceleration $a_{x,k}$ in the vehicle's b-frame. When generating realistic inputs, it is much more intuitive to do this in the n-frame. Consequently, the velocities $v_{x,k}$ and $v_{x,k}$ and the yaw rate $\omega_{z,k}$ are also expressed in the body frame. The INS observation in the n-frame can now be modelled:

$$\begin{aligned} \mathbf{p}_k^{\mathrm{INS}} &= \begin{bmatrix} x_k^{\mathrm{INS}} & y_k^{\mathrm{INS}} & z_k^{\mathrm{INS}} \end{bmatrix}^T \\ \mathbf{V}_k^{\mathrm{INS}} &= \mathbf{C}_{\mathrm{b},k}^{\mathrm{n}} \begin{bmatrix} v_{x,k}^{\mathrm{INS}} & v_{y,k}^{\mathrm{INS}} & v_{z,k}^{\mathrm{INS}} \end{bmatrix}^T \end{aligned} \quad (2\text{-}16)$$

The same holds for the GNSS observations; they can be described in the local n-frame as follows:

$$\begin{aligned}\mathbf{p}_k^{\text{GPS}} &= \begin{bmatrix} x_k^{\text{GPS}} & y_k^{\text{GPS}} & z_k^{\text{GPS}} \end{bmatrix}^T \\ \mathbf{V}_k^{\text{GPS}} &= \mathbf{C}_{\text{b},k}^{\text{n}} \begin{bmatrix} v_{x,k}^{\text{GPS}} & v_{y,k}^{\text{GPS}} & v_{z,k}^{\text{GPS}} \end{bmatrix}^T \end{aligned} \tag{2-17}$$

In the loosely coupled GNSS/INS-integration architecture, we attempt to find the error between the GNSS and the INS. Later on, this error is then subtracted from the INS measurements to determine a final state estimate. Thus, the observations for our filter are obtained by subtracting the GNSS observations from the INS observations:

$$\mathbf{z}_k = \begin{bmatrix} \delta\mathbf{p} \\ \delta\mathbf{V} \end{bmatrix} + \mathbf{v}_k = \begin{bmatrix} \mathbf{p}_k^{\text{INS}} - \mathbf{p}_k^{\text{GPS}} \\ \mathbf{V}_k^{\text{INS}} - \mathbf{V}_k^{\text{GPS}} \end{bmatrix} \tag{2-18}$$

For the measurement noise covariance matrix $\mathbf{R}$, the specifications of the considered GNSS device can be used as in (2-19). The positioning and velocity standard error vectors are denoted as $\boldsymbol{\sigma}_{\text{pos}}$ and $\boldsymbol{\sigma}_{\text{vel}}$, respectively:

$$\mathbf{v}_k \sim N(0, \mathbf{R}), \quad \mathbf{R} = \begin{bmatrix} \mathbf{I}_3 & 0_3 \\ 0_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} \boldsymbol{\sigma}_{\text{pos}}^2 \\ \boldsymbol{\sigma}_{\text{vel}}^2 \end{bmatrix} \tag{2-19}$$

Note that the measurement noise is now modelled as Gaussian, but later on we will that this is not necessarily the case. The fact that the measurement equation of (2-18) calculates the errors between GNSS and INS has little influence on the uncertainty of the measurements. That is the case because the INS errors over one time step are very small compared to the GNSS errors. In other words, we assume that the standard errors of the GNSS device are equal to the standard errors that we can use for constructing the measurement noise covariance $\mathbf{R}$ of our filtering algorithm.

2-4 State of the art

A great literature has been devoted to the traditional loosely coupled integration scheme. Many variations and extensions have been investigated. This section highlights some recent (less than 3 years of age), state-of-the-art examples of such research. When reviewing this literature, it can be seen that many of the newest journal papers focus on a problem that researchers still struggle with in loosely coupled GNSS/INS integration schemes; that of unknown process- and measurement noise covariance matrices $\mathbf{Q}$ and $\mathbf{R}$. Besides, a different branch of papers focuses on mitigating the effects of GNSS outages. Both will be discussed in this section.

2-4-1 Recent works on estimating noise covariance

The work [7] comprises several adaptive Kalman filter-based methods for GNSS/INS integration and an evaluation of their performance. A compact and clear introduction is given for three types of covariance estimation: i) innovation-based adaptive estimation (estimating $\mathbf{R}$),

ii) residual-based adaptive estimation (estimating $\mathbf{R}$) and iii) strong tracking filtering (estimating $\mathbf{Q}$). Some more advanced examples are provided in [49] (innovation-based adaptive estimation) and [24], [29] (strong tracking filtering).

In [76], the authors propose an interacting multiple model (IMM) two stage Kalman filter. A 'bias-free' filter operates without INS biases. Its output is then corrected by that of a second Kalman filter. Within this second filter, the outputs of three filters, each with different process noise covariance matrix $\mathbf{Q}$, are mixed. Likewise, [32] contains a measurement modelling method that exploits an extended state observer. The total, unpredictable uncertainty in the system is obtained by taking the difference between the measurements and extended state observer predictions. Assuming uncertainty mainly comes from measurement noise, a matrix $\mathbf{R}$ is updated based on this difference.

In an approach that bears some resemblance to interacting multiple model filtering, the authors propose a federated Kalman filter with strong tracking properties in [75]. A least-squares principle determines the mixing proportions of the outcomes of the filters. More recently, attempts have been made to estimate the process noise covariance $\mathbf{Q}$ based on the maximum likelihood (ML) principle [28], [66].

A somewhat different approach uses a multitask learning model to find matrices $\mathbf{Q}$ and $\mathbf{R}$ [74]. With the estimated orientation, velocity and INS measurements as input, the learning model is trained when proper location data (e.g. from GNSS-RTK) is available.

The aforementioned literature works are included in Table 2-1, which serves as a brief overview of noise covariance estimation methods for GNSS/INS-based vehicle localization.

Noise covariance estimation method	Estimated matrices	Examples in literature
Innovation-based adaptive estimation	$\mathbf{Q}$, $\mathbf{R}$	[7], [49]
Strong tracking filtering	$\mathbf{Q}$	[24], [29]
Interacting multiple model	$\mathbf{Q}$, $\mathbf{R}$	[41], [75], [76]
Extended state observer	$\mathbf{R}$	[32]
Maximum likelihood	$\mathbf{Q}$	[28], [66]
Learning model	$\mathbf{Q}$, $\mathbf{R}$	[74]

Table 2-1: Collection of some recent works on noise covariance estimation in loosely coupled GNSS/INS-based vehicle localization.

When reviewing the different methods, there is one clear deficiency that might potentially burden the localization process. Since loosely coupled GNSS/INS integration is bound to (nonlinear) Kalman filter approaches, the process- and measurement noise are assumed to be Gaussian. Even the adaptive methods of Table 2-1, though adapting to changes in noise amplitude, approximate the noise with Gaussian distributions. It indicates that there is still a gap of unvisited theory. The next subsection will disclose that the Gaussian assumption for the GNSS measurement noise does not hold under some circumstances.

2-4-2 Non-Gaussian GNSS noise

Some recent works in literature expose the phenomena that lead to non-Gaussian GNSS measurement noise. Various sources of non-Gaussian GNSS noise are studied in [30]. This work models noise as either standard, Gaussian white noise, non-stationary noise due to satellite distribution changes, multipath noise or GNSS outage. Similar findings were presented in [25], where a distinction is made between noise under ideal circumstances (in an open sky environment), harsh environments (routes with trees and clouds that absorb signals) and urban environments (with sky-scrapers and bridges etc.). Correlated noises of GNSS-based velocity estimates are studied in [58], thereby indicating that pink noise (or flicker noise) is also not uncommon for GNSS devices. Outliers in GNSS measurements were investigated in [54]. Changes in the number of visible satellites were seen as the largest contributors of outlier measurements. Table 2-2 is a collection of the specific circumstances that induce non-Gaussian GNSS noise, the corresponding non-Gaussian noise model and its mathematical description.

Circumstances	Noise	Mathematical model
Open sky	Gaussian noise	$\mathbf{v}_k \sim N\left(0, \sigma^2\right)$
Varying conditions	Flicker noise	$S(f) \propto \frac{1}{f}$
Change in satellites	Gaussian mixture	$\mathbf{v}_k \sim (1-\alpha)\, N(\mu, \sigma^2) + \alpha N\left(\mu, \beta\sigma^2\right)$
Urban environments	Random Walk	$\mathbf{v}_k = \alpha \mathbf{v}_{k-1} + \beta, \quad \beta \sim N\left(0, \sigma^2\right)$
Harsh environments	Gaussian, dynamic variance	$\mathbf{v}_k \sim N\left(0, \sigma_i^2\right)$
GNSS outages	Gaussian, high variance	$\mathbf{v}_k \sim N\left(0, \sigma^2\right), \quad \sigma \gg 0$

Table 2-2: Troubling circumstances, along with their corresponding noise types and mathematical description of these noise types.

It is a well-known fact that Kalman filters are sensitive to outliers [31] or other forms of non-Gaussian noise [57]. Logically, there is a strong suspicion that a Kalman-based loosely coupled GNSS/INS integration scheme will suffer from the circumstances of Table 2-2. Even the noise estimation methods of Table 2-1 can, presumably, not cope in non-Gaussian environments. The remainder of this thesis will focus on a possible solution to this problem. This proposition will be discussed extensively in the subsequent Chapter 3. The noise types of Table 2-2 will be implemented according to the noise injection method as discussed earlier in Section 2-2.

2-4-3 Mitigating effects of GNSS outages

In [76], INS biases are estimated by a second-order autoregressive integrated moving average (ARIMA) model, for which the parameters are found by the Burg estimation method [12]. This model is trained during periods with GNSS, and is used to estimate the biases in periods of GNSS outage. A very similar approach has been studied in [41], featuring an extra step in which the INS-data is pre-processing based on empirical mode decomposition for wavelet denoising. The work [79] exploits a Random Forest algorithm, that is trained to learn the input-output relation between INS measurements and location estimate when GNSS is available. Then, during GNSS outages, the system operates in pure INS-mode and uses the input-output

relation that was learned. Likewise, [47] attempts to mitigate the dependence on the (INS-based) process model. An adaptive adjustment factor is used to improve the predictions. A deep learning framework with multiple long short-term memory (multi-LSTM) modules is trained to improve the INS predictions when GNSS is not available. In closing Chapter 5, we will elaborate on this topic.

2-5 Summary

In this chapter, we have seen how we generate ground-truth data, as well as GNSS and INS measurement data. One could artificially construct GNSS and INS data from ground-truth data by a noise injection method. The injected noise should lead to INS location estimates that, over time, drift away from the real track, as where the GNSS location estimates should experience a significant amount of noise.

With this ground-truth and measurement data, we aim to realistically validate filtering-based localization methods. Such filtering-based fusion strategies have been developed to overcome the drawbacks of the GNSS and INS data. One of the most intuitive integration strategies for GNSS/INS-based vehicle localization is the loosely coupled structure. A complete mathematical model for this integration structure is provided, under the assumption that any vertical movement is neglected.

To deal with the problem of varying GNSS uncertainties, several extensions of the well-known Kalman filter have been proposed in recent research. Unfortunately, these works all are all based on the assumption that the GNSS noise is Gaussian. This is, however, not necessarily the case; some specific circumstances might induce non-Gaussian GNSS noise. The next chapter will propose a solution to this problem. At last, some works focus on mitigating the effects of GNSS outages. In general, the INS biases are learned by models during periods of good GNSS reception. The model then predicts the INS biases when GNSS is unavailable.

Chapter 3

Adaptive particle-aided cubature Kalman filter

Loosely coupled integration of global navigation satellite system (GNSS)- and inertial navigation system (INS) data is usually performed by filters of the Kalman family. Such filters provide an intuitive way of determining the proportions in which the GNSS- and INS measurements contribute to the final state estimate. The motivation for using a hybrid filtering architecture with a Kalman- and particle filter, has its origins in the general notion that particle filters are more suitable under non-Gaussian circumstances [17].

Section 3-1 will prove the superiority of the particle filter in the non-Gaussian domain at the hand of an example. The novelties of the proposed hybrid filtering architecture will be discussed in Section 3-2, after which a compact description of the full algorithm is given in Section 3-3. Section 3-4 demonstrates how the non-Gaussian distributions of the algorithm are determined, and a proof of filtering convergence is given in Section 3-5. The preliminary results of Section 3-6 and the summary of Section 3-7 complete this chapter.

3-1 Superiority particle filter in the non-Gaussian domain

There are two reasons for the superiority of the particle filter in the non-Gaussian domain: i) the posterior distribution $p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{z}_k)$ can be represented as any distribution, and ii) state transitions and observation likelihoods can be represented as any distribution. Filtering methods of the Kalman family represent these distributions as Gaussian distributions, which might lead to inaccuracy when in a non-Gaussian environment.

The nonlinear function of (3-1) is used to evaluate various nonlinear transformations that are applied in filtering methods. This model is an adjusted version of the well-known benchmark growth model, that was introduced in the original paper in which particle filters were proposed [26]. For the sake of simplicity, the time-varying component is omitted:

$$y = f(x) = \frac{x}{2} + 25\frac{x}{1+x^2} \tag{3-1}$$

A stochastic variable x has mean $\bar{x}$ and perturbation error e such that $x = \bar{x} + e$. The goal of nonlinear transformations is to estimate the statistics of the stochastic output variable y. With statistics, we indicate the first p-order central moments of this output variable y. For now, we are particularly interested in the first 2 central moments, i.e. the mean and the covariance. Fig. 3-1 below provides a graphical representation of such transformations applied on a dynamic system.

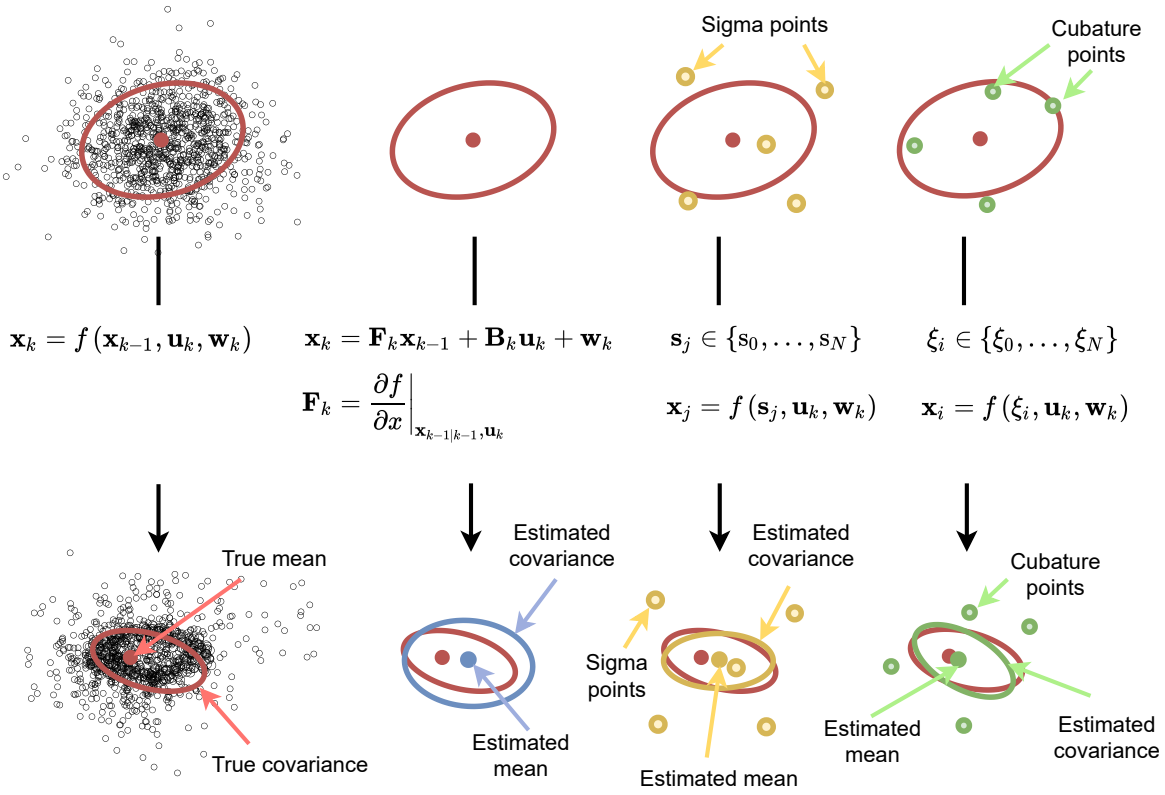

Figure 3-1: Several transformations of a stochastic variable $\mathbf{x}_{k-1}$ through a nonlinear function f. Left: propagation of 1000 Monte Carlo samples through f leads to an approximation of the stochastic variable $\mathbf{x}_k$. Center-left: using a first order Taylor approximation (i.e. the linearization of the extended Kalman filter), the statistics of $\mathbf{x}_k$ can be estimated. Center-right: in an unscented transformation, multiple sigma-points $\mathbf{s}_j$ within the space of $\mathbf{x}_{k-1}$ are generated and propagated through f. Right: a spherical-radial rule is used to generate cubature-points. Similarly to the unscented transform, these points have the property that they can approximate the statistics of $\mathbf{x}_k$. Adopted from [70].

The true distribution of y can be approximated by Monte Carlo simulations; the resulting distribution of variable y after propagating x $1 \cdot 10^5$ times through (3-1) is given by the blue density of Fig. 3-2. Performing more Monte Carlo simulations does not lead to a different result. Performing this many simulations is often infeasible when considering real-time applications. Therefore, several nonlinear transformations have been developed.

To test these nonlinear transformations, two simulations are performed: one with $e_1 \sim N(0, 0.1)$ (top graphs of Fig. 3-2) and one with $e_2 \sim \text{Pois}(\lambda = 0.1)$ (lower two graphs of Fig. 3-2). Note that the variance of x is equal for both cases ($\sigma^2_{e_1} = \sigma^2_{e_2}$). The Poisson distributed variable σ_{e_2} is normalized, such that x has mean 1 in both cases. Thus, the most important simulation conditions, i.e. the first 2 central moments, are exactly the same. However, the shape of the stochastic variable x is not the same. Properties of the shape of distributions (e.g. kurtosis, skewness) are harvested in higher order central moments. The means of the probability distributions of the left graph of Fig. 3-2 are indicated by the dotted lines of the graphs on the right.

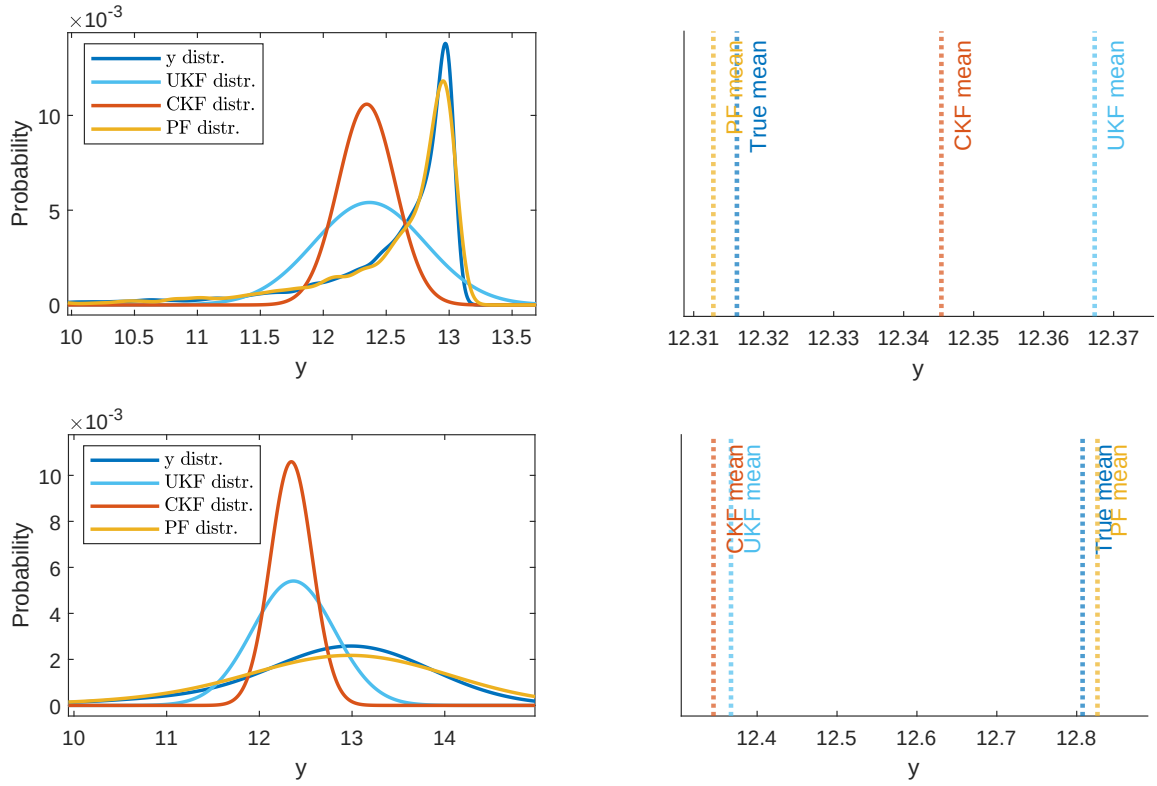

Figure 3-2: Top left: for a Gaussian distributed x; true distribution of $y = f(x)$ (3-1) and the estimated distributions of an unscented transform (UKF, indigo), cubature transform (CKF, orange) and Monte Carlo approximation with 10 samples (particle filter (PF), yellow). Bottom left: similar, but for a stochastic x that follows a Poisson distribution. Top right: enlarged on the true mean and approximated means. Bottom right: similar, but for non-Gaussian x. The Monte Carlo approach convincingly outperforms other approaches in terms of approximating the true mean. Especially for non-Gaussian distributed x, the difference in performance is considerable.

What stands out is that the true probability distribution of y is non-Gaussian, even for normally distributed e. This indicates that propagation through nonlinear functions can change the type of distribution of stochastic variables.

Now, an unscented transformation, a third-degree cubature transformation and a Monte Carlo sampling method (with 100 particles) are applied to find the statistics of y without having to perform $1 \cdot 10^5$ function evaluations. For $e_1 \sim N(0, 0.1)$, the results of these nonlinear

transformations are also included in Fig. 3-2. Both the unscented- and third-degree cubature transformation represent y as a Gaussian, as expected. However, the Monte Carlo sampling method captures the true shape of the density of y much better.

For a second simulation, we assume that there is full knowledge of the higher order statistics of e_2. The unscented- and cubature transformation allow a mean and covariance matrix as inputs for x, but these do not change. There is nothing to adjust in these algorithms. Since the Monte Carlo method can draw and weight particles from any distribution, it can also draw from $e_2 \sim \text{Pois}(\lambda = 0.1)$. With these settings, the same simulations are performed again, and the resulting density estimates are given in the lower-left graph of Fig. 3-2. For this non-Gaussian x, the mean of y is now approximated poorly by the distributions of the unscented- and cubature transformations. The Monte Carlo method results in a much better estimate. The difference in performance has increased considerably with respect to the simulation with a Gaussian distributed x. Thus, with some knowledge of a possibly non-Gaussian variable x, the mean of the variable y can be approximated better by Monte Carlo methods, such as the particle filter. This strong advantage will be exploited in our hybrid filtering algorithm.

3-2 Novel hybrid filtering architecture

For the specific case of the previous section, the Monte Carlo sampling methods (such as the particle filter) are more accurate than nonlinear Kalman-based methods as the unscented Kalman filter and cubature Kalman filter. However, the nonlinear Kalman filters often perform adequately in Gaussian environments. Also, in higher-dimensional systems, more and more particles are needed to obtain adequate results. This results in a considerable number of function evaluations that are necessary if the particle filter were to match the performance of the nonlinear variants of the Kalman filter. Researchers have been attempting to combine both filters in order to use both their strengths.

3-2-1 Traditional hybrid filters

Over the years, several hybrid filtering architectures have been developed that employ both a particle filter and a nonlinear Kalman filter. This section will discuss some common ones, and will briefly introduce the proposed hybrid filtering architecture.

Switching strategy The most straight forward hybrid filter architecture is given by the left-most scheme of Fig. 3-3. The two filters work in parallel, and a switching mechanism based on an arbitrary criterion determines which output is used. A distinction is made between a hard switch (either use the particle filter output or the Kalman-type filter output) and a soft switch (a weighted average of both). Such an approach, using a hard switch, is used for object tracking in [16].

We prefer the cubature Kalman filtering algorithm over the unscented Kalman filtering algorithm for two reasons: i) the former is faster (one function evaluation less is needed) and ii) a square-root configuration is possible. Square-root filtering solves the problem of indefinite covariance matrices. The underlying meaning of covariance has its roots in its positive definiteness; a squared standard deviation can not be negative by definition. However, if very

accurate measurements are processed, or if some states are known with high accuracy, while others are essentially unobservable, the numerically computed covariance $\mathbf{P}$ might turn out non-positive definite [35]. In square-root filtering, a factorization is applied that guarantees positive definiteness, thereby increasing the numerical stability of the filter. The additional steps for square-root configuration will be provided in the full algorithm description of Section 3-3.

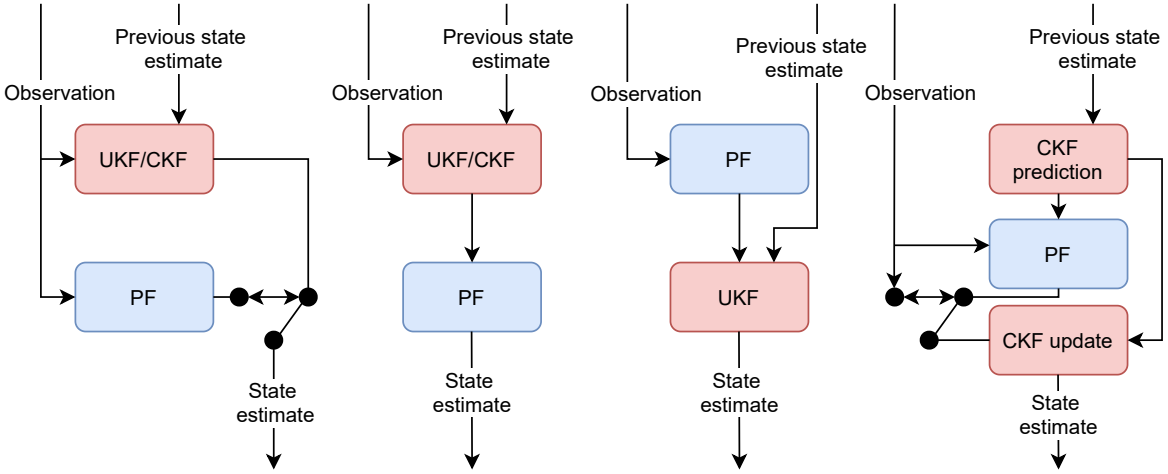

Figure 3-3: From left to right: hybrid switching architecture, unscented/cubature particle filter architecture, particle-aided unscented Kalman filter architecture and the proposed adaptive particle-aided cubature Kalman filter.

Unscented particle filter Perhaps the most common hybrid filtering method is given by the center-left scheme of Fig. 3-3. Since it is generally rather difficult to find a good proposal distribution for the particle filter with limited knowledge, a nonlinear Kalman-type filter provides it. The method gained attention under the name unscented particle filter [71]. Although this method could solve the problem of non-Gaussian measurement noise, the drawbacks of this method are as follows: i) the proposal distribution is exclusively Gaussian, ii) when accurate predictions are available, one does not want to include inaccurate observations in the proposal distribution, and iii) the particle filters sampling procedure might decrease the accuracy with respect to the nonlinear Kalman estimate in pure Gaussian environments. The sampling procedure inevitably leads to estimation errors (unless an infinite number of particles is used), with the magnitude of these errors depending on the number of samples. This might, in some cases, lead to situations (mainly in Gaussian environments) in which the nonlinear Kalman estimate that is used for the proposal distribution is more accurate than the subsequent particle filter estimate. This is, logically, highly undesirable. A switching mechanism prevents this decrease in accuracy due to sampling.

Particle-aided unscented Kalman filter In a recent series of journal papers, the authors propose the center-right approach of Fig. 3-3 for vehicle localization [44, 45, 46]. The particle filter is now connected to the observation channel of the unscented Kalman filter. It has the advantage that the particle filter can work with a reduced version of the state-space model, one that only calculates the measurable states. There is, however, one large downside of this

method; there are now limited tools for constructing the proposal distribution and observation likelihood. We can explain this as follows: in the proposed particle-aided unscented Kalman filter, the particle filter can weight the particles according to a non-Gaussian measurement likelihood, leading to a more accurate observation for the unscented Kalman filter. This should lead to a smaller measurement covariance matrix for the unscented Kalman filter algorithm. An obvious choice for the measurement noise covariance matrix would be to use the covariance matrix of the particle filter estimate. However, using this matrix increases the vulnerability to filter instability, since the particle filters covariance estimate is prone to negative definiteness. The last work of the series of papers [44] solves this issue by employing a discriminative parameter training that learns an optimal measurement likelihood distribution. This looks, however, like a brute force approach that only adds unnecessary complexity to a problem that can be avoided easily.

Adaptive particle-aided cubature Kalman filter The novel hybrid filtering architecture of the right diagram of Fig. 3-3 is more attractive for problems such as GNSS/INS-based vehicle localization. The particle filter can work with any measurement likelihood distribution, and we can implement an adaptive soft switching between Kalman-based filtering and particle filtering, which is based on the Gaussianity of the measurement noise. The latter makes sure that the final state estimates are not disturbed by the sampling procedure of the particle filter in Gaussian circumstances, which was the case for the aforementioned unscented particle filter. A cubature Kalman filter is employed to make a square-root configuration is possible, guaranteeing numerical filter stability. Later on in Subsection 4-2-2, we will explain why we use the cubature Kalman filter predictions as proposal distribution. This chapter will mainly focus on the soft switching mechanism.

3-2-2 A key assumption

Section 3-1 has shown that particle filters are superior in case of non-Gaussian noise, given that some knowledge with respect to the shape of this non-Gaussian distribution (i.e. the higher order central moments) is available. This subsection will discuss an assumption that is necessary to make this possible.

Consider the following multivariate nonlinear model:

$$\begin{aligned}\mathbf{x}_k &= \begin{bmatrix} x_{1,k-1} + \tau x_{2,k-1} \\ x_{2,k-1} + \tau\left(-x_{1,k-1} + \left(x_{1,k-1}^2 + x_{2,k-1}^2 - 1\right) x_{2,k-1}\right) \end{bmatrix} + \mathbf{w}_k \quad , \qquad \mathbf{w}_k \sim N\left(0, \sigma_{\mathrm{p}}^2\right) \\ \mathbf{y}_k &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{x}_k + \mathbf{v}_k \end{aligned} \tag{3-2}$$

with time constant $\tau = 0.001$ and the standard deviation of the process noise set to $\sigma_{\mathrm{p}} = 0.0005$. Imagine that the measurement noise $\mathbf{v}_k$ has a standard deviation of $\sigma_{\mathrm{m}} = 0.5$. The process noise is then much smaller than the measurement noise. In this specific case, with accurate knowledge of the previous state $\mathbf{x}_{k-1}$, the prediction $\hat{\mathbf{x}}_{k|k-1}$ of a nonlinear Kalman

filter, e.g. the cubature Kalman filter, is rather accurate. One might wonder; why not use these accurate predictions as final state estimates? In many realistic systems (e.g. GNSS/INS-based vehicle localization) the prediction error builds up over time. That is why, after a while, the predictions become useless and there is a need for observation corrections in our final state estimate.

Let us say that a filter is convergent, and the previous state is therefore estimated accurately (its estimation error $\mathbf{e}_k$ is small). The predicted observation $\hat{\mathbf{z}}_k$ is then a good approximation of the true outputs $\mathbf{y}_k$. Kalman filters denote the difference between the predicted observation $\hat{\mathbf{z}}_k$ and the actual observation $\mathbf{z}_k$ as the innovation; $\tilde{\mathbf{y}}_k = \mathbf{z}_k - \hat{\mathbf{z}}_k$. Under our specific circumstances, the measurement noise $\mathbf{v}_k$ can be approximated by this innovation signal. This is shown below:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \hat{\mathbf{z}}_k = (\check{\mathbf{x}}_k + \mathbf{v}_k) - (\check{\mathbf{x}}_k + \mathbf{e}_k + \mathbf{w}_k) = \mathbf{v}_k - \mathbf{e}_k - \mathbf{w}_k \approx \mathbf{v}_k \tag{3-3}$$

in which $\check{\mathbf{x}}_k$ is the true state, $\mathbf{w}_k$ and $\mathbf{v}_k$ are the process- and measurement perturbations and $\mathbf{e}_k$ as the estimation error of the previous state.

To summarize, assuming that the innovation signal sufficiently accurately approximates the measurement noise, we possess over the right knowledge to let our particle filter perform superior to the cubature Kalman filter. How the possible non-Gaussian measurement likelihood distribution is build, will be disclosed later on in Section 3-4. Firstly, the next Subsection 3-2-3 will focus on the soft switching mechanism of the proposed hybrid filter.

3-2-3 Kolmogorov-Smirnov test

As stated before, the Gaussianity of the measurement noise will determine the mixing proportions of the particle filter and the cubature Kalman filter for the final filtered state estimate. To determine a measure for the Gaussianity of the measurement noise, a Kolmogorov-Smirnov test [52] can be used. Let us take the last n normalized innovations (i.e. $\tilde{\mathbf{y}}_{i,\text{norm}} = \tilde{\mathbf{y}}_i / |\sigma_{\tilde{\mathbf{y}}}|$) as data points. The method first determines an empirical cumulative distribution function $F_k(x)$ for the n ordered innovations $\tilde{\mathbf{y}}_i$:

$$F_k(x) = \frac{1}{n} \sum_{i=k-n}^{k} 1_{[-\infty,x]} \left(\tilde{\mathbf{y}}_{i,\text{norm}}\right) \tag{3-4}$$

Note that $1_{[-\infty,n]} \left(\tilde{\mathbf{y}}_{i,\text{norm}}\right)$ is the indicator function. This function equals 1 if $\tilde{\mathbf{y}}_{i,\text{norm}} \leq x$ and equals 0 otherwise. The Kolmogorov-Smirnov statistic is then defined as the supremum of the set of distances between a reference cumulative distribution $F(x)$ (in our case a standard normal distribution) and the empirical innovation cumulative distribution $F_k(x)$:

$$D_k = \sup_x |F_k(x) - F(x)| \tag{3-5}$$

A graphical representation can be found in Fig. 3-4.

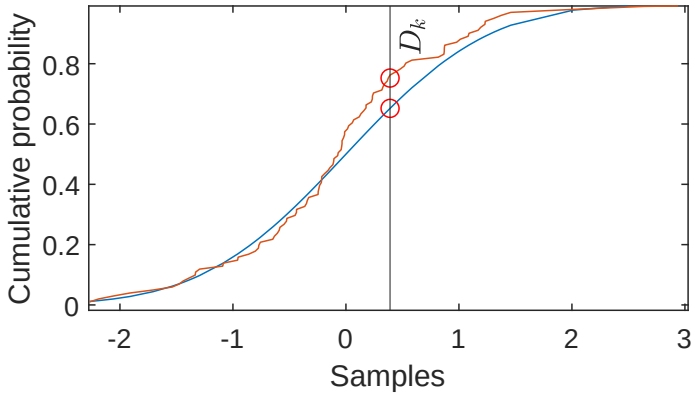

Figure 3-4: Cumulative distribution function of a standard normal distribution (blue), and the empirical cumulative distribution function of the last n normalized innovations $\tilde{\mathbf{y}}_{i,\text{norm}}$ (red). The Kolmogorov-Smirnov statistic is supremum of the set of distances between the two distributions, as indicated by the red markers.

The null hypothesis that the last n normalized innovations come from a standard normal distribution can then be rejected with $1-\alpha$ confidence based on the following:

$$\breve{\mu}_k = \begin{cases} 1, & \sqrt{n}D_k > K_\alpha, \quad K = \sup_{t\in[0,1]} |B(t)|, \qquad p\left(K \leq K_\alpha\right) = 1-\alpha \\ 0, & \sqrt{n}D_k < K_\alpha \end{cases} \tag{3-6}$$

in which $B(t)$ is a Brownian bridge function [51]. Note that the bold symbol $\breve{\boldsymbol{\mu}}_k$ is a vector with its dimension as the number of measurable states, and the symbol $\breve{\mu}_k$ denotes one of its entries. To avoid fast, aggressive switching between filter settings, the definitive Gaussianity indicator $\boldsymbol{\mu}_k$ is build up from the averages of $\breve{\mu}_k$ over the window n: $\mu_k = \frac{1}{n}\sum_{i=k-n}^{k} \breve{\mu}_i$. Stability issues might arise when $\mu_k = 1$, as we will see in Section 3-5. Therefore, a safety factor $\gamma \ll 1$ is employed to avoid this from happening.

$$\boldsymbol{\mu}_k = \min\left(\frac{1}{n}\sum_{i=k-n}^{k} \breve{\boldsymbol{\mu}}_i, \quad 1-\gamma\right), \qquad 0 < \gamma \ll 1 \tag{3-7}$$

3-2-4 Soft switching between two filters

The particle filter is connected to a cubature Kalman filter through the observations channel of the standard algorithm. Instead of using the standard observations vector $\mathbf{z}_k$, a modified version is used:

$$\tilde{\mathbf{z}}_k = (1-\boldsymbol{\mu}_k)\,\mathbf{z}_k + \boldsymbol{\mu}\mathbf{z}_k^{\text{pf}} \tag{3-8}$$

in which $\mathbf{z}_k^{\text{pf}}$ is the final output estimate of the particle filter. For each individual state, this equation makes sure the filter is functioning as a standard cubature Kalman filter when $\mu_k \to 0$, and uses the particle filter's result as observation when $\mu_k \to 1$. When the uncertainty

of this new observation is extremely low, the filter 'trusts' the observations more than the predictions, and eventually, the observations will contribute significantly to the final state estimate. Therefore, we adjust the uncertainty of the observations with the matrix $\tilde{\mathbf{R}}_k$. Instead of using a normal, fixed measurement noise matrix $\mathbf{R}$, the following matrix is used:

$$\mathbf{R} \Rightarrow \tilde{\mathbf{R}}_k = (1 - \boldsymbol{\mu}_k)\,\mathbf{R} \tag{3-9}$$

For each individual state, this adjustment leads to a small measurement noise variance when $\mu_k \to 1$ ($\lim_{\mu_k \to 1} \tilde{\mathbf{R}}_k = 0$), but the normal measurement noise variance is used when $\mu_k \to 0$ ($\lim_{\mu_k \to 0} \tilde{\mathbf{R}}_k = \mathbf{R}$). In the former case, the cubature Kalman filter predictions are neglected, since the process noise is generally larger than $\tilde{\mathbf{R}}_k \to 0$. The cubature Kalman filter then fully uses the particle filter's estimate as the definitive one.

Let it be noted that from now on, the matrices $\tilde{\mathbf{Q}}_k$ and $\tilde{\mathbf{R}}_k$ are used as tools to manipulate the outcome of our state estimate, rather than representing the process- and measurement noise covariance matrices $\mathbf{Q}$ and $\mathbf{R}$. Both the process- and measurement noise covariance matrices can often not be determined accurately. In fact, the noise statistics are often dynamic, and inaccurate, fixed values for the noise covariance can be harmful for the estimation process. Although more configurations are possible, we exploit $\tilde{\mathbf{z}}_k$ and $\tilde{\mathbf{R}}_k$ as a switching mechanism between our filters, and we use $\tilde{\mathbf{Q}}_k$ as an adaptive measure for the unknown components of our system. The instrumental matrix $\tilde{\mathbf{Q}}_k$ is calculated as:

$$\tilde{\mathbf{Q}}_k = \rho \tilde{\mathbf{y}}_k^T \tilde{\mathbf{y}}_k \mathbf{I}_n + \eta \mathbf{I}_n \tag{3-10}$$

with ρ and η as tuning parameters. Later on in the convergence proof of Section 3-5, the reason for this specific choice for $\tilde{\mathbf{Q}}_k$ will be explained. Both the instrumental matrices are only used in specific steps of the algorithm. In other phases of the proposed hybrid filter, a set of standard, fixed $\mathbf{Q}$ and $\mathbf{R}$ are used, as will be made clear in the full algorithm description of Section 3-3.

3-2-5 Demonstration of the hybrid architecture

The hybrid filtering architecture, with the soft switching mechanism that depends on the Gaussianity of the measurement noise, will now be evaluated with the aforementioned multivariate nonlinear model of (3-2). Imagine there would be a small amount of bias on these states, such that the predictions of the process equation are worthless over large time spans. We need the observations to correct for this error. The model is simulated for 50.000 time steps. After 25.000 steps, the measurement noise abruptly changes from $\mathbf{v}_k \sim N(0, 0.25)$ to a Gaussian mixture noise model:

$$\mathbf{v}_k \sim \left(\begin{bmatrix} 0 \\ 0.1 \end{bmatrix} + 0.5 \cdot N(0, 0.25 \cdot \mathbf{I}_2) \right) + \left(\begin{bmatrix} -0.1 \\ 0 \end{bmatrix} + 0.5 \cdot N\left(0, 10 \cdot 0.25 \cdot \mathbf{I}_2\right) \right) \tag{3-11}$$

For comparison, four filters are reviewed: a cubature Kalman filter (CKF), a particle filter (PF), an innovation-based adaptive estimation unscented Kalman filter (IBAE-UKF) and the proposed adaptive particle-aided cubature Kalman filter (PACKF) with $N_p = 30$, adaptive

$\tilde{\mathbf{Q}}_k$ matrix (3-10) and switching variable vector $\boldsymbol{\mu}_k$. For both particle filters, we use simple multinomial resampling and the prior $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$ as proposal distribution. The innovation-based feature of the unscented Kalman filter adapts its measurement noise covariance matrix $\mathbf{R}_k$ based on the innovation signal. Remember from Section 2-4 that this approach accounts for the changes in noise covariance, but it is still bound to a Gaussian shape for representing the noise statistics.

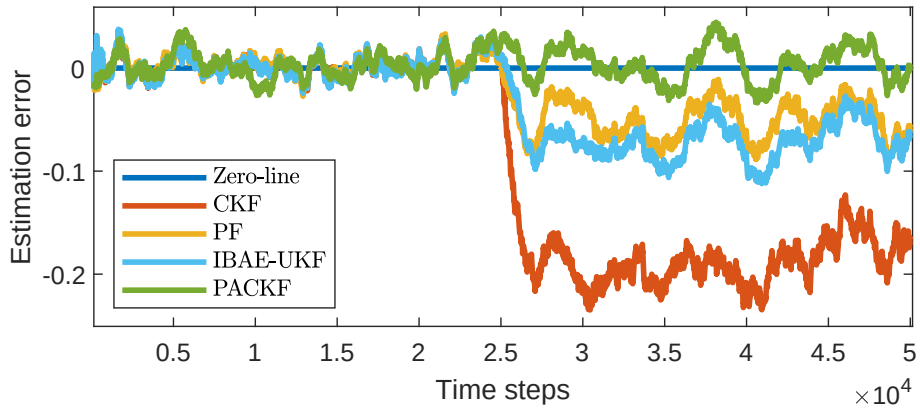

Figure 3-5: Estimation errors for the first state of the multivariate nonlinear model of (3-2). After 25.000 time steps, the measurement noise changes from a Gaussian distribution to a Gaussian mixture model (3-11), which is non-Gaussian. Clearly, the estimation errors of the proposed PACKF algorithm are the lowest in the second half of the simulation.

Fig. 3-5 shows the estimation errors of the first state of the model (3-2). Up until half of the simulation period (i.e. the Gaussian case), all filters perform similarly. From 25.000 time steps onwards, both the cubature Kalman filter estimation error and the innovation-based adaptive estimation error increase. The particle filter and particle-aided cubature Kalman filter perform better, with the proposed method resulting in the lowest estimation errors. It is suspected that the adaptive $\tilde{\mathbf{Q}}_k$ matrix accounts for this difference with respect to the standard particle filter.

3-3 Full algorithm description

The full adaptive particle-aided cubature Kalman filtering algorithm can be split up into three phases; i) the cubature Kalman prediction step, ii) the particle filter and iii) the cubature Kalman update step. Each phase will be highlighted in this section, with the corresponding mathematical steps provided by Tables 3-1, 3-2, and 3-3.

3-3-1 Cubature Kalman prediction

The prediction step follows that of a regular square-root cubature Kalman filter, except for the adaptive matrix $\tilde{\mathbf{Q}}_k$. For the square-root configuration, several triangular factorizations exist. Remember that this step assures that the covariance matrix $\mathbf{P}$ can never have a negative diagonal or become asymmetric. Well-known triangular factorizations are the Cholesky

decomposition [14], the U-D decomposition and the QR-decomposition. These are denoted by the Tria operator. We will use the Cholesky decomposition in our algorithm.

CKF prediction	Steps of the algorithm
1. Decompose last covariance	$\mathbf{S}_{k-1\|k-1} = \text{Tria}\left(\mathbf{P}_{k-1\|k-1}\right)$
2. Calculate cubature points	$\boldsymbol{X}_{i,k-1\|k-1} = \mathbf{S}_{k-1\|k-1}\xi_i + \hat{\mathbf{x}}_{k-1\|k-1}$
3. Propagate through f	$\boldsymbol{X}^*_{i,k\|k-1} = f\left(\boldsymbol{X}_{i,k-1\|k-1}\right)$
4. Predict state	$\hat{\mathbf{x}}_{k\|k-1} = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{X}^*_{i,k\|k-1}$
5. Calculate centered matrix	$\mathcal{X}^*_{k\|k-1} = \frac{1}{\sqrt{m}}\left[\boldsymbol{X}^*_{1,k\|k-1} - \hat{\mathbf{x}}_{k\|k-1} \cdots \boldsymbol{X}^*_{m,k\|k-1} - \hat{\mathbf{x}}_{k\|k-1}\right]$
6. Calculate $\tilde{\mathbf{Q}}_k$	$\tilde{\mathbf{Q}}_k = \rho\tilde{\mathbf{y}}^T_{k-1\|k-1}\tilde{\mathbf{y}}_{k-1\|k-1}\mathbf{I}_n + \eta\mathbf{I}_n$
7. Decompose $\tilde{\mathbf{Q}}_k$	$\mathbf{S}_{Q,k} = \text{Tria}\left(\tilde{\mathbf{Q}}_k\right)$
8. Square-root of covariance	$\mathbf{S}_{k\|k-1} = \text{Tria}\left(\begin{bmatrix}\mathcal{X}^*_{k\|k-1} & \mathbf{S}_{Q,k}\end{bmatrix}\right)$
9. Calculate covariance	$\mathbf{P}_{k\|k-1} = \mathbf{S}_{k\|k-1}\mathbf{S}^T_{k\|k-1}$

Table 3-1: Steps for the prediction step of a square-root cubature Kalman filter. The Tria denotes an arbitrary triangular factorization.

The instrumental matrix $\tilde{\mathbf{Q}}_k$ is determined at the hand of fixed, pre-defined constants ρ and η, and the previous innovation signal $\tilde{\mathbf{y}}_{k-1|k-1}$. A total of $m = 2n$ cubature points ξ_i are defined in an initialization step, which is omitted for the sake of compactness. An extensive, theoretical background on the origin of these points can be found in the original work of its inventors [8].

3-3-2 Particle filter

Table 3-2 below shows the steps of the particle filter algorithm. A great advantage of the proposed hybrid filter, is that it is still compatible with any type of proposal distribution and any measurement likelihood distribution. In fact, it hands us extra tools for constructing a proposal distribution, since the prior and posterior of the cubature Kalman filter can be used as well.

Especially in case of small process noise covariance, these extra tools are useful. That is, because the most common choice for a proposal distribution, i.e. the standard prior distri-

bution $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$, has extremely short tails that do not cover the tails of the observation likelihood. Thus, the most common choice for the proposal distribution will result in particle degeneracy.

PF	Steps of the algorithm
10. Draw particles from proposal distribution	$\mathbf{x}_k^i \sim q(\mathbf{x}_k^i \mid \mathbf{z}_k, \mathbf{x}_{k-1}^j)$
11. Update particle weights *	$w_k^i \propto w_{k-1}^j \cdot \frac{p\left(\mathbf{z}_k \mid \mathbf{x}_k^i\right) p\left(\mathbf{x}_k^i \mid \mathbf{x}_{k-1}^j\right)}{q\left(\mathbf{x}_k^i \mid \mathbf{x}_{k-1}^j, \mathbf{z}_k\right)}$
12. Calculate particle filter estimate	$\hat{\mathbf{x}}_{k\mid k}^{\mathrm{pf}} \approx \sum_{i=1}^{N} w_k^i \mathbf{x}_k^i$
13. Propagate through h	$\mathbf{z}_k^{\mathrm{pf}} = h\left(\hat{\mathbf{x}}_{k\mid k}^{\mathrm{pf}}\right)$

Table 3-2: Steps of the particle filter that is used in the proposed hybrid filtering architecture. * If one uses the prior $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$ as proposal distribution, the equation reduces to $w_k^i \propto w_{k-1}^j \cdot p\left(\mathbf{z}_k \mid \mathbf{x}_k^i\right)$.

As we have seen in Section 3-1, particle filters perform better than Kalman-like filters, given the condition that some knowledge on the non-Gaussian statistics is available. Is it then wise to draw from a non-Gaussian proposal distribution? Not necessarily.

In non-Gaussian environments, the shape of the proposal distribution is not that important. It must be close in shape to the true posterior, and a Gaussian shape often performs well. More important is that one draws particles in the neighborhood of the observations and true states. Thus, the support, which in the case of a Gaussian proposal distribution can be characterized by the 3σ-bounds, is relevant rather than the exact shape. This means that drawing particles from a non-Gaussian distribution does not solve the problem of filtering non-Gaussian measurement noise. That is one of the reasons that we make the support of our proposal distribution adaptive with the instrumental matrix $\tilde{\mathbf{Q}}_k$; to make sure that the support of the proposal distribution adapts to the measurement uncertainty. Later on, Fig. 3-8 will delicately explain why the support of a proposal distribution is important rather than the shape.

Then where do we need to implement a non-Gaussian distribution? The particles are weighted with the observation likelihood $p\left(\mathbf{z}_k \mid \mathbf{x}_k^i\right)$, and if this is a non-Gaussian distribution, this is where we should implement our non-Gaussian distribution approximation. The next section will describe how the non-Gaussian observation likelihood $p\left(\mathbf{z}_k \mid \mathbf{x}_k^i\right)$ can be determined. As such, the superiority of the particle filter as described in Section 3-1 can be exploited.

3-3-3 Cubature Kalman filter update

In the last phase of the algorithm, a compromise is made between the standard observation $\mathbf{z}_k$, and the observation $\mathbf{z}_k^{\mathrm{pf}}$ that comes from the particle filter state estimate. The stan-

dard measurement update algorithm of the cubature Kalman filter is adopted, only with two adjustments. The parameter $\boldsymbol{\mu}_k$ is implemented as described by (3-8) and (3-9), thereby changing the algorithm at step 18 and 23 of Table 3-3.

CKF Update	Steps of the algorithm
14. Evaluate cubature points	$\boldsymbol{X}_{i,k\|k-1} = \mathbf{S}_{k\|k-1}\xi_i + \hat{\mathbf{x}}_{k\|k-1}$
15. Propagate through h	$\boldsymbol{Z}_{i,k\|k-1} = h\left(\boldsymbol{X}_{i,k\|k-1}\right)$
16. Predict measurement	$\hat{\mathbf{z}}_{k\|k-1} = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{Z}_{i,k\|k-1}$
17. Calculate centered matrix	$\mathcal{Z}_{k\|k-1} = \frac{1}{\sqrt{m}}\left[\boldsymbol{Z}_{1,k\|k-1} - \hat{\mathbf{z}}_{k\|k-1} \cdots \ \boldsymbol{Z}_{m,k\|k-1} - \hat{\mathbf{z}}_{k\|k-1}\right]$
18. Decompose $\tilde{\mathbf{R}}_\mathbf{k}$	$\mathbf{S}_{R,k} = \text{Tria}\left(\left(1 - \boldsymbol{\mu}_k\right)\mathbf{R}\right)$
19. Decompose $\tilde{\mathbf{y}}$ covariance	$\mathbf{S}_{zz,k\|k-1} = \text{Tria}\left(\left[\mathcal{Z}_{k\|k-1} \quad \mathbf{S}_{R,k}\right]\right)$
20. Calculate centered matrix	$\mathcal{X}_{k\|k-1} = \frac{1}{\sqrt{m}}\left[\boldsymbol{X}_{1,k\|k-1} - \hat{\mathbf{x}}_{k\|k-1} \cdots \ \boldsymbol{X}_{m,k\|k-1} - \hat{\mathbf{x}}_{k\|k-1}\right]$
21. Calculate cross-covariance	$\mathbf{P}_{xz,k\|k-1} = \mathcal{X}_{k\|k-1}\mathcal{Z}_{k\|k-1}^T$
22. Determine Kalman gain	$\mathbf{K}_k = \left(\mathbf{P}_{xz,k\|k-1}/\mathbf{S}_{zz,k\|k-1}^T\right)/\mathbf{S}_{zz,k\|k-1}$
23. Calculate observation	$\tilde{\mathbf{z}}_{k\|k} = \left(1 - \boldsymbol{\mu}_k\right)\mathbf{z}_k + \boldsymbol{\mu}_k\,\mathbf{z}_k^{\text{pf}}$
24. Calculate state estimate	$\hat{\mathbf{x}}_{k\|k} = \hat{\mathbf{x}}_{k\|k-1} + \mathbf{K}_k\left(\tilde{\mathbf{z}}_{k\|k} - \hat{\mathbf{z}}_{k\|k-1}\right)$
25. Square-root of covariance	$\mathbf{S}_{k\|k} = \text{Tria}\left(\left[\mathcal{X}_{k\|k-1} - \mathbf{K}_k\mathcal{Z}_{k\|k-1} \quad \mathbf{K}_k\,\text{Tria}(\mathbf{R})\right]\right)$

Table 3-3: Measurement update step of the proposed algorithm. The Tria denotes an arbitrary triangular factorization.

Note that in the final step 25 of the algorithm, i.e. the calculation of the square-root of the covariance matrix, the pre-defined, time-invariant measurement noise covariance matrix $\mathbf{R}$ is used. It is used instead of the earlier calculated $\mathbf{S}_{R,k}$, as would be the case in the standard algorithm. We do this for two reasons. At first, the following might occur:

$$\boldsymbol{\mu}_k \to 1 \quad \Rightarrow \quad (1 - \boldsymbol{\mu}_k)\mathbf{R} \to 0 \tag{3-12}$$

This would mean that we are dealing with extremely accurate observations, and the square-root of the covariance of the final state estimate would become very small as well. This can,

however, not be the case in a realistic system. We merely manipulate the algorithm to use the particle filter estimate instead of the cubature Kalman prediction.

As a second, a flawed covariance estimate in step 25 is problematic when, in the next time step, the cubature points are drawn based on this covariance. The predictions will be very poor. Also, when this covariance is used for the proposal distribution of the particle filter, or when the estimate is to be fused with other estimates in later steps, an inaccurate covariance matrix induces problems. Therefore, a fixed estimate for $\mathbf{R}$ is used in step 25 of Table 3-3.

3-4 Adaptive Kernel Density Estimation

The particle weighting step of Table 3-2 requires the determination of the observation likelihood $p(\mathbf{z}_k \mid \mathbf{x}_k)$. In practice, gaining true statistics of the actual observation likelihood is difficult. Often, this likelihood is assumed to be Gaussian, and is held constant.

Unfortunately, the measurement noise statistics might be time-varying and non-Gaussian in reality. To overcome the problem of inaccurate particle weighting due to incorrect measurement modelling, an adaptive particle weighting scheme is proposed. The last n innovations $\tilde{\mathbf{y}}_k$, which are essentially the measurement noise values due to our assumption of (3-3), are used to approximate the true observation likelihood. The following calculation shows how this probability distribution function can be obtained:

$$\widehat{f}_{\mathrm{KDE}}(x) = \frac{1}{n}\sum_{i=1}^{n} K_h\left(x - \tilde{y}_i\right) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-\tilde{y}_i}{h}\right) \tag{3-13}$$

The function K indicates an arbitrary kernel, $\tilde{\mathbf{y}}_i$ represents one of the last n innovations, and the parameter h denotes the bandwidth. The latter is a free parameter that determines the smoothness of the resulting distribution function. A common choice for this bandwidth, the one that minimizes the mean integrated squared error, can be found with:

$$h = \theta\,\hat{\sigma}\,n^{-\frac{1}{5}}, \quad \theta = 1.06 \tag{3-14}$$

Such an approach for estimating densities at the hand of data points, is called kernel density estimation [19, 56]. For the kernel function $K\left((x-\tilde{\mathbf{y}}_i)/h\right) = K(u)$, an Epanechnikov kernel [21] is used. This is a parabolic kernel function generally performs adequate for kernel density estimation. The kernel can be calculated as:

$$K(u) = \frac{3}{4}\left(1-u^2\right), \text{for} \quad |u| \leq 1 \tag{3-15}$$

Each particle will be weighted according to the approximated measurement likelihood, provided by the kernel density estimate $\hat{f}_{\mathrm{KDE}}(x)$. With this method, particles can be weighted appropriately in case of time-varying and non-Gaussian measurement noise.

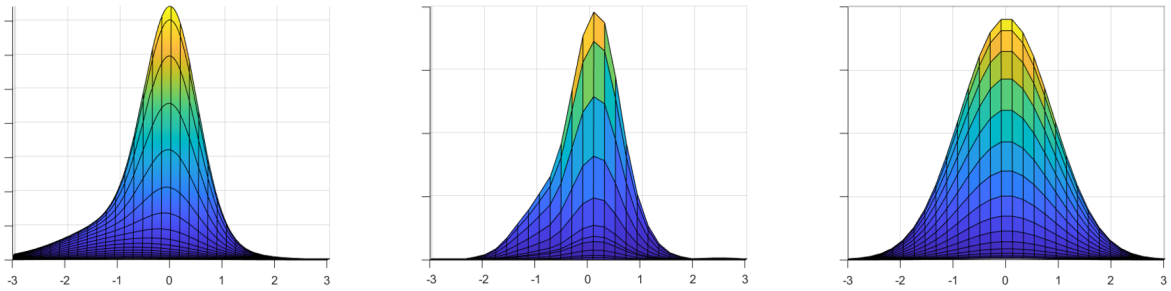

Figure 3-6: Let us say that the two-dimensional Gaussian mixture distribution on the left side represents the true probability distribution of some measurement noise. This distribution has a second mode on the left side of the origin. It has a higher skewness (i.e. asymmetry) and kurtosis (i.e. heavy tails). If the measurement noise were to be approximated by a two-dimensional Gaussian, one obtains the distribution in the right-most plot. If we take $n = 50$ innovations, and use this data to approximate the true distribution with our kernel density estimation algorithm, the center distribution is obtained. The proposed algorithm manages to capture the shape and tailed behavior better than the Gaussian approximation.

Fig. 3-6 illustrates the effect of the kernel density estimation; the shape of the true measurement likelihood can be captured better by the kernel density estimate than by a Gaussian estimate. Thus, weighting particles with the approximated the measurement likelihood $p\left(\mathbf{z}_k \mid \mathbf{x}_k^i\right)$ is more accurate than using a Gaussian approximation. This will, hypothetically, lead to better state estimation.

To verify this, we consider the multivariate nonlinear model of (3-2) once more. The model is now simulated for 50.000 time steps. The simulation period is split up into five sectors (I to V) of 10.000 time steps, each with different statistics for the measurement noise $\mathbf{v}_k$. The noise types are represented mathematically in the parameterized descriptions of Table 3-4, with the parameter values listed in the table caption.

Sector	Time ($\times 10^3$)	Noise model	Mathematical description
Sector I	0-10.000	Gaussian	$\mathbf{v}_k \sim N\left(0, \sigma_1^2\right)$
Sector II	10.000-20.000	Gaussian mixture	$\mathbf{v}_k \sim (1-\alpha_2)\, N(\mu, \sigma_2^2) + \alpha_2 N\left(\mu, \beta_2 \sigma_2^2\right)$
Sector III	20.000-30.000	Random walk	$\mathbf{v}_k = \alpha_3 \mathbf{v}_{k-1} + \beta_3, \quad \beta_3 \sim N\left(0, \sigma_3^2\right)$
Sector IV	30.000-40.000	Poisson noise	$\mathbf{v}_k \sim \text{Poiss}\left(\lambda_4\right)$
Sector V	40.000-50.000	Gaussian	$\mathbf{v}_k \sim N\left(0, \sigma_5^2\right)$

Table 3-4: The parameter values that are used: $\sigma_1 = 0.5$, $\alpha_2 = 0.5$, $\sigma_2 = 0.5$, $\beta_2 = 10$, $\alpha_3 = 1$, $\sigma_3 = 0.5$, $\lambda_4 = 1$ and $\sigma_5 = 0.5$

For both a standard particle filter and a filter that uses the proposed kernel density estimation algorithm, the development of the estimation error of the first state x_1 are presented in Fig. 3-7 below. Under Gaussian circumstances (sectors I and V), the filters perform similarly. In the sectors with non-Gaussian noise (II, III and IV), the standard particle filter exhibits considerably larger estimation errors.

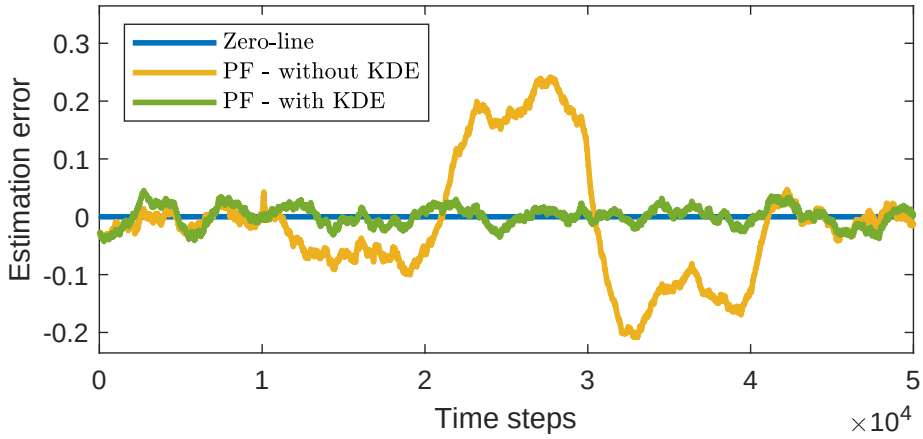

Figure 3-7: Estimation error for the first state of model (3-2) for two particle filter: one with Gaussian measurement noise assumption (yellow) and a particle filter with the measurement likelihood approximated by the adaptive kernel density estimation algorithm (green). In the non-Gaussian sectors (time step 10.000-40.000), the proposed method performs better in terms of forcing the error towards the blue zero-line.

3-5 Proof of convergence

This section will show that in order to prove convergence of the proposed hybrid filter, there are three elements required: cubature Kalman filter convergence (Subsection 3-5-1), particle filter convergence (Subsection 3-5-2) and stability of the adaptive particle weighting scheme (Subsection 3-5-3). Definitions for filter convergence are different for the cubature Kalman and the particle filter. Therefore, any convergence definitions will be introduced in the corresponding subsections.

At first, stability conditions regarding the considered system are placed outside the scope of this section. That is, it is assumed that the system we apply our filtering algorithm on, is a system that has the properties listed in assumption 1 and assumption 2 below.

Assumption 1 *Consider the following non-autonomous, nonlinear stochastic system model with additive process noise* $\mathbf{w}_k$*, measurement noise* $\mathbf{v}_k$*:*

$$\begin{aligned} \mathbf{x}_k &= f\left(\mathbf{x}_{k-1}, \mathbf{u}_k\right) + \mathbf{w}_k \\ \mathbf{y}_k &= h\left(\mathbf{x}_k\right) + \mathbf{v}_k \end{aligned} \tag{3-16}$$

The system is N-locally uniformly rank observable. The functions f *and* h *are differentiable. Matrices* $\mathbf{F}_k^{-1}$ *and* $\mathbf{H}_k^{-1}$ *exist, and there also exist real numbers* $f_{\max} > 0$ *and* $h_{\min} > 0$*, such that (3-17) below holds.*

$$\begin{aligned} \|\mathbf{F}_k\| \leq f_{\max}, \quad \mathbf{F}_k &= \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k} \\ \|\mathbf{H}_k\| \leq h_{\max}, \quad \mathbf{H}_k &= \left.\frac{\partial h}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k-1|k-1}} \end{aligned} \tag{3-17}$$

Assumption 2 *The system (3-16) is a Markov process, i.e. the probability of a state depends on the previous state only, and the probability of an observation depends on the current state only, as shown in (3-18).*

$$\begin{aligned} \mathbf{x}_k &\sim p\left(\mathbf{x}_k \mid \mathbf{x}_{1:k-1}\right) = p\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}\right) \\ \mathbf{z}_k &\sim p\left(\mathbf{z}_k \mid \mathbf{x}_{1:k}\right) = p\left(\mathbf{z}_k \mid \mathbf{x}_k\right) \end{aligned} \tag{3-18}$$

3-5-1 Cubature Kalman filter convergence

Before we dive into the conditions for cubature Kalman filter convergence, remember that the version in our hybrid filter has two adjustments with respect to the regular algorithm. Our version works with i) an adjusted observation vector $\tilde{\mathbf{z}}_k$ instead of $\mathbf{z}_k$, and ii) adjusted matrices $\tilde{\mathbf{Q}}_k$ and $\tilde{\mathbf{R}}_k$ instead of $\mathbf{Q}$ and $\mathbf{R}$, respectively. The consequences of these adjustments will be brought to light in the following paragraphs. Remark 1 below will be used throughout the proof. This remark shows that our adjusted algorithm does not harm any deterministic properties of the system (3-16) mentioned in Assumption 1.

Remark 1 *Under Assumption 1 and the assumption that the particle filter's estimate is convergent in mean square, the adjusted observation $\tilde{\mathbf{z}}_k$ is bounded with respect to the true states $\mathbf{x}$ at the least, i.e. there exists some real $\mathbf{N}$ such that*

$$|\tilde{\mathbf{z}}_k| = |\left(1 - \boldsymbol{\mu}_k\right)\mathbf{z}_k + \boldsymbol{\mu}_k h\left(\mathbf{z}_k^{\mathrm{pf}}\right)| \leq \mathbf{N}, \quad \forall\, \mathbf{x} \in \mathbb{R}^n \tag{3-19}$$

Then, for $\boldsymbol{\mu}_k \in [0, 1)$, the deterministic stability properties of the measurement equation of a given nonlinear system are not affected by the adjustment. Only the stochastic stability properties, harvested in the measurement noise term $\mathbf{v}_k$, change:

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \quad \Rightarrow \quad \tilde{\mathbf{z}}_k = (1 - \boldsymbol{\mu}_k)h(\mathbf{x}_k) + \tilde{\mathbf{v}}_k \tag{3-20}$$

Thus, by proving that $\mathbf{v}_k$ and $\tilde{\mathbf{v}}_k$ have similar properties, it is proven that the deterministic stability properties of Assumption 1 and 2 are not affected by the adjustments of our algorithm.

Throughout the following five conditions for the cubature Kalman filters convergence, we will indicate which elements of the proofs are affected by our adjustments, and how we can prove that these adjustments do not harm the hybrid filters' convergence.

1. **Assumption 3** *The initial estimation error is assumed to be bounded, i.e. $E\left\{\|\hat{\mathbf{x}}_0\|^2\right\} \leq \epsilon$ for small real constant $\epsilon > 0$. Besides, we assume there exists a bound $\delta > 0$ such that $\left\|\mathbf{w}_k^{\mathrm{T}}\mathbf{w}_k\right\| \leq \delta$, $\left\|\mathbf{v}_k^{\mathrm{T}}\mathbf{v}_k\right\| \leq \delta$.*

   If the particle filter is convergent, it has a finite estimate and there will also exist a bound δ such that the adjusted noise term $\tilde{\mathbf{v}}_k$ of (3-20) is also bounded, i.e. $\left\|\tilde{\mathbf{v}}_k^{\mathrm{T}}\tilde{\mathbf{v}}_k\right\| \leq \delta$. This assumption will therefore not be affected by our hybrid structure, given that the particle filter is convergent. Thus, Remark 1 is valid.

2. The covariance matrix $\mathbf{P}_k$ must be bounded. This holds when there exist positive scalars $\theta_{\min}$ and $\theta_{\max}$ such that the following holds:

$$\theta_{\min}\mathbf{I}_n < \mathbf{P}_k^{-1} < \theta_{\max}\mathbf{I}_n \tag{3-21}$$

To obtain this, the system should be N-locally uniformly rank observable [8].

Theorem 1 *N-locally uniformly rank observable implicates that there exists an integer* $N \geq 1$ *such that*

$$\operatorname{rank} \frac{\partial}{\partial \mathbf{x}} \left(\begin{array}{c} h_{\mathbf{u}_k}(\mathbf{x}) \\ h_{\mathbf{u}_{k+1}} \circ f_{\mathbf{u}_k}(\mathbf{x}) \\ \vdots \\ h_{\mathbf{u}_{k+N-1}} \circ f_{\mathbf{u}_{k+N-2}} \circ \cdots \circ f_{\mathbf{u}_k}(\mathbf{x}) \end{array} \right) \Bigg|_{\mathbf{x}=\mathbf{x}_k} = n \tag{3-22}$$

for all $\mathbf{x}_k \in K$ *and* N*-tuple of controls* $(\mathbf{u}_k, \cdots, \mathbf{u}_{k+N-1}) \in U$*, in which* K *and* U *are two compact subsets of* $\mathbb{R}^n$ *and* $\mathbb{R}^r$*, respectively.* [13]

This is a deterministic system property that falls under Assumption 1, which is not lost by the adjustments made to our cubature Kalman filter (see (3-20) of Remark 1). Thereby, this conditions is fulfilled.

3. The matrices $\mathbf{F}_k$ and $\mathbf{H}_k$ are uniformly bounded, and the inverse $\mathbf{F}_k^{-1}$ exists. We have assumed these deterministic properties earlier on (Assumption 1) and they are not affected by the hybrid architecture (Remark 1). This condition is therefore fulfilled.

4. For the square-root configuration, the instrumental matrices $\tilde{\mathbf{Q}}_k$ and $\tilde{\mathbf{R}}_k$ need to be positive definite at all times, in order to perform the triangularization step. In other words, there should exist real constants $\hat{q}_{\min} > 0$ and $\hat{r}_{\min} > 0$, such that

$$\begin{aligned} \hat{q}_{\min}\mathbf{I} &\leq \tilde{\mathbf{Q}}_k \\ \hat{r}_{\min}\mathbf{I} &\leq \tilde{\mathbf{R}}_k \end{aligned} \tag{3-23}$$

This can be achieved by setting both instrumental matrices as diagonal matrices with only positive diagonal entries. The term $\eta\mathbf{I}_n$ of (3-10) guarantees this condition for $\tilde{\mathbf{Q}}_k$. A constant, diagonal, positive definite matrix $\mathbf{R}$ is pre-defined in our filtering algorithm. However, since our algorithm uses $\tilde{\mathbf{R}}_k = (1 - \boldsymbol{\mu}_k)\mathbf{R}$ instead of $\mathbf{R}$, this condition might be violated for values close to $\mu = 1$. Therefore, we put a bound on μ: $\mu \leq 1 - \gamma$, with γ close to 0, e.g. $\gamma = 0.01$, such that $\hat{r}_{\min}\mathbf{I} \leq \tilde{\mathbf{R}}_k$, $\forall\, \mu \in [0, 1 - \gamma]$. As a result, both conditions of (3-23) are met.

5. The previous conditions assure boundedness, but not necessarily convergence. A Lyapunov argument is used to prove that the estimation error is locally asymptotically convergent in mean square.

Let us define $\tilde{\mathbf{x}}_{k|k-1}$ as prediction error (i.e. $\tilde{\mathbf{x}}_{k|k-1} = \mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}$) and $\tilde{\mathbf{x}}_k$ as estimation error ($\tilde{\mathbf{x}}_{k|k} = \mathbf{x}_k - \hat{\mathbf{x}}_{k|k}$). The linearized version of this equation, with $\mathbf{F}_k$ from (3-17), then becomes as follows:

$$\tilde{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \tilde{\mathbf{x}}_{k-1} + \mathbf{w}_k \tag{3-24}$$

This error equation can be adjusted to (3-25) below, in which the unknown diagonal matrix $\boldsymbol{\beta}_k = \text{diag}\left(\beta_{1,k} \cdots \beta_{n,k}\right)$ captures the first-order linearization errors and the errors caused by noise:

$$\tilde{\mathbf{x}}_{k|k-1} = \boldsymbol{\beta}_k \mathbf{F}_k \tilde{\mathbf{x}}_{k-1} \tag{3-25}$$

Following the example of [13], a Lyapunov function is proposed:

$$V_k(\mathbf{x}) = \tilde{\mathbf{x}}_k^T \mathbf{P}_k^{-1} \tilde{\mathbf{x}}_k \tag{3-26}$$

Note that this Lyapunov function must have the properties of Theorem 2 below.

Theorem 2 *Let $x = 0$ be an equilibrium point for the autonomous system $x_k = f(x_{k-1})$ where $f : D \to \mathbb{R}^n$ is locally Lipschitz in $D \subset \mathbb{R}^n$ and $0 \in D$. Suppose there exists a continuous function $V(x) : D \to \mathbb{R}$, and there exists a $\lambda \in (0, 1]$ such that*

$$\begin{aligned} &V_k(0) = 0 \\ &V_k(x) > 0, \forall\, x \in D - \{0\} \\ &V_{k+1}(x) - (1-\lambda) V_k(x) \leq 0 \in D \end{aligned} \tag{3-27}$$

Then $V(x)$ is decreasing, and the equilibrium $x = 0$ is stable. Moreover, if there exists a $\lambda \in (0, 1)$ such that

$$V_{k+1}(x) - (1-\lambda) V_k(x) < 0, \forall\, x \in D \tag{3-28}$$

then $V(x)$ is strictly decreasing, and the equilibrium $x = 0$ is asymptotically stable.

When using the error dynamics of (3-26), we could use Theorem 2 with $\tilde{\mathbf{x}}_{k|k-1} = 0$ as equilibrium point. The first condition of (3-27) is satisfied for the Lyapunov function of (3-26). The second condition of (3-27) of Theorem 2 is valid for positive definite $\mathbf{P}_k^{-1}$. We know that for an arbitrary non-singular, positive definite diagonal matrix K, all its eigenvalues are positive. The eigenvalues of the inverse K^{-1} can then be calculated by $\lambda_i\left(K^{-1}\right) = \frac{1}{\lambda_i(K)}$, i.e. the inverse of the eigenvalues, which remains positive. Therefore, $\mathbf{P}_k^{-1}$ is positive definite as well. Since we are using a square-root form for $\mathbf{P}_k$, it is always positive definite and the second condition of (3-27) can be guaranteed. Conditions for the square-root form are included in the fourth element of this proof.

The last condition of (3-27) can be verified by performing several substitutions and intermediate steps [77]. The last condition of (3-27) then reduces to:

$$\overline{\sigma}\left[\boldsymbol{\beta}_k\right]^2 \leq \underline{\sigma}\left[\boldsymbol{\beta}_k\right]^2 \frac{\underline{\sigma}\left[\mathbf{C}_k\right]^2 \underline{\sigma}\left[\mathbf{P}_{k|k-1}\right]}{\overline{\sigma}\left[\mathbf{P}_{zz,k}\right]} + (1-\lambda) \frac{\underline{\sigma}\left[\mathbf{P}_{k-1}^{-1}\right] \underline{\sigma}\left[\mathbf{P}_{k|k-1}\right]}{\overline{\sigma}\left[\mathbf{F}_k\right]^2} \tag{3-29}$$

in which $\underline{\sigma}$ and $\overline{\sigma}$ are the minimum and maximum singular values, and λ must be within $\lambda \in (0,1)$ for asymptotic stability. Condition (3-29) can be guaranteed when an appropriate (maximum) value for $\underline{\sigma}\left[\mathbf{P}_{k|k-1}\right]$ can be realized. Substituting $\mathbf{P}_{k|k-1}$ from the equations of the cubature Kalman filtering algorithm, this term becomes as follows:

$$\underline{\sigma}\left[\mathbf{P}_{k|k-1}\right] = \underline{\sigma}\left[\frac{1}{m}\sum_{i=1}^{m} \boldsymbol{X}^{*}_{i,k|k-1}\boldsymbol{X}^{*T}_{i,k|k-1} - \hat{\mathbf{x}}_{k|k-1}\hat{\mathbf{x}}^{T}_{k|k-1} + \tilde{\mathbf{Q}}_k\right] \tag{3-30}$$

To adhere to (3-29), matrix $\tilde{\mathbf{Q}}_k$ should be chosen such that (3-30) is maximized [77]. Following [13], the process covariance of the measurable states can be defined as in (3-31). This instrumental matrix $\tilde{\mathbf{Q}}_k$ is based on the innovation $\tilde{\mathbf{y}}_k = \mathbf{z}_k - h\left(\hat{\mathbf{x}}_{k|k-1}\right)$ such that in (3-28), $\lambda < 1$ at all times:

$$\tilde{\mathbf{Q}}_k = \rho\tilde{\mathbf{y}}^{T}_k\tilde{\mathbf{y}}_k\mathbf{I}_n + \eta\mathbf{I}_n \tag{3-31}$$

with $\rho > 0$ chosen large enough and $\eta > 0$ chosen small enough. This particular choice for the instrumental $\tilde{\mathbf{Q}}_k$ ensures that in case of large estimation errors, the condition (3-29) is still valid, since the upper bound of the right side of (3-29) increases because of the term $\tilde{\mathbf{y}}^{T}_k\tilde{\mathbf{y}}_k$. On the other hand, in case of small estimation errors, the term $\tilde{\mathbf{y}}^{T}_k\tilde{\mathbf{y}}_k$ converges to zero and the small term η determines the process noise covariance. With such instrumental $\tilde{\mathbf{Q}}_k$, (3-29) will not be violated and Lyapunov decrease is proven [13], regardless of the choice for $\tilde{\mathbf{R}}_k$.

The third condition of (3-27) is now fulfilled; $V_k(x)$ is a strictly decreasing sequence. This completes the fifth and last condition for this proof. We will now prove convergence of the cubature Kalman filter.

Since $\mathbf{P}_k$ is a bounded matrix (3-21), and $0 < \theta_{\min}\mathbf{I} \leq \mathbf{P}^{-1}_k$ (3-21), we can perform the steps below:

$$\begin{aligned} &0 \leq \theta_{\min}\tilde{\mathbf{x}}^{T}_k\tilde{\mathbf{x}}_k \leq \tilde{\mathbf{x}}^{T}_k\mathbf{P}^{-1}_k\tilde{\mathbf{x}}_k = V_k(x) \leq (1-\lambda)^k V_0(x) \\ \Rightarrow\ &0 \leq \theta_{\min}\lim_{k\to\infty}\left(\tilde{\mathbf{x}}^{T}_k\tilde{\mathbf{x}}_k\right) \leq \lim_{k\to\infty}\left(V_k(x)\right) \leq V_0(x)\lim_{k\to\infty}(1-\lambda)^k = 0 \end{aligned} \tag{3-32}$$

This implicates that the error converges to zero:

$$\theta_{\min}\lim_{k\to\infty}\left(\tilde{\mathbf{x}}^{T}_k\tilde{\mathbf{x}}_k\right) \Rightarrow \lim_{k\to\infty}\tilde{\mathbf{x}}_k = \lim_{k\to\infty}\left(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}\right) = 0 \tag{3-33}$$

The proof that the cubature Kalman filter is locally asymptotically convergent is now complete. □

3-5-2 Particle filter convergence

Remark 1 requires the particle filter to be convergent. In this subsection, we will show that the convergence of the particle filter mainly depends on the choice for the proposal distribution.

For a particle filter to work properly, the support of the proposal distribution must include the true posterior at all times. We will show how we arrive at this conclusion.

First, let $B(\mathbb{R}^n)$ be the space of all Borel measurable test functions f_k on the state-space $\mathbb{R}^n$. A particle filter estimates the true posterior $p(\mathbf{x}_k \mid \mathbf{z}_k)$ by a weighted set of particles $\{(w_k^i, \mathbf{x}_k^i) : i = 1, \ldots, N\}$ as follows:

$$p(\mathbf{x}_k \mid \mathbf{z}_k) = \int f_k(\mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{z}_k)\, d\mathbf{x}_k \approx \sum_{i=1}^{N_p} w_k^i f_k\left(\mathbf{x}_k^i\right) \tag{3-34}$$

Now let us have a look at Theorem 3.

Theorem 3 *Let $\{\hat{\mathbf{G}}_{k|k}^i\}_{i=1}^{N_p}$ be a set of random approximations of $\mathbf{G}_{k|k}$, then the mean of the approximations $\hat{\mathbf{G}}_{k|k}$ converges to $\mathbf{G}_{k|k}$ if, for any Borel bounded measurable function $f_k \in B(\mathbb{R}^n)$, the mean square error converges to zero for $N_p \to \infty$:*

$$\lim_{N_p \to \infty} E\left[\left(\hat{\mathbf{G}}_{k|k} - \mathbf{G}_{k|k}\right)^2\right] = 0 \tag{3-35}$$

In other words, a particle filter is said to converge if its approximation becomes exact (in mean square sense) in the limit of an infinite number of particles. We will show that this result holds for the particle filter in our hybrid architecture. To make Theorem 3 applicable, let us say $\mathbf{G}_{k|k} = \int f_k(\mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{z}_k) d\mathbf{x}_k$ and $\|f_k\| = \sup |f_k(\mathbf{x}_k)|$.

Theorem 4 *Consider a Markov state-space model in a stochastic form:*

$$\begin{aligned} \mathbf{x}_k &\sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) \\ \mathbf{z}_k &\sim p(\mathbf{z}_k \mid \mathbf{x}_k) \end{aligned} \tag{3-36}$$

[71] If these two densities are bounded and the unnormalized importance weights w_k^i are upper bounded for any $(\mathbf{x}_{k-1}, \mathbf{z}_k)$, then, for all $k \geq 0$, there exists c_k independent of N_p, such that for any $f_k \in B\left(\mathbb{R}^{n_x \times (k+1)}\right)$:

$$E\left[\left(\hat{\mathbf{G}}_{k|k} - \mathbf{G}_{k|k}\right)^2\right] = E\left[\left(\frac{1}{N_p}\sum_{i=1}^{N_p} w_k^i f_k\left(\mathbf{x}_k^i\right) - \int f_k(\mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{z}_k) d\mathbf{x}_k\right)^2\right] \leq c_k \frac{\|f_k\|^2}{N_p} \tag{3-37}$$

What this theorem says, is that the particle filter admits a mean square error of order $1/N_p$. Thus, the squared error between the estimated state and true state vanishes when $N_p \to \infty$. The expectation at the left-hand side of (3-37) is a representation of the degree of randomness that the particle filter's sampling introduces. We now know that if Theorem 4 can be proven, Theorem 3 is valid and the proof of convergence is complete.

A noteworthy aspect; it seems as if the rate of convergence $1/N_p$ is independent of the number of states n, which would mean that particle filters beat the curse of dimensionality [17]. However, N_p might depend on c_k, which might depend on n. So, for high dimensional state-spaces, the curse of dimensionality might still occur.

Now let us look at the conditions for Theorem 4. If $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$ is bounded, $p(\mathbf{z}_k \mid \mathbf{x}_k)$ is bounded and w_k^i is bounded, the particle filter is convergent. Assuming that we are dealing with a stable system (Assumptions 1, 2 and Remark 1), proving $\|w_k^i\| \leq B_w$, with B as any positive real number, would complete the proof. Usually, a particle weight has the following form:

$$w_k^i \propto w_{k-1}^j \cdot \frac{p(\mathbf{z}_k \mid \mathbf{x}_k^i)\, p\left(\mathbf{x}_k^i \mid \mathbf{x}_{k-1}^j\right)}{q\left(\mathbf{x}_k^i \mid \mathbf{x}_{k-1}^j, \mathbf{z}_k\right)} \tag{3-38}$$

with i as the current particle index and j as the previous particle index. An upper bound on this weight can easily be accomplished as follows:

$$\begin{aligned} p(\mathbf{z}_k \mid \mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) &\leq \mathbf{B}_p, \quad \mathbf{B}_p > 0 \\ \mathbf{B}_q &\leq q(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{z}_k), \quad \mathbf{B}_q > 0 \end{aligned} \tag{3-39}$$

The first condition indicates that $p(\mathbf{z}_k \mid \mathbf{x}_k)$ and $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$ should have at least some variance. The only case in which this is not true is when the probability distributions are a Dirac delta function. The state-space equations would then be deterministic, and one could easily set the corresponding probability to 1 instead (i.e. omit it from the weight equation (3-38)) to solve this. That would mean the first condition can be satisfied by looking at the system properties. The next Subsection will elaborate on the boundedness of the measurement likelihood. The second condition says there must be a lower bound on the proposal distribution. Theoretically, any Gaussian proposal distribution would suffice, since its tails converge to zero, but never reach it. Later on, we will opt for a Gaussian proposal distribution. Thereby, the conditions of (3-39) are fulfilled, and the particle filter converge proof is complete. □

Choice for proposal distribution This proof seems to be counterintuitive, since boundedness is much different from convergence. Imagine a realistic situation in which a lower bounded proposal distribution and upper bounded observation likelihood do not match well (i.e. no overlapping tails), and a finite number of particles will be sampled. Then almost all particles will fall outside the measurement likelihood, resulting in all weights being very close to zero (Fig. 3-8). The estimate then drifts away and will not converge. However, remember the definition of convergence of Theorem 3; it holds for $N_p \to \infty$. Logically, when an infinite number of samples is drawn, there will also be samples at the very outer edges of the (Gaussian) proposal distribution that might fall within the support of the measurement likelihood. Therefore, let it be noted that there is a considerable difference between a particle filter's convergence as stated in Theorem 3, and convergence in circumstances that are computationally viable.

Reviewing the problem more intuitively; we do not want all weights to be zero, infinite or equal to each other, as can be seen in Fig. 3-8. Ideally, the particles have varying weights, indicating that little information about the true posterior is lost. In other words, the proposal distribution must be a good approximation of the true posterior. In case of small process noise, this is the same as that the proposal distribution must be close in support (and preferably also shape) to the measurement noise distribution.

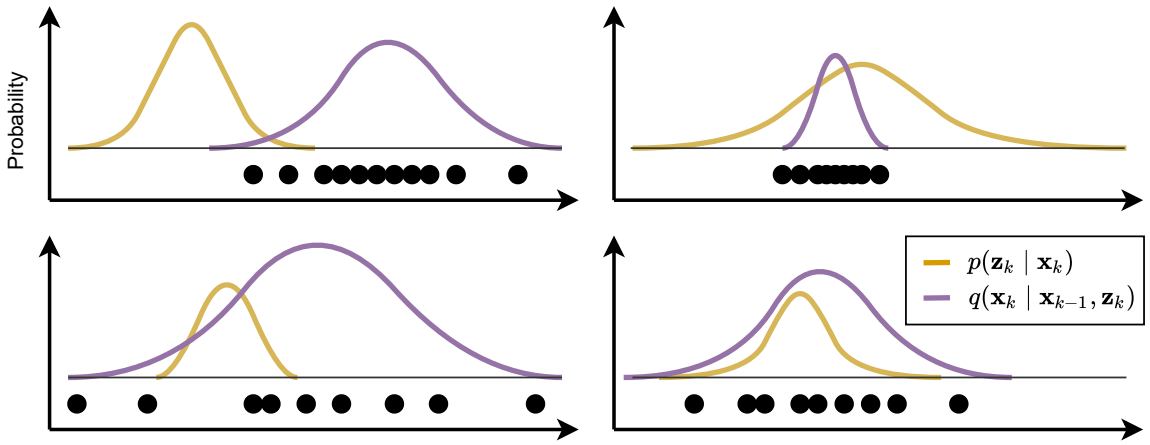

Figure 3-8: Four configurations for a proposal distribution and measurement likelihood. Top left: the proposal distribution has a small support that does not cover the measurement likelihood. Only two particles get assigned a weight, and it is impossible that this weighted average approaches the measurement likelihood mean. Top right: the proposal distribution is too narrow. All particles get assigned a weight of similar order, and the weighted particle mean will not be a good estimate of the observation mean. Bottom left: the proposal distribution has a wide support, but only a few particles are weighted. This leads to inaccurate estimation. Bottom right: the proposal distribution is similar in shape and support as the measurement likelihood. The weighted average of the particles will be a good approximation of the measurement likelihood mean.

One option is to use the prior $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$ as proposal distribution. This is a Gaussian that includes the effect of $\tilde{\mathbf{Q}}_k$. Remember that in our case, this covariance matrix scales with the innovation. Therefore, it will have a wider support in case a measurement does not fall near the proposal distribution (lower left graph of Fig. 3-8). There will then still be particles that get assigned a weight. Later on in Subsection 4-2-2, we will elaborate on the choice for our proposal distribution.

3-5-3 Stability of the adaptive particle weighting scheme

In this part of the proof, we will show that the observation likelihood $p(\mathbf{z}_k \mid \mathbf{x}_k)$ has an upper bounded probability density function, and has tails with similar properties to that of a continuous distribution.

Standard, continuous Gaussian density functions have a value (i.e. probability) for the whole space of the states $\mathbf{x} \in \mathbb{R}^n$. The tails of such a probability density function $f(x)$ extend indefinitely, and in these regions, the probability under the curve gets infinitesimally small:

$$\lim_{x \to \infty} f(x) \to 0 \tag{3-40}$$

In our case, however, a discrete Epanechnikov kernel is fitted to innovation data points. This density approximation is built up of bins with a certain bin width and a certain probability. Such a discrete nature would mean that we would have to compute likelihoods for an infinite number of bins if we want to capture the endless tails of continuous density functions. We

solve this by extending the outermost bins of our constructed probability density function $f(x)$ at the 3σ bound:

$$f(x) = \begin{cases} f(\mu_x - 3\sigma_x), & x \leq \mu_x - 3\sigma_x, \\ f(\mu_x + 3\sigma_x), & x \geq \mu_x + 3\sigma_x \end{cases} \tag{3-41}$$

For standard Gaussian distributions, 1 in 370 samples ($\approx 0.3\%$) will fall out of this interval and will be therefore be weighted inaccurately. In case of heavy-tailed distributions, it could be slightly more. This does not weight up against the computational win that we gain with these bounds. Filter convergence is still kept, since (i) we prevent the likelihood $p(\mathbf{z}_k \mid \mathbf{x}_k)$ and therefore the weight of (3-38) from taking value zero, and (ii) we are setting a lower bound for the innovation variance $\sigma_{\tilde{y}}^2 \geq \kappa$ for the rare case that all samples take exactly the same value ($\sigma_{\tilde{y}}^2 = 0$). In this highly unlikely case, the likelihood $p(\mathbf{z}_k \mid \mathbf{x}_k)$ has a Dirac delta function as probability density function, which is not upper bounded. This would be in conflict with the conditions stated Theorem 3. With these bounds, we prove that our adaptive particle weighting scheme does not tamper with any stability or convergence properties. □

3-5-4 Stability of the full hybrid architecture

Now we will briefly link the previous elements to obtain a full proof of convergence for our hybrid filtering architecture.

At first, several assumptions are made on the stability of a given system. Given these assumptions, Subsection 3-5-1 has shown that our version of the cubature Kalman filter converges, if the particle filter estimate converges (Remark 1). Subsection 3-5-2 has indicated that for $N_p \to \infty$, an upper bounded $p(\mathbf{z}_k \mid \mathbf{x}_k)$ and $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$ and a lower bounded $q(\mathbf{x}_k \mid \mathbf{x}_{k-1})$, the particle filter estimate converges. The upperbounds are guaranteed for any stochastic system, and can be replaced by 1 for deterministic transition- or measurement equations. The lower bound of the proposal distribution is, theoretically, guaranteed for any Gaussian. Thereby, all conditions for particle filter convergence are met, indicating that the full architecture is convergent.

For finite N_p, we have seen that numerical issues might arise when an inappropriate proposal distribution is selected. Therefore, the choice for a proposal distribution is important for practical convergence. Since our proposal distribution is adaptive through the influence of adaptive matrix $\tilde{\mathbf{Q}}_k$, convergence can be realized in practise as well.

3-6 Results for a benchmark example

With both the adaptive kernel density estimation and switching features implemented, a simulation with the noise settings of Table 3-4 is performed. To illustrate the effect of the switching, the development of the parameter μ for the two states of model (3-2) is shown in Fig. 3-9 below. For this simulation, the safety factor γ of (3-7) is set as $\gamma = 1 \cdot 10^{-2}$.

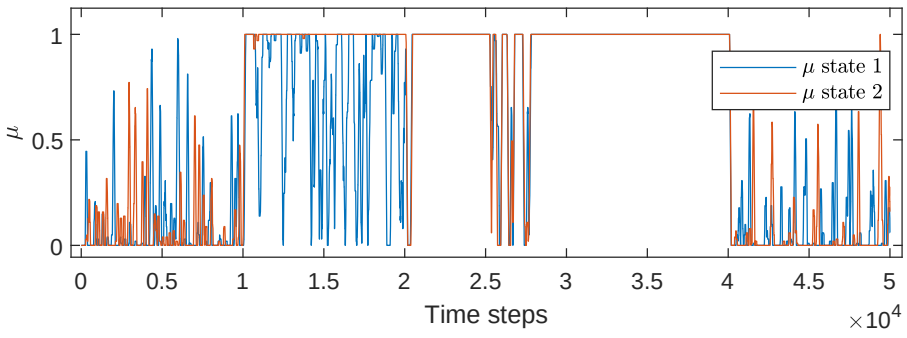

Figure 3-9: For the simulation settings of Table 3-4: development of the switching parameter μ for both states of (3-2). Gaussian sectors I and V show small values for μ, while non-Gaussian sectors II-IV show values close to 1.

As can be seen, the parameter adapts well to the Gaussianity of the measurement noise. The average values $\bar{\mu}$ for both states are given in Table 3-5 below. Clearly, the sectors with Gaussian noise (I and V) have lower values for μ, as where the sectors with non-Gaussian noise have high values for μ.

Sector	I	II	III	IV	V
$\bar{\mu}$ state 1	0.08	0.63	0.82	0.99	0.10
$\bar{\mu}$ state 2	0.07	0.97	0.98	0.99	0.10

Table 3-5: Mean values $\bar{\mu}$ for each of the five sectors of the simulation.

The development of the estimation error of the first state is given in Fig. 3-10 below. Especially under the non-Gaussian circumstances of sectors II, III and IV, the proposed filtering method outperforms the other method in terms of minimal errors. An enlarged version of each of the sectors is provided in the Appendix B-1. Also, the mean root-mean-square error (RMSE)s of 25 separate Monte Carlo simulations is provided in Appendix B-1.

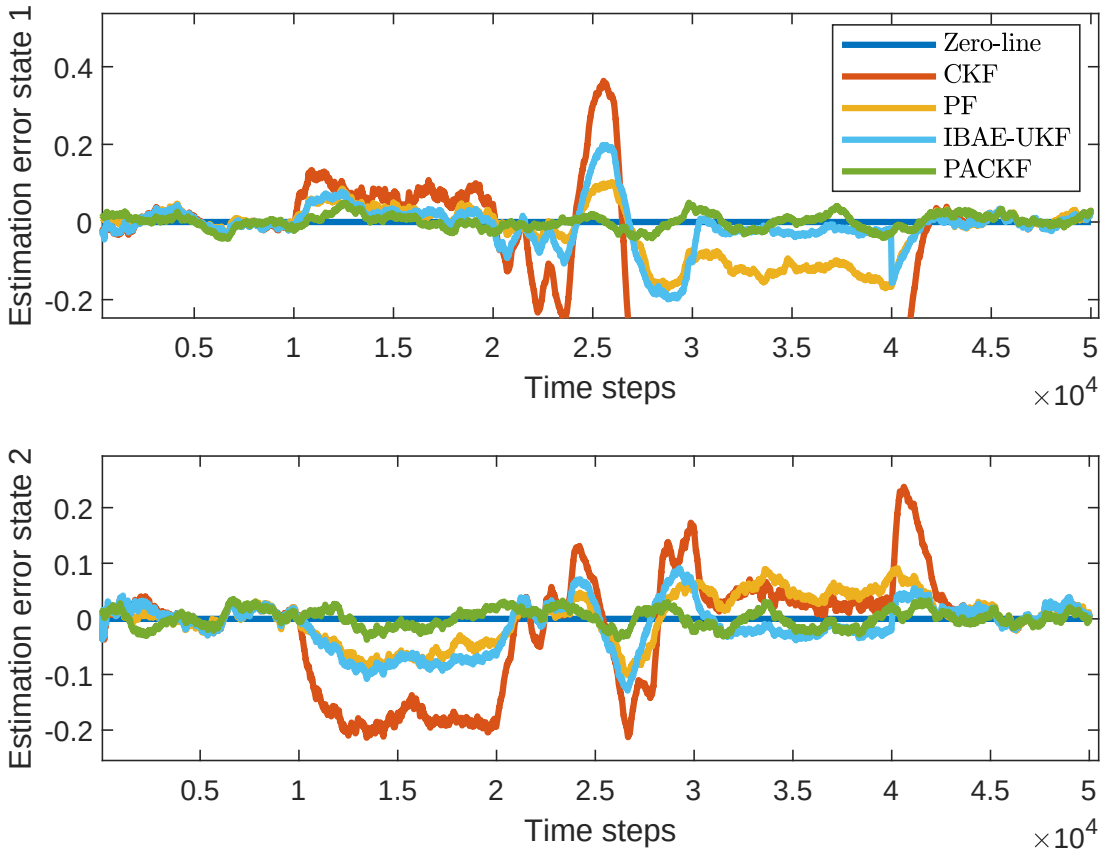

Figure 3-10: Development of the estimation errors for state 1 and 2 of a cubature Kalman filter (CKF), particle filter (PF), innovation-based adaptive estimation unscented Kalman filter (IBAE-UKF) and the proposed adaptive particle-aided cubature Kalman filter (PACKF).

3-7 Summary

In this chapter, a new hybrid filtering algorithm is proposed. Because of this hybrid architecture, the proposed adaptive particle-aided cubature Kalman filtering algorithm works under both Gaussian and non-Gaussian measurement noise, given that the process noise is relatively small. The contributions of this proposition are summarized by the following notes:

- Particle filters perform better in the non-Gaussian domain than nonlinear variants of the Kalman filter, since they are not bound to Gaussian distributions for representing stochastic variables.
- A switching architecture allows for usage of the particle filter under non-Gaussian measurement noise, while the cubature Kalman filter is used in case of Gaussian measurement noise. The sampling process of the particle filter then does not harm the estimates in the Gaussian case.
- Since the process noise $\mathbf{w}$ is relatively small, the measurement noise $\mathbf{v}$ can be approximated by the innovation sequence $\tilde{\mathbf{y}}$.

- A parameter $\boldsymbol{\mu}$ is used as a measure for Gaussianity. It follows from a Kolmogorov-Smirnov test on the innovation sequence $\tilde{\mathbf{y}}$. It provides a soft switching between a cubature Kalman filter and an adjusted particle filter.
- An innovation-based, adaptive matrix $\tilde{\mathbf{Q}}_k$ is used for two reasons: i) to guarantee convergence of the cubature Kalman filter and ii) to adjust the proposal distribution $q(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{z}_k)$ such that it support encloses that of the likelihood distribution $p(\mathbf{z}_k \mid \mathbf{x}_k)$. The latter will become clear in the next chapter.
- The innovation sequence $\tilde{\mathbf{y}}$ is used in a kernel density estimation algorithm to accurately weight particles with any measurement likelihood distribution $p(\mathbf{z}_k \mid \mathbf{x}_k)$ (Gaussian and non-Gaussian).
- The theoretical convergence of the proposed hybrid filter is proven.

Chapter 4

Localization simulations for performance evaluation

The complete adaptive particle-aided cubature Kalman filter is now applied on the complete loosely coupled scheme for integration of global navigation satellite system (GNSS)- and inertial navigation system (INS) data of Chapter 2. Section 4-1 will outline the data acquisition process, and Section 4-2 will discuss some case-specific tuning processes. The closing Section 4-3 provides the results of the simulations and a compact analysis. Logically, the goal of the GNSS/INS-based localization is to find an accurate location at the hand of a set of GNSS and INS measurements. As a performance metric in achieving this goal, the root-mean-square error (RMSE) between the true position and the filter position estimates is used. The RMSE over N time steps is calculated in the Cartesian, local navigation frame (n-frame):

$$\text{RMSE}_{\text{pos}} = \frac{1}{N}\sum_{k=1}^{N}\sqrt{\left((x_k - \hat{x}_k)^2 + (y_k - \hat{y}_k)^2\right)} \tag{4-1}$$

4-1 Data acquisition

The ground-truth data is generated with the model (2-2) and process noise $\mathbf{w} = 0$ for all time steps k. We simulate 4 minutes of driving in east direction, with both the GNSS- and INS update rate at 10Hz. In each simulation, we generate three different inputs, which each will be used for a third of the simulation time. In each of the three periods, the acceleration is set such that the longitudinal speed ranges from 50 km/h to 110 km/h. An arbitrary, mild steering angle ($\delta < 0.004$ rad) is used to resemble highway driving conditions.

GNSS measurement data is generated with (2-4). For the GNSS noise, the simulation period is split up into 7 sectors, each with a different type of measurement noise. A description of these noise types can be found in Table 4-2 below.

Sector	Time period	Noise model	Mathematical description
Sector I	0s-40s	Gaussian	$\mathbf{v}_k \sim N(0, \mathbf{R})$
Sector II	40s-80s	Gaussian mixture	$\mathbf{v}_k \sim 0.9N(0, \mathbf{R}) + 0.1N(0, 300\,\mathbf{R})$
Sector III	80s-120s	Varying variance	$\mathbf{v}_k \sim N(0, \alpha\mathbf{R}), \quad \alpha \in [1, 5]$
Sector IV	120s-160s	Flicker noise	$S(f) \propto \frac{1}{f}$
Sector V	160s-170s	High variance	$\mathbf{v}_k \sim N(0, 100\,\mathbf{R})$,
Sector VI	170s-210s	Random walk	$\mathbf{v}_k = 0.6\,\mathbf{v}_{k-1} + \beta, \quad \beta \sim N(0, \mathbf{R})$
Sector VII	210s-240s	Gaussian	$\mathbf{v}_k \sim N(0, \mathbf{R})$

Table 4-1: Table with settings for the GNSS measurement noise sequence. Note that these correspond to the phenomena that induce non-Gaussian noise of Section 2-4-2.

The INS noise is modelled by means of a Gaussian distribution. The process- and measurement noise covariance matrices $\mathbf{Q}$ and $\mathbf{R}$ are diagonal matrices as described in (2-15) and (2-19) of Chapter 2. The standard errors of the sensors are given in Table 4-2 below.

Standard error type	Symbol	Value	Unit
GNSS position	$\boldsymbol{\sigma}_{\text{pos}}$	4	m
GNSS velocity	$\boldsymbol{\sigma}_{\text{vel}}$	1	m/s
Accelerometer noise	$\boldsymbol{\sigma}_{\text{a}}$	0.03	$\text{mg}\sqrt{\text{Hz}}$
Accelerometer bias in-run stability	$\boldsymbol{\sigma}_{\text{b,a}}$	0.015	mg
Accelerometer bias correlation time constant	$\boldsymbol{\tau}_{\text{a}}$	200	-
Gyroscope noise	$\boldsymbol{\sigma}_{\omega}$	0.006	$^\circ/\text{s}\sqrt{\text{Hz}}$
Gyroscope bias in-run stability	$\boldsymbol{\sigma}_{\text{b},\omega}$	3	$^\circ/\text{h}$
Gyroscope bias correlation time constant	$\boldsymbol{\tau}_{\omega}$	200	-

Table 4-2: These values are based on the sensor specifications highlighted in [3]. These standard errors are valid for any direction (x,y,z-axes and everything int between).

Some first simulations indicate that the sensors are modelled properly. For these simulations, the GNSS measurement data, as well as the INS measurement data, are given in Fig. 4-1 below. As can be seen in the x,y-plane of the graphs, the error between the GNSS and INS builds up over time.

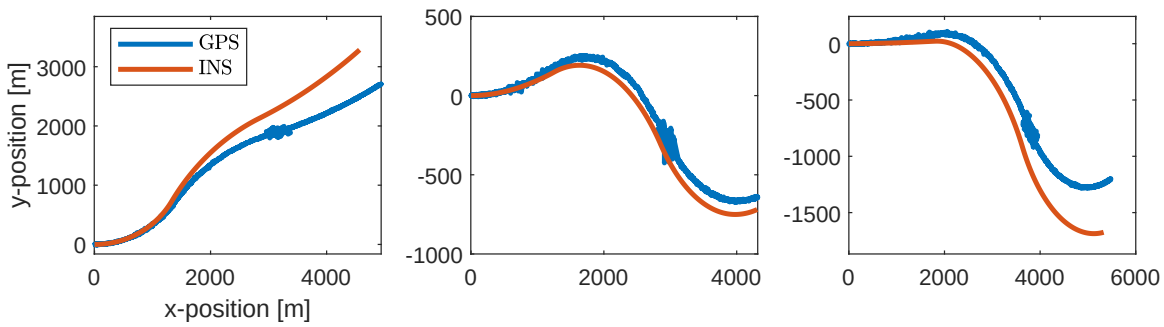

Figure 4-1: In the x,y-plane; GNSS and INS location measurements of three randomly generated paths in east direction.

4-2 Implementation of the algorithm

At first, Subsection 4-2-1 will describe an adjustment to the algorithm that makes it more suitable for periods of GNSS outage. After specifying this adjustment, there are still multiple settings that can be modified within the particle-aided cubature Kalman filtering algorithm architecture. The settings of the particle filter are the most important, since there are still various options for, for example, the number of particles, the proposal distribution and the resampling strategy. It is difficult to determine proper settings for these entities, compared to tuning a single parameter. Therefore, the particle filter settings will be determined first in Subsection 4-2-2. Then, the algorithm's remaining parameters will be tuned in Subsection 4-2-3.

4-2-1 Adjustment for GNSS outages

In case of GNSS outages, the measurements will have extremely high variance, or there might even be no signal at all. As a consequence, our instrumental matrix $\tilde{\mathbf{Q}}_k$, which is based on the innovation $\tilde{\mathbf{y}}_k$, will increase considerably as well. Also, the likelihood distribution $p\left(\mathbf{z}_k \mid \mathbf{x}_k^i\right)$ from the kernel density estimation algorithm (the one that we weight the particles with) will have a wide support. This will lead to large estimation errors, since there is a low particle density around the true states. For short periods of GNSS outage, it is therefore better to completely ignore the measurements, and to fully trust the INS estimates. This can be established as follows. When the estimated measurement noise standard error exceeds a threshold ($\bar{\mathbf{y}} \approx \hat{\boldsymbol{\sigma}} \geq \boldsymbol{\kappa}$), GNSS outage is detected, and the measurements are turned 'off' in the filter (extremely high variance is set). Moreover, the support of the proposal distribution will not grow anymore. The mean of the innovation sequence over the window n, denoted by $\bar{\mathbf{y}}$, is used to detect GNSS outage. When detected, the matrices will be adjusted as follows:

$$\begin{aligned} \tilde{\mathbf{R}}_k &= \begin{cases} \tilde{\mathbf{R}}_k \cdot 1 \cdot 10^{10}, & \text{if } \bar{\mathbf{y}} \geq \boldsymbol{\kappa} \\ \tilde{\mathbf{R}}_k, & \text{otherwise} \end{cases} \\ \tilde{\mathbf{Q}}_k &= \begin{cases} \boldsymbol{\kappa}, & \text{if } \bar{\mathbf{y}} \geq \boldsymbol{\kappa} \\ \tilde{\mathbf{Q}}_k, & \text{otherwise} \end{cases} \end{aligned} \tag{4-2}$$

4-2-2 Particle filter settings

In the convergence proof of Subsection 3-5-2, we underscore the importance of the choice for the proposal distribution. To guarantee particle filter convergence, the support of the proposal distribution must include the observation. This can be achieved by making sure that the proposal distribution is overlapping the observation likelihood as much as possible. Usually, there are no other uncertainties to base a proposal distribution on than the process uncertainty, which is proportional to $\mathbf{Q}$. Therefore, a common choice for the proposal distribution is the prior $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$. However, the INS process uncertainties are very small in our case. Earlier on, Fig. 3-8 has shown the consequences of inappropriate proposal distributions.

When employing the adaptive particle-aided cubature Kalman filtering algorithm for loosely coupled GNSS/INS integration, we have access to the fairly accurate predictions of the cubature Kalman filter before drawing particles for our particle filter. It is very likely that these

predictions fall close to the true posterior. Thus, sampling around the prior seems the obvious choice. The proposal distribution then takes the following form:

$$q(\mathbf{x}_k \mid \mathbf{x}_{k-1}) = N\left(\hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1}\right) \tag{4-3}$$

Regular particle filters resample or re-use particles from the previous time step. This would be a good approach for systems with large process uncertainty. In our specific case, however, it makes more sense to sample around the predictions, since these are already very accurate. Re-using old particles would lead to very high resampling rates, since they quickly become inaccurate compared to the new predictions. One might then as well just resample all particles around the predictions at every time step. This is basically a sequential-importance-resampling (SIR) particle filter with a high threshold N_T for the effective sample size $\hat{N}_{\text{eff}}$:

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N_p} \left(w_k^i\right)^2} \leq N_T \tag{4-4}$$

This threshold is then set as $N_T = N_p$, indicating we are resampling all particles at every time step. The influence of the previous particle weights can then be neglected. The particle weight equation from Table 3-2 then reduces to:

$$w_k^i \propto w_{k-1}^j \cdot \frac{p\left(\mathbf{z}_k \mid \mathbf{x}_k^i\right) p\left(\mathbf{x}_k^i \mid \mathbf{x}_{k-1}^j\right)}{q\left(\mathbf{x}_k^i \mid \mathbf{x}_{k-1}^j, \mathbf{z}_k\right)} \quad \Rightarrow \quad w_k^i \propto p\left(\mathbf{z}_k \mid \mathbf{x}_k^i\right) \tag{4-5}$$

This particular choice for the proposal distribution and resampling strategy has the following advantages and disadvantages:

+ The predicted covariance matrix $\mathbf{P}_{k|k-1}$ partially depends on the instrumental matrix $\tilde{\mathbf{Q}}_k$, which depends on the innovation $\tilde{\mathbf{y}}_k$ (see Table 3-1). This indicates that our proposal distribution adapts to changes in estimation uncertainty. The lower-right proposal distribution of Fig. 3-8 could then be realized at all times.

+ Instead of propagating N_p particles, resulting in N_p function evaluations, there is no function evaluation needed at all. The prediction step of the cubature Kalman filter has already propagated $2n$ cubature points to produce a prediction. As a consequence, the particle filter becomes extremely fast, since no function evaluation is needed. Only the particles have to be weighted.

- In our particle weighting step, the influence of the previous particle weights are neglected. Besides, including the innovation $\tilde{\mathbf{y}}_k$ in $\tilde{\mathbf{Q}}_k$ indicates that we are sensitive to outlier measurements. This might lead to a somewhat noisy particle filter output, since the state estimates can abruptly change. To filter the most aggressive changes, a Savitzky-Golay filter [59] is applied to the last $n = 5$ outputs of the particle filter. This is a highly effective step that fits a curve to the last 5 measurements. From the perspective of the cubature Kalman filter, we pre-process incoming observations with this filtering step. Pre-processing observation data is not uncommon in localization [41, 50].

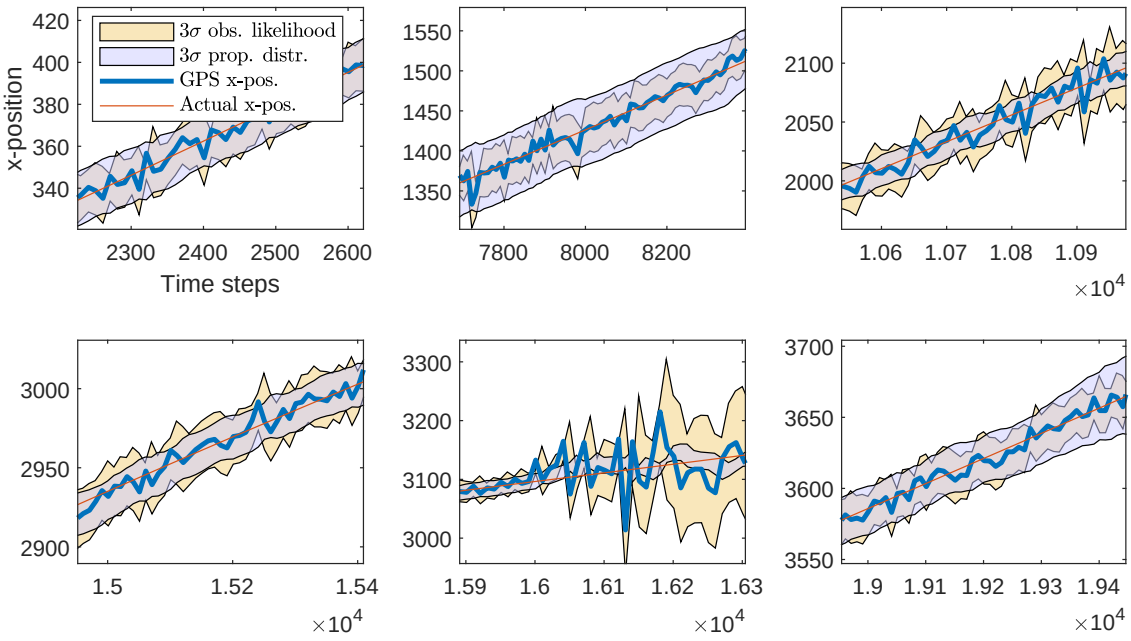

Figure 4-2: Enlarged for sectors I-VI; the 3σ bound of the proposal distribution, the 3σ bound for the observation likelihood (note that this is not necessarily a Gaussian, but the bound gives an indication), a smooth version of the GNSS measurements and the actual ground-truth for the x-position. As can be seen, the support of the proposal distribution varies with time, and manages to follow the support of the measurement likelihood. Only in sector V, i.e. the sector with GNSS outage, the proposal distribution support does increase, but still manages to evolve around the true state. Therefore, the state estimates will still be fairly accurate during GNSS outages.

From the perspective of convergence, this proposal distribution seems to be a good choice. Simulations with other proposal distributions lead, in some cases, to instability, while the proposed choice does not. Fig. 4-2 shows that the distribution leads to the desired result.

After a trade-off between speed and accuracy, we decide to use $N_p = 50$. It appears that larger numbers of particles do not necessarily results in much better performance.

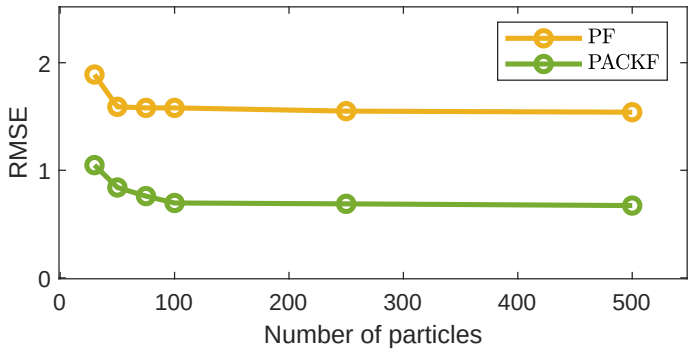

Figure 4-3: For the particle filter and particle-aided cubature Kalman filter: RMSE for a simulation with Gaussian measurement noise for different numbers of particles. Increasing the number of particles after $N_p = 100$ leads to no significant improvement.

4-2-3 Tuning the algorithm

The cubature Kalman algorithm itself does not require much tuning. The additional features that were implemented along the way, do require some tuning. The choices for these tuning parameters are disclosed in Table 4-3 below.

Parameter	Value	Formula
Window size n	$n = 100$	(3-4, 3-6, 3-7, 3-13, 3-14)
Gaussianity test confidence	$\alpha = 0.1$	(3-6)
Safety factor for $\boldsymbol{\mu}$	$\gamma = 0.1$	(3-7)
Scaling factors for $\tilde{\mathbf{Q}}$	$\rho = 0.01, \quad \eta = 0.005$	(3-10)
Bandwidth parameter	$\theta = 3.5$	(3-14)
Density number of bins	$N_b = 100$	-
GNSS outage detection	$\boldsymbol{\kappa}_{\text{pos}} = 50, \quad \boldsymbol{\kappa}_{\text{vel}} = 5$	(4-2)

Table 4-3: Tuning parameters of the proposed algorithm. The values above are found through empirical methods, based on the RMSE outcome.

For the sake of code compactness and efficiency, the window size n is used for both the Kolmogorov-Smirnov test and the kernel density estimation. Longer windows result in a reduced flexibility (i.e. a slower response to changes in noise) and shorter windows lead to inaccurate density estimation. The Gaussianity test confidence factor α must be set with $0.05 \leq \alpha \leq 0.2$. After trying different values, we use $\alpha = 0.1$. The safety factor γ sets an upper bound on $\boldsymbol{\mu}$ ($\boldsymbol{\mu} \leq 0.9$), which smoothens the filter's output and guarantees that the term $(1 - \boldsymbol{\mu}_k)\mathbf{R}$ remains large enough for numerical stability.

After trying different values for ρ and η, the values of Table 4-3 appeared to work best. The lower bound η has shown to guarantee convergence, while this choice for ρ results in proper adaptation to the uncertainty in the system. Besides, this choice results in a suitable proposal distribution, since the distribution indirectly dependents on $\tilde{\mathbf{Q}}_k$. The choice for the kernel bandwidth parameter θ results in smooth, accurate distribution estimates for each of the specified noise types of Table 4-2. The number of bins for the kernel density approximation is traded-off between accuracy and speed. After tuning, the bounds for GNSS outage detection are set to $\boldsymbol{\kappa}_{\text{pos}} = 50$ and $\boldsymbol{\kappa}_{\text{vel}} = 5$.

4-3 Results and analysis

The filters that will be used for evaluation are: a standard cubature Kalman filter (CKF), a particle filter (PF), a cubature Kalman filter-aided particle filter / cubature particle filter (CKFAPF), an innovation-based adaptive estimation unscented Kalman filter (IBAE-UKF) and the proposed adaptive particle-aided cubature Kalman filter (PACKF). The particle filter has the exact same settings as the one in the proposed method; the proposal distribution of (4-3), full resampling at every time step, the same number of particles and an additional Savitzky-Golay filter. The cubature particle filter has a proposal distribution that is provided by a cubature Kalman filter. Thus, in this case, the proposal distribution also includes information on the latest observation.

4-3-1 Localization errors

For sectors I-VI, the localization errors of the x-position and y-position of the local n-frame are presented in Fig. 4-4.

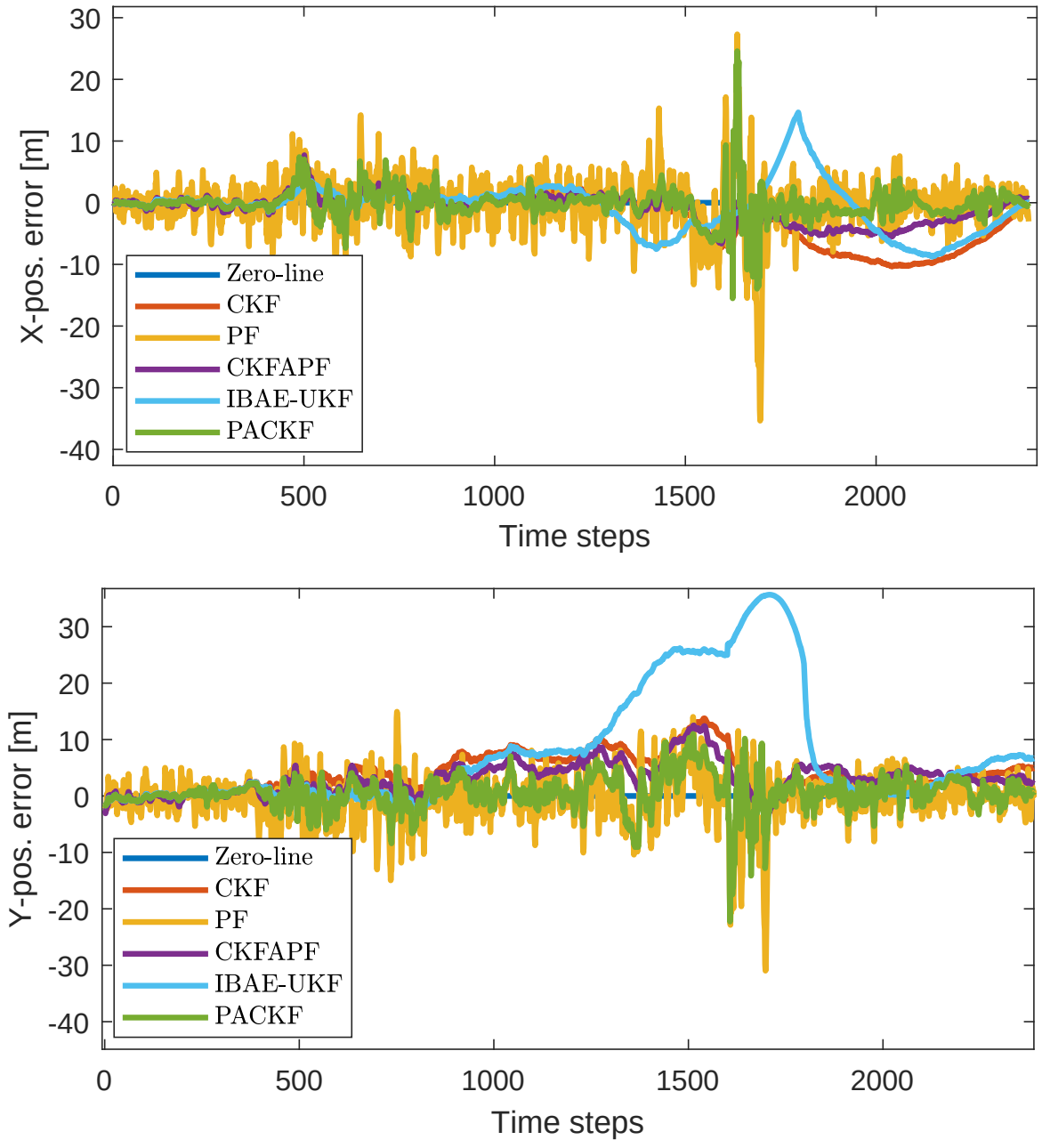

Figure 4-4: The x-position estimation errors for several filtering methods. Though the algorithms perform similarly in earlier sectors, there is a clear difference in performance in the later sectors. The sector with GNSS outage seems, however, still problematic for the proposed filtering method.

The particle filter estimate and, to a lesser extent, the particle-aided cubature Kalman filter estimate, stand out by their noisy character. It is suspected that this is a result of our choice for the proposal distribution and resampling strategy. After all, we do not re-use any knowledge from the last time step. So the final estimate is sensitive to the current measurements. We see, however, that lower thresholds N_T lead to higher inaccuracy, since particle degeneracy happens rather quickly. The mean of the estimation errors of the proposed method appears to be the smallest with these settings. In Appendix B-2, an enlarged version of each sector of Fig. 4-4 is provided.

A total of 25 separate Monte Carlo runs are performed. For each sector, the average RMSE of these 25 runs is included in Table B-1 below. The runs have been performed with different initial conditions and varying tracks in east direction. A visual representation of Table B-1 is provided by Fig. 4-5.

Sector	I	II	III	IV	V	VI	VII	Mean
CKF	0.77	3.54	5.31	8.37	7.57	11.40	10.06	6.42
PF	1.54	3.74	2.39	4.89	9.00	2.90	1.76	3.20
CKFAPF	0.68	2.88	3.99	8.24	7.76	9.28	8.25	5.49
IBAE-UKF	0.78	2.14	4.53	21.44	31.72	18.28	15.19	10.93
PACKF	0.61	2.44	1.43	3.93	7.34	1.97	1.21	2.26

Table 4-4: Average RMSE values of the GNSS/INS integration process for 25 Monte Carlo runs. The proposed algorithm is the overall best performer. The lowest average RMSE values are, for each sector, highlighted in green.

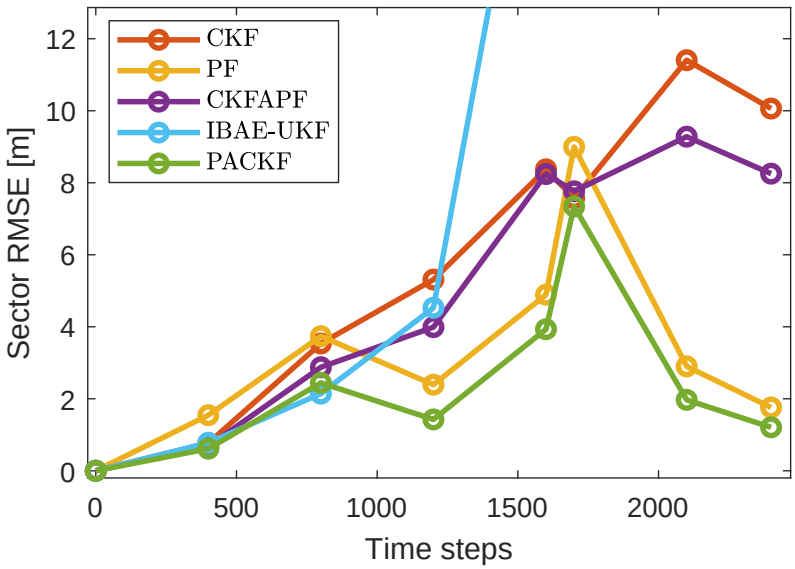

Figure 4-5: The average RMSE of 25 Monte Carlo runs at the end of each sector. The proposed filter has the lowest errors in all but one sector.

Fig. 4-5 shows that the GNSS outages, indicated by the RMSE peak at the 1700^{th} time step, are still problematic for the proposed hybrid filtering method. In the recommendations of Section 5-2, we will elaborate on potential solutions for this.

Furthermore, Fig. 4-5 shows that the standard particle filter performs adequate in the non-Gaussian domain. Under Gaussian measurement noise, however, it performs much worse than the other algorithms. Since it is expected that the majority of the measurement noise will be (close to) Gaussian, particle filters would not be that competitive under normal circumstances. Moreover, the innovation-based adaptive estimation unscented Kalman filter experiences problems with most non-Gaussian noise types. It adjusts its setting for $\mathbf{R}$ to reduce the estimation errors, but the innovation does not become much smaller after a while, due to the non-Gaussian nature that can not be captured. It seems as if the standard algorithm suffers from this feature, rather than that it is helping.

4-3-2 Validating the key assumption

Remember that it is assumed that the innovation sequences provide a good approximation of the measurement noise. To validate this, both the measurement noise $\mathbf{v}$ and the innovation sequence $\tilde{\mathbf{y}}$ for the x-position are given in Fig. 4-6. This figure validates our assumption that the innovation forms an adequate approximation of the measurement noise.

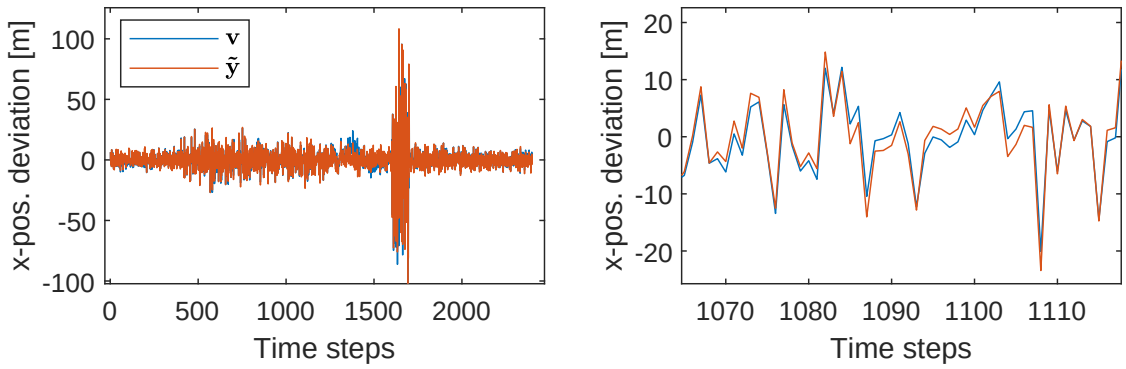

Figure 4-6: Left: comparison of GNSS measurement noise $\mathbf{v}$ and innovation $\tilde{\mathbf{y}}$ for the x-position. Right: a zoomed version of the left graph. Clearly, the innovation forms a good approximation of the measurement noise.

4-3-3 Evaluation of computation time

Logically, the hybrid architecture with all its additional features will not be as fast as the raw, standard filtering algorithms. Therefore, it is interesting to know whether the proposed method could even function in a real-time scenario, or if it increases the computation time by, for example, a factor 1000. If the latter is the case, the proposed method would be much less relevant, or even worthless.

Since a wide variety of different software, hardware and coding configurations can be used, a proper comparison of optimal computation times is difficult to establish. To be able to give an indication of computation times, all filtering algorithms have been programmed similarly. In other words, similar techniques are used and many lines of code are re-used. The computation times for each of the elements of the code structure are included in the left bar chart of Fig. 4-7. The right bar chart of Fig. 4-7 shows the total computation times of some of the

aforementioned standard algorithms. Note that this gives us a rough idea of the computational burden of the proposed method with respect to other methods, rather than a scientifically justified ranking.

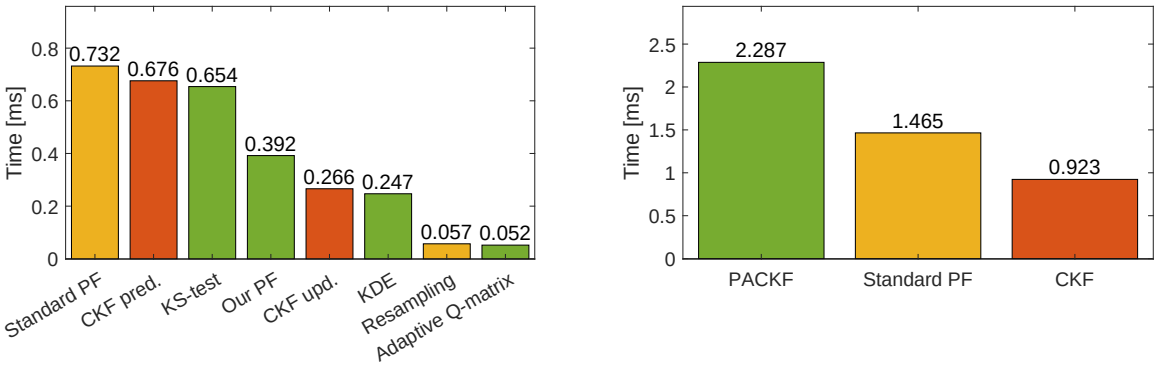

Figure 4-7: Left: computation times of different processes of the filtering algorithms. Right: total computation time of a cubature Kalman filter (CKF), particle filter (PF) and particle-aided cubature Kalman filter (PACKF) for a single time step. Note that the cubature Kalman prediction and update are also part of the particle-aided cubature Kalman filter process.

When compromising between computation complexity and estimation accuracy, the kernel density estimation algorithm might be one of the most promising novelties. This lightweight method produces the observation likelihood as a discrete 4-dimensional space (x,y-position and longitudinal- and lateral velocity), instead of building a complete 4-dimensional probability distribution. Weighting particles then becomes easy; one has to find the probability of the bins with the shortest distance to the particle for each of the measurable states. The product of these probabilities then forms the particle weight. This is much faster than weighting particles with continuous distribution, which is the general standard. For the wins in terms of accuracy that we gain with this method, the computational efforts are minimal.

4-4 Summary

In this chapter, the implementation process of the proposed particle-aided cubature Kalman filter is specified. This is required to let the algorithm perform optimal for our use-case: loosely coupled GNSS/INS integration. An additional feature for periods of GNSS outage is implemented, as well as a somewhat unusual resampling strategy; all particles are resampled at every time step. Because of this step, the algorithm becomes highly efficient in computational sense. When reviewing the localization errors, however, the proposed method shows a somewhat noisy filter estimate. The algorithm also shows to be sensitive to periods of GNSS outage. Nevertheless, the proposed method shows to beat other competitive filtering methods when considering overall performance in terms of RMSE. The algorithm is a factor 2.5 slower than the fastest standard algorithm (the cubature Kalman filter). It is expected that real-time implementation is therefore possible.

Chapter 5

Conclusion and recommendation

5-1 Concluding remarks

Accurate vehicle localization is considered to be a key element of future automated driving systems. A network of multiple sensors is employed to deliver information for this localization process. Integration of a global navigation satellite system (GNSS) and inertial navigation system (INS) is a common example of a fusion algorithm for positioning. Especially the loosely coupled integration architecture provides an elegant, intuitive solution to the GNSS/INS fusion problem. This architecture makes use of nonlinear variants of the Kalman filter family, such as the extended Kalman filter, unscented Kalman filter or cubature Kalman filter.

One of the problems of this approach, is that the process- and measurement noise covariance matrices are often not available. The GNSS measurement noise uncertainties, in particular, are highly dynamic and, depending on the specific environment, might follow a non-Gaussian distribution. We have seen that the problem of time-varying measurement noise characteristics can be solved through various estimation methods. The occurrence of non-Gaussian GNSS noise, however, still seems to be unsolved. At the hand of an example, it is demonstrated that particle filters are superior in a non-Gaussian environment. This forms the basis for a proposition for a novel hybrid filtering architecture.

The proposed adaptive particle-aided cubature Kalman filter exploits 2 features that are possible because of the hybrid structure: i) the particle filter uses an accurate proposal distribution that is provided by the cubature Kalman filter, and ii) a soft switching between the particle filter estimates and cubature Kalman filter estimates is realized. Other hybrid filtering architectures use only one of these features.

The switching algorithm compromises between the filter outputs by analyzing a measure for the Gaussianity of the GNSS noise. Because of its simplicity and effectiveness, a Kolmogorov-Smirnov algorithm is used to determine this measure. Besides, a lightweight kernel density estimation algorithm is used to approximate the current non-Gaussian distributions of the GNSS noise at any given time. These algorithms depend on knowledge of the measurement noise statistics, which is available through the assumption that the innovation sequence is a

good approximation of the GNSS measurement noise. Individually, both the hybrid switching architecture and the kernel density estimation show positive results with respect to other common filtering methods. Moreover, proof is provided that these adjustments to the standard algorithm do not tamper with the filtering convergence.

After generating ground-truth data and injecting common non-Gaussian noise sequences in the measurement data, the proposed particle-aided cubature Kalman filter is implemented in the loosely coupled GNSS/INS-integration process. An extensive tuning process is performed, with the specific choice for the proposal distribution as the biggest challenge. The results in terms of localization errors are reviewed, and the proposed method shows to be very competitive to other methods. The key assumption on the measurement noise approximation seems to be valid. It is suspected that our choice for the proposal distribution and resampling strategy induces a somewhat noisy filter output. The average root-mean-square error (RMSE) results of 25 separate Monte Carlo runs, however, indicate that the proposed adaptive particle-aided cubature Kalman filter is the overall best performer. Also in terms of computation times, the proposed method shows to be competitive.

5-2 Recommendations

There are several topics that lend itself for further research. Some smaller subjects could be investigated first. A deeper, more elaborate analysis of the effects of different choices for the proposal distribution and resampling strategy could be established. Perhaps, the noisy filter output could then be smoothened. Also, with some extra tuning efforts, there might be some small wins that can be achieved.

One could also improve the algorithm and its validation with some more high-level adjustments. A more sophisticated GNSS/INS integration strategy could be implemented, such as the ultra-tightly coupled integration architecture. Such an extension, however, makes more sense when applied to realistic data. Thus, real GNSS and INS datasets can be exploited to perform a more accurate validation step. Taking it one step further, the algorithm could be tested on an actual test setup. Then, besides an even better validation process, one could also investigate whether the algorithm is suitable for real-time applications.

In our opinion, there are two topics for further research that have more potential. At first, recent works in literature focus on accurate localization during periods of GNSS outage, which could occur in, for example, tunnel environments. Subsection 5-2-1 will concentrate on this topic. As a second, one could explore the applicability of the proposed filter in other domains. The filtering method could theoretically be applied on any system. It would, however, only function adequate in cases when i) the process noise is much smaller than the measurement noise, and ii) there is some form of bias that make the predictions of the process model inaccurate over longer periods. These are rather restrictive circumstances. Subsection 5-2-2 will elaborate on this.

5-2-1 Mitigating effects of GNSS outages

The learning approaches as highlighted in 2-4-3 show promising results. Since the proposed algorithm experiences relatively large estimation errors in periods of GNSS outage, such an

addition to the algorithm could make it more robust. In the method of [79], for example, a random forest algorithm is trained to convert specific force data from an INS to accurate positioning data. Then, when the GNSS outage occurs, the INS position- and velocity errors are predicted by the random forest-based dual model. This method does not seem to be too difficult to implement, and might improve the robustness of our algorithm.

5-2-2 Application in other domains

Localization and target-tracking are the most straight-forward applications for the proposed filter. In practice, it is difficult to find some clear examples of systems that experience a relatively small amount of process noise compared to the measurement noise. However, in theory, every system could be modelled extensively to make a process model accurate. There are endless applications possible, and we will provide a few.

In [72], an example of a double tank process is provided. The system consists of two tanks, a large reservoir and a pump. The level in the tanks has to be estimated. The Bernoulli energy equations form the process model, and noisy tank level measurements can be obtained. Imagine the two tank system would operate for a few days. It would then be very unlikely that the Bernoulli energy equations, which are logically simplified with respect to the real world process, provide an accurate tank level estimate after a few days. The noisy tank level measurements would, after a few days, be a better estimate. Thinking of the nature of this elementary example triggers some ideas for applications in other domains.

One could, for instance, think of application in monitoring bioprocesses. In such processes, concentrations of several liquids often should be monitored. Some advanced process model equations can be used to predict these concentrations, but fermentation processes are rather unpredictable in the long term. One could take samples and measure the concentration within this sample, but this is known to be a rather inaccurate method. An example for a penicillin production fed-batch process is given in [67].

In power systems, one could think of voltage- or current estimation in electrical circuits, or state-of-charge estimation of batteries. In such circuits, process- and measurement noise are often assumed to have similar proportions [68]. However, when modelled accurately, measurement noise could be dominant, such as in the example of [78].

When proper knowledge of measurement noise statistics is available, the proposed method can be applied to practically any domain in which Kalman-based filtering methods fit in nicely. This implicates, for example, orbit determination, macroeconomics, sensorless AC motor control, speech enhancement, weather forecasting, structural health monitoring, tracking of objects in computer vision and so on.

Appendix A

Full discrete state-space model

The discrete state-space model for loosely coupled GNSS/INS integration is given below:

$$\mathbf{x}_{\mathrm{k}} = \begin{bmatrix} \mathrm{I}_3 + \mathbf{F}_{11}\Delta t & \mathbf{F}_{12}\Delta t & 0_3 & 0_3 & 0_3 \\ \mathbf{F}_{21}\Delta t & \mathrm{I}_3 + \mathbf{F}_{22}\Delta t & \mathbf{F}_{23}\Delta t & \mathbf{C}^{\mathrm{n}}_{\mathrm{b},k}\Delta t & 0_3 \\ \mathbf{F}_{31}\Delta t & \mathbf{F}_{32}\Delta t & \mathrm{I}_3 + \mathbf{F}_{33}\Delta t & 0_3 & \mathbf{C}^{\mathrm{n}}_{\mathrm{b},k}\Delta t \\ 0_3 & 0_3 & 0_3 & \mathrm{I}_3 - \frac{1}{\boldsymbol{\tau}_a}\Delta t & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & \mathrm{I}_3 - \frac{1}{\boldsymbol{\tau}_\omega}\Delta t \end{bmatrix}_{k-1} \mathbf{x}_{k-1}$$
$$+ \mathbf{w}_k, \quad \mathbf{w}_k \sim N(0, \mathbf{Q}), \quad \mathbf{Q} = \begin{bmatrix} 0_3 & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & \mathbf{C}^{\mathrm{n}}_{\mathrm{b},k} & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & \mathbf{C}^{\mathrm{n}}_{\mathrm{b},k} & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & \mathrm{I}_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & \mathrm{I}_3 \end{bmatrix} \begin{bmatrix} 0 \\ \boldsymbol{\sigma}^2_{\mathrm{a}} \\ \boldsymbol{\sigma}^2_{\omega} \\ \boldsymbol{\sigma}^2_{\mathrm{b,a}} \\ \boldsymbol{\sigma}^2_{\mathrm{b},\omega} \end{bmatrix} \tag{A-1}$$

All symbols have been described in Subsection 2-3-1 of Chapter 2, except for the matrices $\mathbf{F}_{ij}$. These matrices are given below.

$$\mathbf{F}_{11} = \begin{bmatrix} 0 & \frac{V_{\mathrm{E}}\tan\phi}{R_{\mathrm{m}}+\mathrm{h}} & \frac{-V_{\mathrm{E}}}{R_{\mathrm{n}}+\mathrm{h}} \\ 0 & 0 & \frac{-V_{\mathrm{n}}}{R_{\mathrm{m}}+\mathrm{h}} \\ 0 & 0 & 0 \end{bmatrix} \tag{A-2}$$

$$\mathbf{F}_{12} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A-3}$$

$$\mathbf{F}_{21} = \begin{bmatrix} 0 & \frac{2\omega_{\mathrm{e}}(V_{\mathrm{U}}\sin\phi + V_{\mathrm{N}}\cos\phi)}{R_{\mathrm{m}}+\mathrm{h}} + \frac{V_{\mathrm{E}}V_{\mathrm{N}}}{(R_{\mathrm{m}}+\mathrm{h})(R_{\mathrm{n}}+\mathrm{h})\cos^2\phi} & \frac{V_{\mathrm{E}}(V_{\mathrm{U}} - V_{\mathrm{N}}\tan\phi)}{(R_{\mathrm{n}}+\mathrm{h})^2} \\ 0 & \frac{-2\omega_{\mathrm{e}}V_{\mathrm{E}}\cos\phi}{R_{\mathrm{m}}+\mathrm{h}} - \frac{V_{\mathrm{E}}^2}{(R_{\mathrm{m}}+\mathrm{h})(R_{\mathrm{n}}+\mathrm{h})\cos^2\phi} & \frac{-2\omega_{\mathrm{e}}V_{\mathrm{E}}\cos\phi}{R_{\mathrm{m}}+\mathrm{h}} - \frac{V_{\mathrm{E}}^2}{(R_{\mathrm{m}}+\mathrm{h})(R_{\mathrm{n}}+\mathrm{h})\cos^2\phi} \\ 0 & \frac{-2\omega_{\mathrm{e}}V_{\mathrm{E}}\sin\phi}{R_{\mathrm{m}}+\mathrm{h}} & \frac{-V_{\mathrm{N}}^2}{(R_{\mathrm{m}}+\mathrm{h})^2} - \frac{V_{\mathrm{E}}^2}{(R_{\mathrm{n}}+\mathrm{h})^2} + \frac{2\ \mathrm{g}}{R_{\mathrm{e}}+\mathrm{h}} \end{bmatrix} \tag{A-4}$$

$$\mathbf{F}_{22} = \begin{bmatrix} \frac{V_N \tan\phi}{R_m+h} - \frac{V_U}{R_n+h} & 2\omega_e \sin\phi + \frac{V_e \tan\phi}{R_n+h} & -2\omega_e \cos\phi - \frac{V_e}{R_n+h} \\ -2\omega_e \sin\phi - \frac{2V_e \tan\phi}{R_n+h} & \frac{-V_U}{R_m+h} & \frac{-V_N}{R_m+h} \\ 2\omega_e \cos\phi + \frac{2V_e}{R_n+h} & \frac{V_N}{R_m+h} & 0 \end{bmatrix} \tag{A-5}$$

$$\mathbf{F}_{23} = \begin{bmatrix} 0 & f_U & -f_N \\ -f_U & 0 & f_E \\ f_N & -f_E & 0 \end{bmatrix} \tag{A-6}$$

$$\mathbf{F}_{31} = \begin{bmatrix} 0 & 0 & \frac{-V_N}{(R_m+h)^2} \\ 0 & \frac{\omega_e \sin\phi}{R_m+h} & \frac{V_E}{(R_n+h)^2} \\ 0 & \frac{-\omega_e \cos\phi}{R_m+h} - \frac{V_E}{(R_m+h)(R_n+h)\cos^2\phi} & \frac{V_E \tan\phi}{(R_n+h)^2} \end{bmatrix} \tag{A-7}$$

$$\mathbf{F}_{32} = \begin{bmatrix} 0 & \frac{1}{(R_m+h)} & 0 \\ \frac{-1}{R_n+h} & 0 & 0 \\ \frac{-\tan\phi}{R_m+h} & 0 & 0 \end{bmatrix} \tag{A-8}$$

$$\mathbf{F}_{33} = \begin{bmatrix} 0 & \omega_e \sin\phi + \frac{V_E \tan\phi}{R_n+h} & -\omega_e \cos\phi - \frac{V_E}{R_n+h} \\ -\omega_e \sin\phi - \frac{V_E \tan\phi}{R_n+h} & 0 & \frac{-V_n}{R_m+h} \\ \omega_e \cos\phi + \frac{V_E}{R_n+h} & \frac{V_n}{R_m+h} & 0 \end{bmatrix} \tag{A-9}$$

Appendix B

Enlarged error plots

B-1 Multivariate nonlinear model

The model of (3-2) is simulated with similar settings as were used for Fig. 3-10. The estimation errors for the first state are included in Fig. B-1 (top left), as well as an enlarged area for each of the sectors (I-V) of Table 3-4.

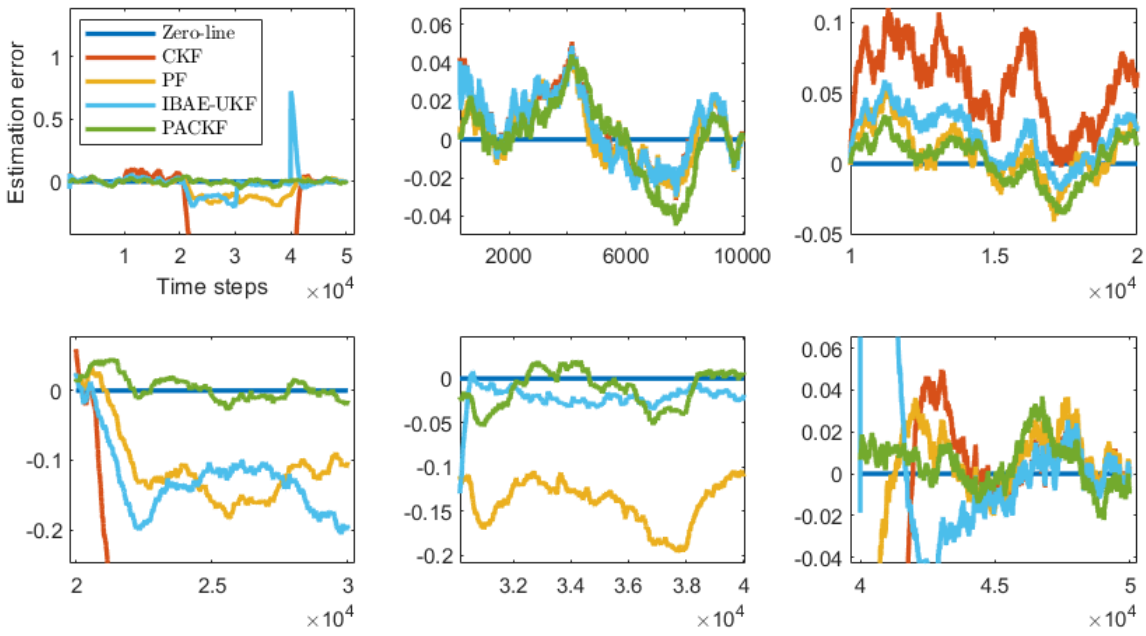

Figure B-1

For each of the sectors, the average RMSE of 25 separate Monte Carlo runs is included in Table B-1 below.

Sector	I	II	III	IV	V	Mean
CKF	0.0202	0.0624	1.4907	1.3894	0.3829	0.9277
PF	0.0264	0.0200	0.1220	0.1419	0.0251	0.0852
IBAE-UKF	0.0204	0.0269	0.1364	0.0321	0.1732	0.1007
PACKF	0.0195	0.0159	0.0179	0.0233	0.0121	0.0181

Table B-1: Average RMSE for each of the 5 sectors. Clearly, the proposed method performs best in each of the sectors.

B-2 Full loosely-coupled GNSS/INS-integration

This section contains enlarged versions of the plots of Fig. 4-4 for each sector.

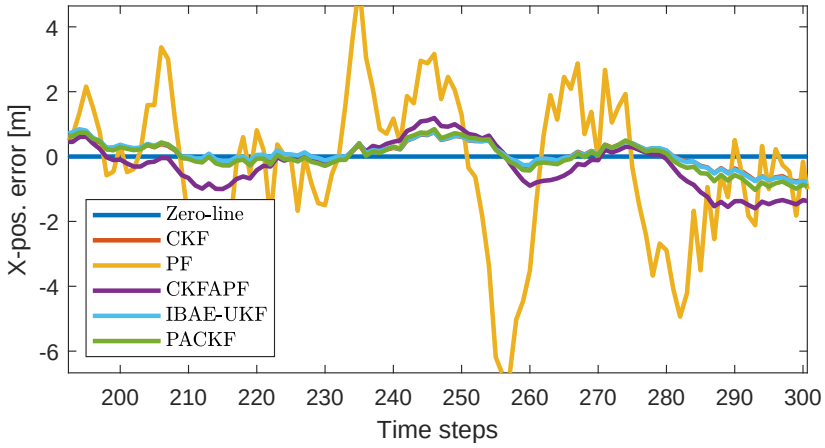

Figure B-2

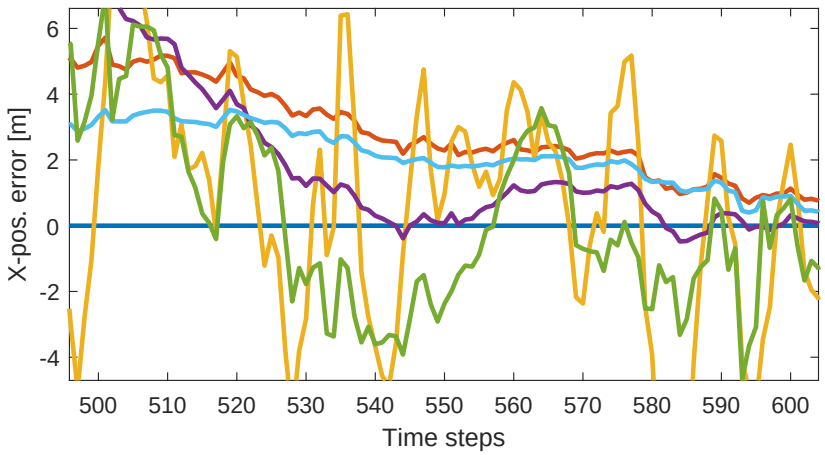

Figure B-3

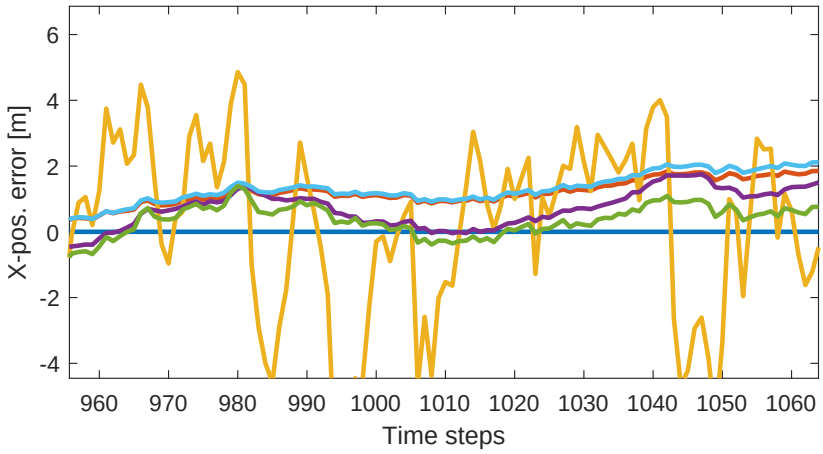

Figure B-4

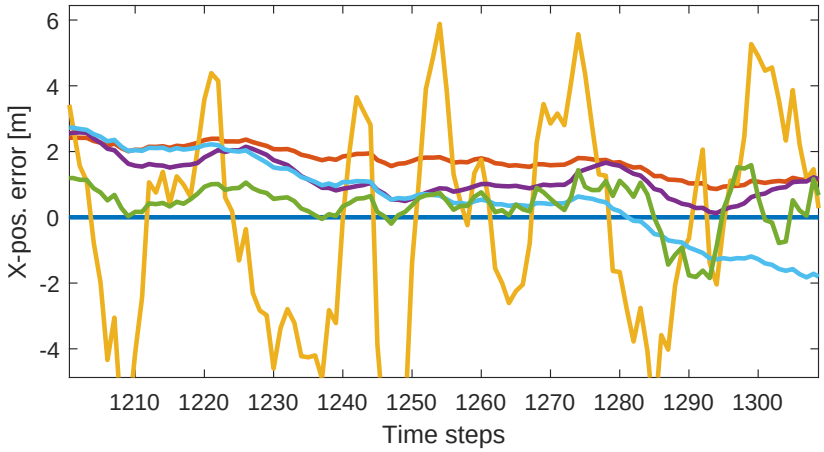

Figure B-5

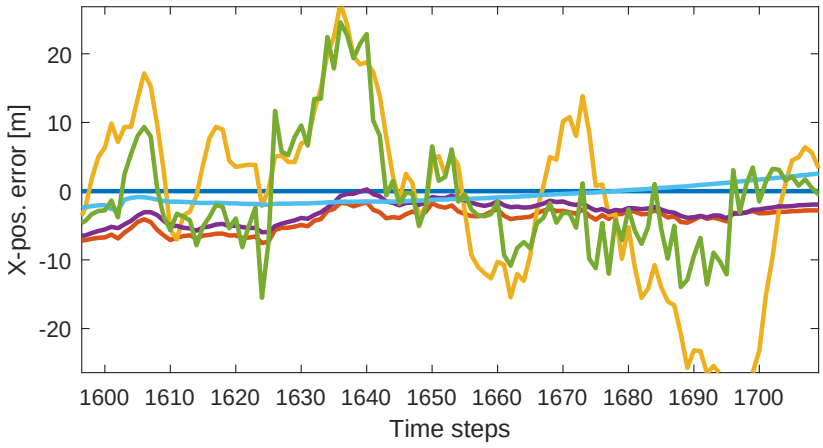

Figure B-6

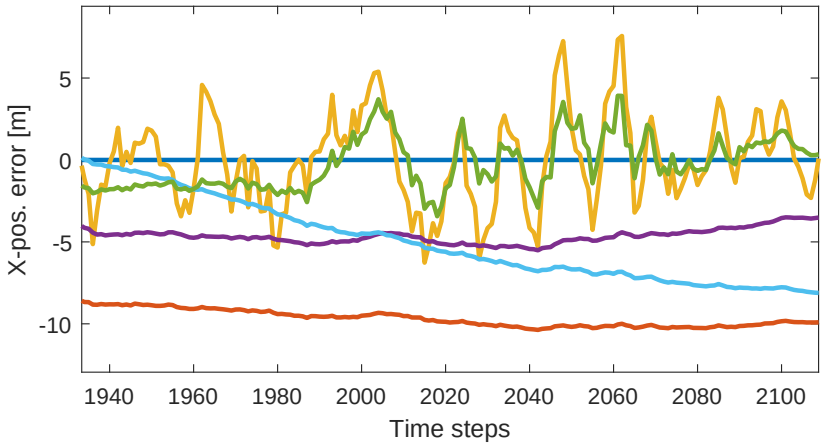

Figure B-7

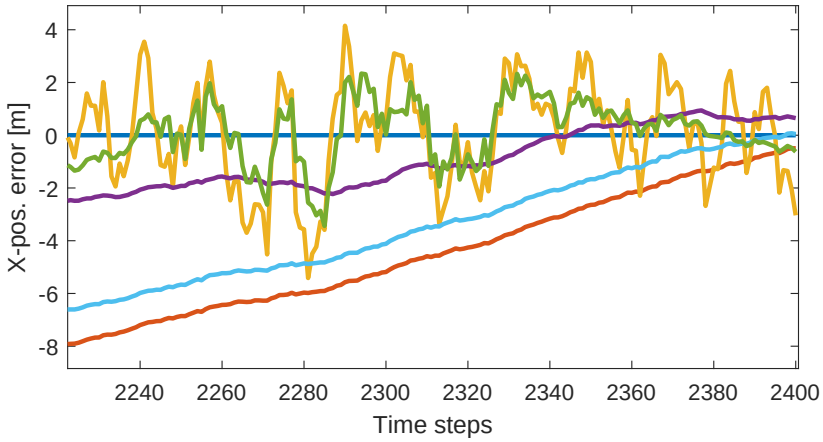

Figure B-8

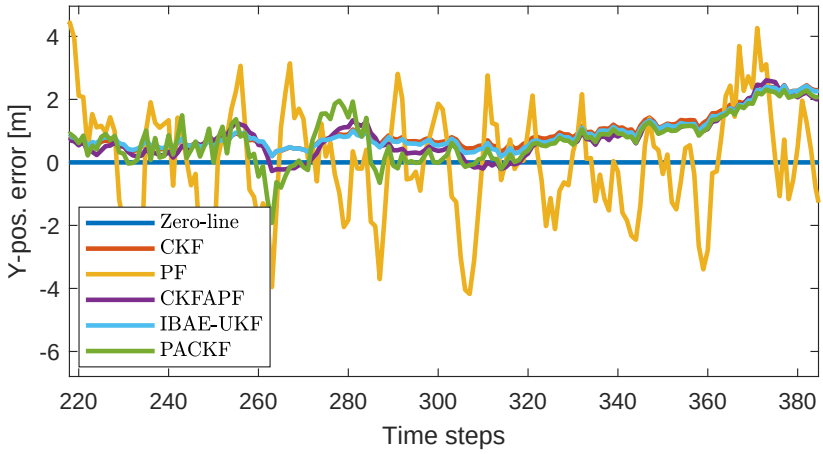

Figure B-9

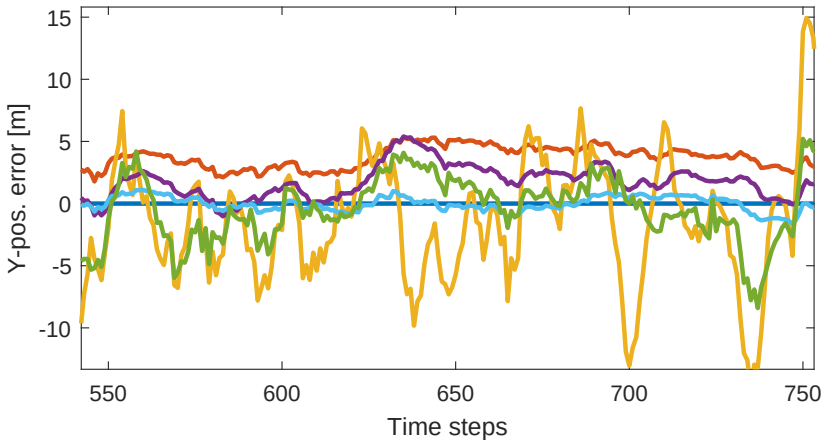

Figure B-10

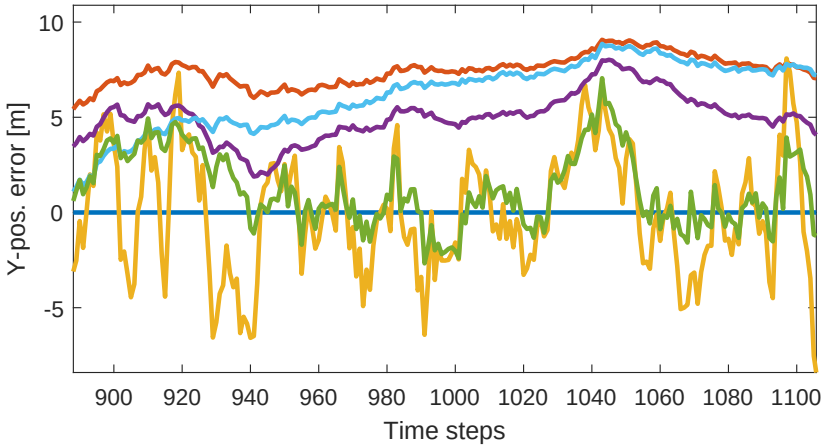

Figure B-11

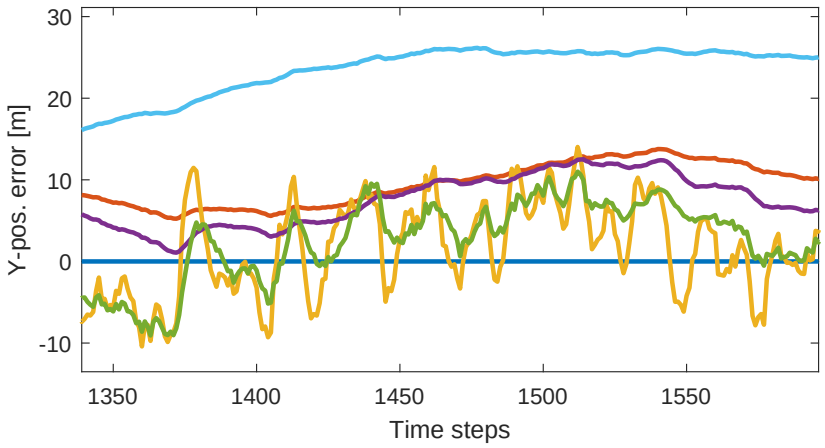

Figure B-12

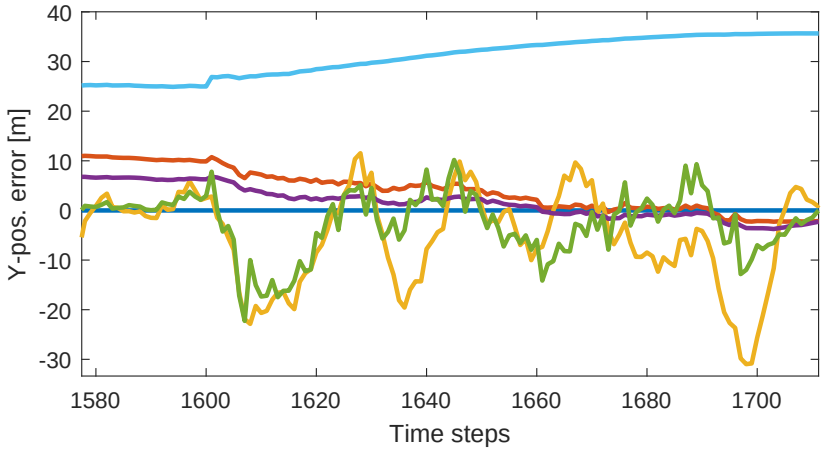

Figure B-13

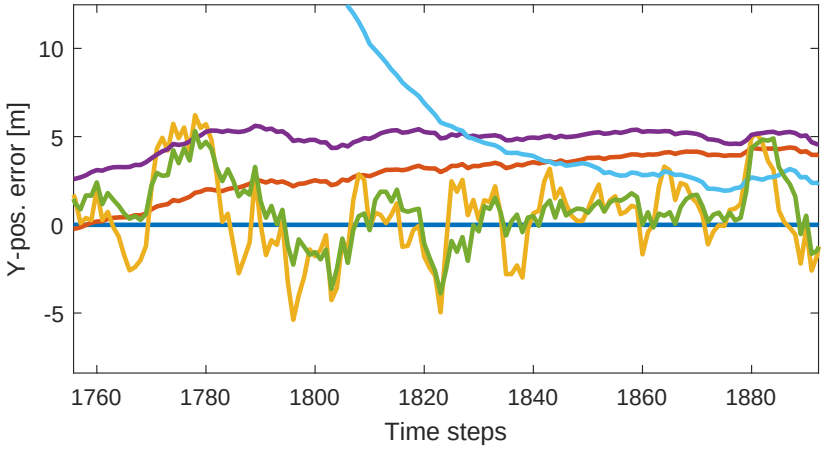

Figure B-14

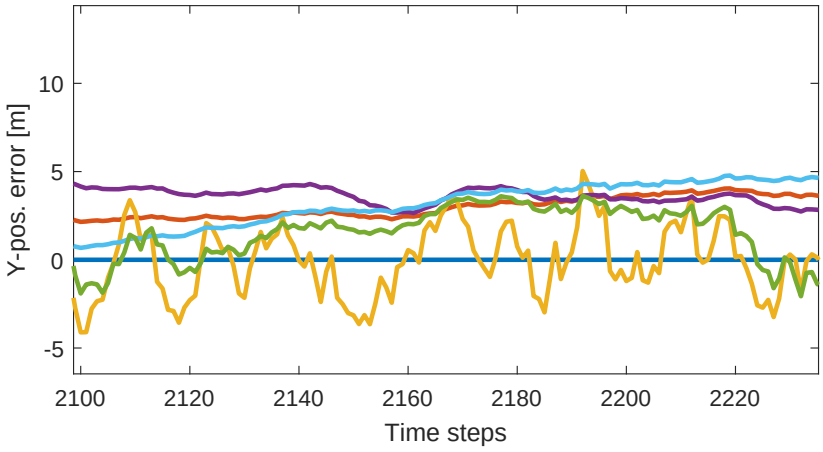

Figure B-15

Appendix C

Link to code samples

All MALTAB R2021a files of the simulations can be found on a GitHub page. The link for this page is:

https://github.com/dirkdenboer97/Thesis-files-dirk

Appendix D

Research paper

Adaptive Particle-Aided Cubature Kalman Filter for GNSS/INS-based Vehicle Localization

Dirk den Boer

Abstract - **Accurate vehicle localization is considered to be a key element of future automated driving systems. A network of multiple sensors is employed to deliver information for this localization process. Loosely coupled integration of global navigation satellite systems (GNSS) and inertial navigation systems (INS) data is a text-book example of a fusion algorithm for positioning. One of the problems of this approach, is that exact knowledge of the process- and measurement noise covariance matrices is often not available. The GNSS measurement noise uncertainties, in particular, are highly dynamic and, depending on the specific environment, might follow a non-Gaussian distribution. Since particle filters are known to be superior in non-Gaussian environments, a hybrid filtering variant is proposed: adaptive particle-aided cubature Kalman filtering. This algorithm compromises between a particle filter with kernel density estimation algorithm in periods of non-Gaussian GNSS noise, and a standard cubature Kalman filter in case of Gaussian GNSS noise. The results of GNSS/INS-based localization simulations indicate that the proposed adaptive particle-aided cubature Kalman filter outperforms traditional filtering methods in terms of minimal localization errors.**

Index terms - **GNSS/INS-based vehicle localization, particle filter (PF), cubature Kalman filter (CKF), hybrid archtitecture, kernel density estimation, switching strategy**

I. Introduction

One element keeps coming back when advancing vehicle technology; the need for accurate vehicle localization [1]. For higher levels of autonomy, a vehicle must be able to know its location up to decimeter level [2]. Almost all driver assistance systems depend on this localization.

Ideally, all incoming information for automated vehicles is reliable and robust. In reality, complex and degraded working conditions could drastically degrade the reliability and robustness of sensor information. Though some types of information sources most definitely outperform others, there is a broad consent on the fact that any stand-alone hardware system is not fit for vehicle localization on its own [3]. Current localization techniques therefore exploit multiple sensors that complement each other [4]. Integration of GNSS and INS measurements is an example of such information fusion. These two sensors are almost exclusively present in any vehicle localization architecture.

Dirk den Boer is a master student Systems & Control at TU Delft, The Netherlands. E-mail addresses: dgadenboer@gmail.com.

A global navigation satellite system (GNSS) provides geospatial positioning on a global scale. The great advantage of GNSS is its global coverage. Unfortunately, GNSS signals might be lost in areas where the direct line to satellites is obstructed; think of tunnels, valleys, forests and urban areas. Besides, for localization purposes, low-end models are often employed, which have an accuracy of approximately 5m for the cheapest models [5], or 2.5m for slightly better models [6]. This accuracy does not meet the requirements for higher levels of autonomy in vehicles.

An inertial navigation system (INS) can determine the position of a vehicle at the hand of a set of motion sensors, rotation sensors, wheel encoders and a computer. The data comes from an inertial measurement unit (IMU), which often consists of an accelerometer, gyroscope and possibly a magnetometer. Since the motion sensors capture the relative motion with respect to the previous measurement, a so-called dead-reckoning method is used for determining a location. Dead-reckoning uses speed and heading estimates as well as time measurements to determine a current position.

Data from GNSS and INS is considered to be a golden combination for sensor fusion, since the properties of these data sources are opposing in some sense. INS location estimates encounter problems of accumulating errors over longer distances, thereby failing in long term navigation operations [7]. The integration of GNSS can correct this error accumulation in intervals, thereby limiting the localization errors. Besides, the smooth INS location estimates are used instead of the noisy GNSS signal. It results in both smooth and accurate location estimates.

Recursive Bayesian filtering algorithms form a common method for fusion of GNSS and INS data. The celebrated Kalman filter [8] is often employed. A Kalman filter, as well as its nonlinear variants such as the extended Kalman filter [9], the unscented Kalman filter ([10], [11]) and the cubature Kalman filter [12], require the process- and measurement noise covariance matrix ($\mathbf{Q}$ and $\mathbf{R}$, respectively) to be known. The ratio between these two matrices determines the proportions in which the predictions and measurements contribute to the final state estimate. The traditional Kalman filter assumes *a priori* defined, constant values for these matrices. In reality, however, these matrices might be time-varying due to changing conditions [13]. Therefore, one often attempts to accurately adjust the process- and measurement noise covariance matrices to changing conditions.

Unfortunately, filters of the Kalman family are still bound to a Gaussian process- and measurement noise representation. This might harm the localization accuracy in cases of, for example, non-Gaussian measurement noise. In other user-

domains, such problems are solved by exploiting a hybrid filtering architecture that incorporates a particle filter. This article features a proposition for such a hybrid filtering architecture for GNSS/INS-based vehicle localization: adaptive particle-aided cubature Kalman filtering (PACKF). The contributions of this work can be summarized as follows:

1) A hybrid filtering architecture allows for soft switching between a particle filters output and a cubature Kalman filters output. This switching strategy is based on a measure for the Gaussianity of the measurement noise. The measurement noise is approximated by the innovation sequence of the cubature Kalman filter predictions. A Kolmogorov-Smirnov test is performed on the innovation sequence to determine this measure for Gaussianity.
2) An innovation-based, adaptive process noise covariance matrix is used to adjust the proposal distribution such that it support encloses that of the measurement likelihood distribution at any specific moment. This is important for guaranteeing the particle filter's convergence.
3) The innovation sequence is used in a kernel density estimation algorithm to accurately weight particles with any measurement likelihood distribution (Gaussian or non-Gaussian).

The remainder of this article is structured as follows; Section II discusses some recent works on process- and measurement noise covariance estimation and hybrid filtering architectures. An introduction to stochastic filtering problems is given in Section III, along with a demonstration of the superiority of particle filters in the non-Gaussian domain. The core elements of the proposed algorithm are explained in Section IV, after which Section V contains an experimental evaluation of the algorithm on both a simple benchmark example as a full loosely coupled GNSS/INS-integration strategy. The concluding remarks are given in Section VI.

II. Related work

A great literature has been devoted to noise covariance estimation in GNSS/INS-based vehicle localization. Some recent works on this topic are discussed in Subsection II-A. Common hybrid filtering architectures that use both a particle filter and a Kalman-type filter, are discussed in Subsection II-B.

A. Noise covariance

The work [14] comprises several adaptive Kalman filter-based methods for GNSS/INS integration, and an evaluation of their performance. A compact, very clear introduction is given for three types of covariance estimation: i) innovation-based adaptive estimation (estimating $\mathbf{R}$), ii) residual-based adaptive estimation (estimating $\mathbf{R}$) and iii) strong tracking filtering (estimating $\mathbf{Q}$). Some more advanced examples are provided in [15] (innovation-based adaptive estimation) and [16], [17] (strong tracking filtering).

In [18], the authors propose an interacting multiple model (IMM) two stage Kalman filter. A 'bias-free' filter operates without INS biases. Its output is then corrected by that of a second Kalman filter. Within this second filter, the outputs of three filters, each with different process noise covariance matrix $\mathbf{Q}$, are mixed. Likewise, [19] contains a measurement modelling method that exploits an extended state observer. The total, unpredictable uncertainty in the system is obtained by taking the difference between the measurements and extended state observer predictions. Assuming uncertainty mainly comes from measurement noise, a matrix $\mathbf{R}$ is updated based on this difference.

In an approach that bears some resemblance to interacting multiple model filtering, the authors propose a federated Kalman filter with strong tracking properties in [20]. A least-squares principle determines the mixing proportions of the outcomes of the filters. More recently, attempts have been made to estimate the process noise covariance $\mathbf{Q}$ based on the maximum likelihood (ML) principle [21], [22].

A somewhat different approach uses a multitask learning model to find matrices $\mathbf{Q}$ and $\mathbf{R}$ [23]. With the estimated orientation, velocity and INS measurements as input, the learning model is trained when proper location data (e.g. from GNSS-RTK) is available.

To the best of our knowledge, filtering non-Gaussian GNSS noise distributions has not yet been investigated intensively. In addition, employing a kernel density estimation algorithm to form the measurement likelihood distribution for a particle filter has not been attempted in any domain.

B. Hybrid architectures

Several hybrid filtering architectures have been developed that employ both a particle filter and a nonlinear variant of the Kalman filter. In the most straight forward one, two filters work in parallel, and a switching mechanism based on an arbitrary criterion determines which output is used. A distinction is made between a hard switch (either use the particle filter output or the Kalman-type filter output) and a soft switch (a compromise/weighted average of both). An example of such a hard switching algorithm can be found in [24], in which it is applied to object tracking.

Perhaps the most common hybrid filtering method is the unscented particle filter [25]. Since it is generally rather difficult to find a good proposal distribution for the particle filter with limited knowledge, an unscented Kalman filter can be used to provide this distribution. Although this method could solve the problem of non-Gaussian measurement noise, the drawbacks of this method are as follows: i) the proposal distribution is then exclusively Gaussian, ii) when accurate predictions are available, one does not want to include inaccurate observations in the proposal distribution, and iii) the particle filters sampling procedure might decrease the accuracy with respect to the unscented Kalman estimate in pure Gaussian environments.

In a recent series of journal papers, the authors propose a particle-aided unscented Kalman filter for vehicle localization [26], [27], [28]. The particle filter is now connected to the observation channel of the unscented Kalman filter, without any switching mechanism. It has the advantage that the particle filter can work with a reduced version of the state-space model (one that only calculates the measurable states). There is, however, one significant downside of this method; there

are now limited tools for constructing a proper proposal distribution and observation likelihood. The last work of the series of papers [26] solves this issue by employing a discriminative parameter training algorithm that learns an optimal measurement likelihood distribution. This looks, however, like a brute force approach that only adds unnecessary complexity to a problem that can be avoided in a simpler way. The method proposed in this work overcomes these issues of the current hybrid architectures.

III. Bayesian filtering in the non-linear non-Gaussian domain

In stochastic filtering problems, one aims to find the states of a given system at the hand of a set of noisy observations and model predictions. The stochastic nature of such problems comes from stochastic noise sequences $\mathbf{w}_k$ and $\mathbf{v}_k$. Consider the following discrete state-space model with additive process- and measurement noise sequences $\mathbf{w}_k$ and $\mathbf{v}_k$, respectively:

$$\begin{aligned} \mathbf{x}_{k+1} &= f\left(\mathbf{x}_k, \mathbf{u}_k\right) + \mathbf{w}_k \\ \mathbf{z}_k &= h\left(\mathbf{x}_k\right) + \mathbf{v}_k \end{aligned} \tag{1}$$

In this equation, subscript k is a time indicator, $\mathbf{x}$ denotes the state vector, $\mathbf{u}$ is the input vector, $\mathbf{z}$ is the observation vector and $\mathbf{w}_k$ and $\mathbf{v}_k$ are noise sequences with unknown statistics. The first equation of (1) indicates the state transition by function f. The probability that a current state takes the value $\mathbf{x}_k$, given the previous state $\mathbf{x}_{k-1}$ and input $\mathbf{u}_k$, is denoted as $p\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k\right)$. The second equation of (1) is the measurement function h, which describes the probability $p\left(\mathbf{z}_k \mid \mathbf{x}_k\right)$ that a measurement takes value $\mathbf{z}_k$ given state $\mathbf{x}_k$.

The objective of filtering is to find the optimal state, i.e. the best estimate, of $\mathbf{x}$ at time k. This comes down to, given initial probability density $p\left(\mathbf{x}_0\right)$, measurement likelihood $p\left(\mathbf{z}_k \mid \mathbf{x}_k\right)$ and prior state transition probability density $p\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k\right)$, finding the posterior density $p\left(\mathbf{x}_k \mid \mathbf{z}_k\right)$. From this posterior distribution, a state estimate $\hat{\mathbf{x}}_{k|k}$ and, when necessary, a corresponding estimate for the covariance matrix $\hat{\mathbf{P}}_{k|k}$ can be obtained.

A. Traditional methods

Two methods are very common in stochastic filtering problems: Kalman filtering [8] and particle filtering [29]. The well-known Kalman filter is the optimal estimator in the linear-Gaussian domain. These circumstances are, however, very restrictive when considering that many realistic systems are nonlinear, with possibly non-Gaussian noise sources.

The main advantages of particle filters over Kalman-type filters are the fact that they i) are suitable for complex, highly nonlinear systems, and ii) that they can cope with non-Gaussian distributions. The latter will be highlighted in the next subsection. A major drawback of particle filters their computational complexity, a factor that often depends on the number of particles that is employed. The accuracy of the state estimate depends on the chosen number of particles, as well as other settable parameters such as the proposal distribution and resampling strategy. In general, nonlinear Kalman filtering methods require less computational time and tuning efforts than particle filtering methods.

B. Superiority of the particle filter in the non-Gaussian domain

The nonlinear function of (2) is used to evaluate the nonlinear transformations of various filtering methods. This is an adjusted version of the well-known benchmark growth model that was introduced in the original paper on particle filtering [29]. For the sake of simplicity, the time-varying component is omitted.

$$y = f(x) = \frac{x}{2} + 25\frac{x}{1+x^2} \tag{2}$$

A stochastic variable x has mean $\bar{x}$ and perturbation error e such that $x = \bar{x} + e$. The goal of nonlinear transformations is to estimate the statistics of the stochastic output variable y. With statistics, we indicate the first p-order central moments of this output variable y. For now, we are particularly interested in the first 2 central moments, i.e. the mean and the covariance.

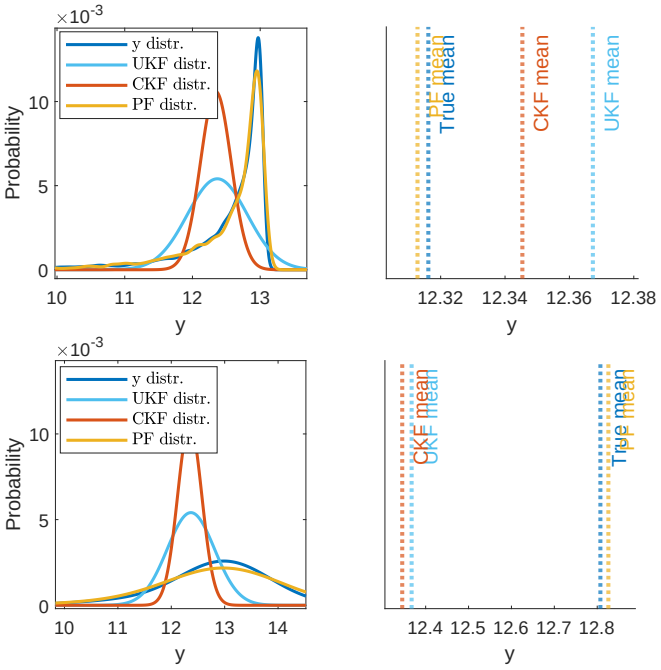

Fig. 1. Top left: for a Gaussian distributed x; true distribution of $y = f(x)$ (2) and the estimated distributions of an unscented transform (UKF, indigo), cubature transform (CKF, orange) and Monte Carlo approximation with 100 samples (PF, yellow). Bottom left: similar, but for a stochastic x that follows a Poisson distribution. Top right: enlarged on the true mean and approximated means. Bottom right: similar, but for non-Gaussian x. Clearly, the Monte Carlo approach performs best in terms of approximating the true mean.

The true distribution of y can be approximated by Monte Carlo simulations; the resulting distribution of variable y after propagating x $1 \cdot 10^5$ times through (2) is given by the blue density of Fig. 1. Two simulations are performed: one with $e_1 \sim N(0, 0.1)$ (top graphs) and one with $e_2 \sim \text{Pois}(\lambda = 0.1)$ (lower two graphs). Note that the variance of x is equal for both cases ($\sigma^2_{e_1} = \sigma^2_{e_2}$). The Poisson variable e_2 is normalized, such that x has mean 1 in both simulations. Thus, the first 2 central moments of x are exactly the same. However, the shape of the distribution of x is different. Properties of the shape of distributions (e.g. kurtosis, skewness) are harvested

in higher order central moments. The means of the probability distributions of the left graph of Fig. 1 are indicated by the dotted lines of the graphs on the right.

An unscented transformation, a third-degree cubature transformation and a Monte Carlo sampling method (with 100 particles) are applied to find the statistics of y without having to perform $1{\cdot}10^5$ function evaluations. For $e_1 \sim N(0, 0.1)$, the results of these nonlinear transformations are also included in Fig. 1. Both the unscented- and third-degree cubature transformation represent y as a Gaussian, as expected. However, the Monte Carlo sampling method captures the true shape of the density of y much better.

For a second simulation, we assume that all filters have full knowledge of the higher order statistics of e_2. The unscented- and cubature transformation allow a mean and covariance matrix as inputs for x, but these do not change. There is nothing to adjust in these algorithms. Since the Monte Carlo method can draw from any distribution, it can also draw from $e_2 \sim \text{Pois}(\lambda = 0.1)$. With these settings, the same simulations are performed again, and the resulting density estimates are given in the lower-left graph of Fig. 1. For this non-Gaussian x, the mean of y is now approximated poorly by the distributions of unscented- and cubature transformations. Again, the Monte Carlo method results in a much better estimate. Thus, with some knowledge of possibly non-Gaussian distributed disturbances, the mean of a nonlinearly transformed stochastic variable x can be approximated better by Monte Carlo methods, such as the particle filter. This is a very strong property that will be exploited in our hybrid filtering algorithm.

IV. Adaptive particle-aided cubature Kalman filtering

First, Subsection IV-A explains what key assumption forms the basis for our filtering algorithm. The remainder of this section then highlights the three main contributions of the proposed algorithm. Subsection IV-B will discuss the implementation of the soft switching strategy in the hybrid filtering architecture. The determination of the adaptive matrix $\tilde{\mathbf{Q}}$, along with its advantages, will be described in Subsection IV-C. At last, Subsection IV-D features the kernel density estimation algorithm that is used for constructing the adaptive measurement likelihood distribution of our particle filter. The full algorithm can be found in Appendix A.

A. A key assumption

Let us say that a filter is convergent, and the previous state can be estimated properly (its estimation error $\mathbf{e}_{k-1}$ is small). The predicted observation $\hat{\mathbf{z}}_k$ of a cubature Kalman filter is then an accurate approximation of the true system outputs. Kalman filters denote the difference between the predicted observation $\hat{\mathbf{z}}_k$ and the actual observation $\mathbf{z}_k$ as the innovation; $\tilde{\mathbf{y}}_k = \mathbf{z}_k - \hat{\mathbf{z}}_k$. When the proportion of the process noise over a single time step is much smaller than the proportion of the measurement noise, the measurement noise $\mathbf{v}_k$ can be approximated by this innovation signal. This assumption yields the following:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \hat{\mathbf{z}}_k = (\check{\mathbf{x}}_k + \mathbf{v}_k) - (\check{\mathbf{x}}_k + \mathbf{e}_{k-1} + \mathbf{w}_k) \approx \mathbf{v}_k \quad (3)$$

in which $\check{\mathbf{x}}_k$ is the true state, $\mathbf{w}_k$ and $\mathbf{v}_k$ are the process- and measurement errors and $\mathbf{e}_{k-1}$ is the estimation error of the previous state. To summarize, assuming that the innovation signal sufficiently accurately approximates the measurement noise, we possess over the right knowledge to let our particle filter perform better than the cubature Kalman filter. One might wonder; why not use the accurate predictions as final state estimates? In many systems (e.g. GNSS/INS-based vehicle localization) a prediction error builds up over time. That is why, after a while, the predictions become useless and there is a need for measurement corrections in our final state estimate.

B. Soft switching strategy

Different from other filtering approaches, the matrices $\tilde{\mathbf{Q}}_k$ and $\tilde{\mathbf{R}}_k$ are used as tools to manipulate the outcome of our state estimate, rather than representing the process- and measurement noise covariance matrices $\mathbf{Q}$ and $\mathbf{R}$. Both the process- and measurement noise covariance matrices can often not be determined accurately. In fact, the noise statistics are often dynamic, and inaccurate, fixed values for the noise covariance can be harmful for the estimation process. Although more configurations are possible, we exploit $\tilde{\mathbf{z}}_k$ and $\tilde{\mathbf{R}}_k$ as a switching mechanism between our filters.

The particle filter is connected to a cubature Kalman filter through the observations channel of the standard algorithm. Its help can be turned off during periods of Gaussian measurement noise, but can be increased during period of non-Gaussian measurement noise. Instead of using the standard observations vector $\mathbf{z}_k$, the modified version of (4) below is used:

$$\mathbf{z}_k \Rightarrow \tilde{\mathbf{z}}_k = (1 - \boldsymbol{\mu}_k)\, \mathbf{z}_k + \boldsymbol{\mu} \mathbf{z}_k^{\text{pf}} \quad (4)$$

This equation makes sure that for each individual state, the filter is functioning as a standard cubature Kalman filter when $\mu_k \to 0$ and uses the particle filter's result as observation when $\mu_k \to 1$. When the uncertainty of this new observation is extremely low, the filter 'trusts' the observations more than the predictions, and eventually, the observations will contribute significantly to the final state estimate. Therefore, we adjust the uncertainty of the observations with the matrix $\tilde{\mathbf{R}}_k$. Instead of using a normal, fixed measurement noise matrix $\mathbf{R}$, the following matrix is used:

$$\mathbf{R} \Rightarrow \tilde{\mathbf{R}}_k = (1 - \boldsymbol{\mu}_k)\, \mathbf{R} \quad (5)$$

This adjustment leads to a small measurement noise variance when $\mu_k \to 1$ ($\lim_{\mu_k \to 1} \tilde{\mathbf{R}}_k = 0$), but the normal measurement noise variance is used when $\mu_k \to 0$ ($\lim_{\mu_k \to 0} \tilde{\mathbf{R}}_k = \mathbf{R}$). In the former case, the influence of the cubature Kalman filter predictions are reduced, since the process noise is generally larger than $\tilde{\mathbf{R}}_k \to 0$. The cubature Kalman filter then fully uses the particle filter's estimate as the definitive one.

As stated before, the Gaussianity of the measurement noise will determine whether the final state estimate is mainly

determined by the particle filter, or the cubature Kalman filter. To determine a measure for the Gaussianity of the measurement noise, a Kolmogorov-Smirnov test [30] can be used. Let us take the last n normalized innovations (i.e. $\tilde{\mathbf{y}}_{i,\text{norm}} = \tilde{\mathbf{y}}_i/|\sigma_{\tilde{\mathbf{y}}}|$) as data points. The method first determines an empirical cumulative distribution function $F_k(x)$ for n ordered innovations $\tilde{\mathbf{y}}_i$ as follows:

$$F_k(x) = \frac{1}{n}\sum_{i=k-n}^{k} 1_{[-\infty,x]}\left(\tilde{\mathbf{y}}_{i,\text{norm}}\right) \tag{6}$$

Note that $1_{[-\infty,n]}\left(\tilde{\mathbf{y}}_{i,\text{norm}}\right)$ is the indicator function. This function equals 1 if $\tilde{\mathbf{y}}_{i,\text{norm}} \leq x$ and equals 0 otherwise. The Kolmogorov-Smirnov statistic D_k is then defined as the supremum of the set of distances between a reference cumulative distribution $F(x)$ (in our case a standard normal distribution) and the empirical innovation cumulative distribution $F_k(x)$. The statistics D_k can be calculated as follows:

$$D_k = \sup_x |F_k(x) - F(x)| \tag{7}$$

A graphical representation can be found in Fig. 2.

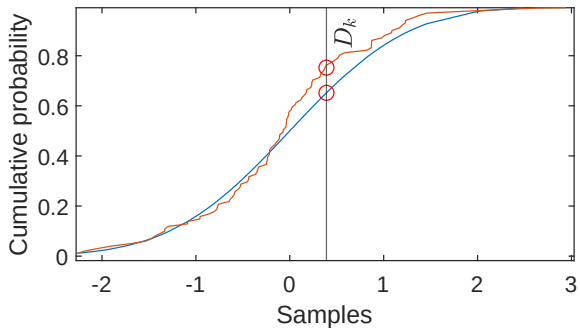

Fig. 2. Cumulative distribution function of a standard normal distribution (blue), and the empirical cumulative distribution function of the last n normalized innovations $\tilde{\mathbf{y}}_{i,\text{norm}}$ (red). The Kolmogorov-Smirnov statistic D_k is supremum of the set of distances between the two distributions, as indicated by the red markers.

The null hypothesis that the last n normalized innovations come from a standard normal distribution can then be rejected based on $\sqrt{n}D_k > K_\alpha, \quad K = \sup_{t\in[0,1]}|B(t)|$ with $1-\alpha$ confidence, in which $B(t)$ is a Brownian bridge [31]. When rejected, $\breve{\mu}_k$ takes value $\breve{\mu}_k = 1$. To avoid fast switching between filter settings, the definitive Gaussianity indicator vector $\boldsymbol{\mu}_k$ is taken as:

$$\boldsymbol{\mu}_k = \min\left(\frac{1}{n}\sum_{i=k-n}^{k}\breve{\boldsymbol{\mu}}_i, \quad 1-\gamma\right), \qquad 0 < \gamma \ll 1 \tag{8}$$

Stability issues might arise when $\mu_k = 1$, since then $\tilde{\mathbf{R}}_k = 0$. Therefore, a safety factor $\gamma \ll 1$ is employed to avoid this from happening.

C. Adaptive matrix $\tilde{\mathbf{Q}}$

Following [32], we use $\tilde{\mathbf{Q}}_k$ as an adaptive measure for the unknown components of our system. The instrumental matrix $\tilde{\mathbf{Q}}_k$ is calculated as follows:

$$\tilde{\mathbf{Q}}_k = \rho\tilde{\mathbf{y}}_k^T\tilde{\mathbf{y}}_k\mathbf{I}_n + \eta\mathbf{I}_n \tag{9}$$

with tuning parameters $\rho > 0$ chosen large enough and $\eta > 0$ chosen small enough. This particular choice for the instrumental $\tilde{\mathbf{Q}}_k$ ensures that in case of large estimation errors, the conditions for cubature Kalman filter convergence are valid at all times [33]. Besides, when using this matrix for the proposal distribution of the particle filter, the support of the proposal distribution grows with the innovation. As a consequence, the support of the proposal distribution will always enclose, or approach at the least, the support of the measurement likelihood distribution. This is a key requirement for particle filter convergence, as can be seen in Fig. 3.

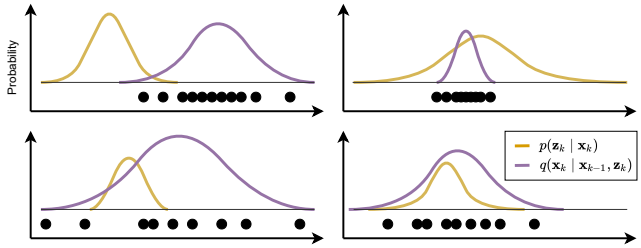

Fig. 3. Four configurations for a proposal distribution and measurement likelihood. Top left: the proposal distribution has a small support that does not cover the measurement likelihood. Only two particles get assigned a weight, and it is highly unlikely that this weighted average approaches the measurement likelihood mean. Top right: the proposal distribution is too narrow. All particles get assigned a weight of similar order, and the weighted particle mean will not be a good estimate of the observations mean. Bottom left: the proposal distribution has a wide support, but only a few particles are weighted. This leads to inaccurate estimation. Bottom right: the proposal distribution is similar in shape and support as the measurement likelihood. The weighted average of the particles will be a good approximation of the measurement likelihood mean.

We have access to the fairly accurate predictions of the cubature Kalman filter before drawing particles for our particle filter. It is very likely that these predictions fall close to the true posterior. Thus, sampling around the cubature Kalman prediction seems an obvious choice. The proposal distribution then becomes as follows:

$$q(\mathbf{x}_k \mid \mathbf{x}_{k-1}) = N\left(\hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1}\right) \tag{10}$$

The predicted covariance matrix $\mathbf{P}_{k|k-1}$ partially depends on the instrumental matrix $\tilde{\mathbf{Q}}_k$, which depends on the innovation $\tilde{\mathbf{y}}$. In other phases of the proposed hybrid filter, a set of standard, fixed $\mathbf{Q}$ and $\mathbf{R}$ are used, as can be seen in the full algorithm description of Appendix A.

D. Kernel density estimation algorithm

The particle weighting step requires the determination of the observation likelihood $p\left(\mathbf{z}_k \mid \mathbf{x}_k\right)$. In practice, gaining true statistics of the actual observation likelihood is difficult. Often, this likelihood is assumed to be constant and Gaussian. To overcome the problem of inaccurate particle weighting due to incorrect measurement modelling, an adaptive particle weighting scheme is proposed. The last n innovations $\tilde{\mathbf{y}}_i$, which are essentially the measurement noise values due to

our assumption (3), are used to approximate the true observation likelihood. The calculation of (11) below shows how this probability distribution function estimate $\hat{f}_{\mathrm{KDE}}(x)$ can be obtained for a single state:

$$\hat{f}_{\mathrm{KDE}}(x) = \frac{1}{n}\sum_{i=1}^{n} K_h\left(x - \tilde{y}_i\right) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-\tilde{y}_i}{h}\right) \quad (11)$$

The function K indicates an arbitrary kernel, $\tilde{y}_i$ represents one of the last n samples, and the parameter h denotes the bandwidth. The latter is a free parameter that determines the smoothness of the resulting distribution function. A common choice for this bandwidth is the one that minimizes the mean integrated squared error:

$$h = \left(\frac{4\hat{\sigma}^5}{3n}\right)^{\frac{1}{5}} \approx 1.06\,\hat{\sigma}\,n^{-\frac{1}{5}} \quad (12)$$

Such an approach for estimating densities at the hand of data points, is called kernel density estimation [34], [35]. For the kernel function $K\left((x-\mathbf{y}_i)/h\right) = K(u)$, an Epanechnikov kernel [36] is used. This is a parabolic kernel function that can be calculated as follows:

$$K(u) = \frac{3}{4}\left(1-u^2\right), \text{for} \quad |u| \leq 1 \quad (13)$$

Each particle will be weighted according to the approximated measurement likelihood, provided by the kernel density estimates $\hat{f}_{\mathrm{KDE}}(x)$ of each measurable state. With this approach, particles can be weighted appropriately in case of time-varying and non-Gaussian measurement noise. Note that this approach is possible only because we have sufficiently accurate measurement noise approximations. A demonstration of the kernel density estimation algorithm is provided by Fig. 4.

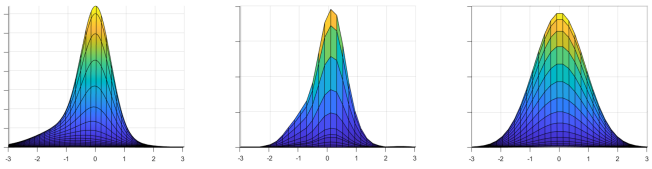

Fig. 4. A two-dimensional Gaussian mixture distribution (left) represents the true probability distribution of the measurement noise of system (2). This distribution has a second mode on the left side of the origin and has a relatively high skewness (i.e. asymmetry) and kurtosis (i.e. heavy tails). If the measurement noise were to be approximated by a two-dimensional Gaussian, one obtains the distribution in the right-most plot. If we take $n = 50$ innovations, and use this data to approximate the true distribution with our kernel density estimation algorithm, the center distribution is obtained. The proposed algorithm manages to capture the shape and tailed behavior better than the Gaussian approximation.

V. Experimental Evaluation and Analysis

An evaluation of the performance of the proposed method is provided in this section. At first, the individual highlights (adaptive hybrid switching architecture and kernel density estimation algorithm) of the algorithm will be evaluated one-by-one through a simple benchmark example in Subsection V-A. Then, Subsection V-B contains the description and results of a full loosely coupled GNSS/INS-integration strategy.

A. Results for a benchmark example

Consider the following multivariate nonlinear model:

$$\begin{aligned}\mathbf{x}_{k+1} &= \begin{bmatrix} x_{1,k} + \tau x_{2,k} \\ x_{2,k} + -\tau x_{1,k} + \left(x_{1,k}^2 + x_{2,k}^2 - 1\right)\tau x_{2,k} \end{bmatrix} + \mathbf{w}_k \\ \mathbf{y}_k &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\mathbf{x}_k + \mathbf{v}_k \end{aligned} \quad (14)$$

with time constant $\tau = 0.001$ and $\mathbf{w}_k \sim N\left(0, 0.0005^2\mathbf{I}_2\right)$. Imagine there would be a small amount of bias present in the process model. Then, the predictions of the process equation are worthless over large time spans. We need the observations to correct for this error.

1) Hybrid architecture evaluation: The model is simulated for 50.000 time steps. After 25.000 steps, the measurement noise abruptly changes from $\mathbf{v}_k \sim N(0, 0.25\cdot\mathbf{I}_2)$ to a Gaussian mixture as described by (15) below:

$$\begin{aligned}\mathbf{v}_k \sim &\left(\begin{bmatrix}0\\0.1\end{bmatrix} + 0.5\cdot N(0, 0.25\cdot\mathbf{I}_2)\right) + \\ &\left(\begin{bmatrix}-0.1\\0\end{bmatrix} + 0.5\cdot N\left(0, 10\cdot 0.25\cdot\mathbf{I}_2\right)\right)\end{aligned} \quad (15)$$

Fig. 5 shows the values of the switching vector $\boldsymbol{\mu}_k$.

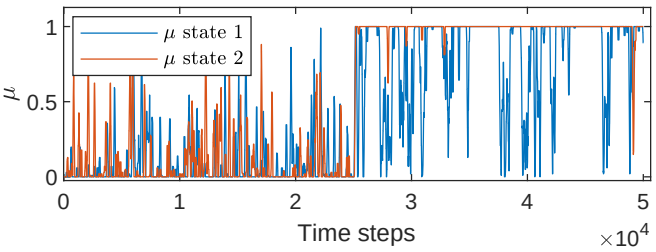

Fig. 5. Development of $\boldsymbol{\mu}$ over time. The graph indicates that the cubature Kalman filter has a large effect on the final state estimate during the first half ($\boldsymbol{\mu}_k$ is close to 0), and that particle filter estimates are mainly used for the Gaussian mixture after 25.000 time steps ($\boldsymbol{\mu}_k$ is close to 1).

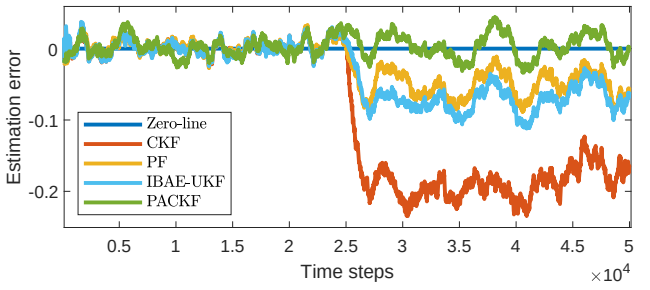

Fig. 6. Estimation errors for the first state of the multivariate nonlinear model of (14). After 25.000 time steps, the measurement noise changes from a Gaussian distribution to a Gaussian mixture model, which is non-Gaussian. Clearly, the proposed PACKF algorithm has the lowest errors in the second half of the simulation.

For comparison, four filters are reviewed: a standard cubature Kalman filter (CKF), a particle filter (PF), an innovation-based adaptive estimation unscented Kalman filter (IBAE-UKF) and the proposed particle-aided cubature Kalman filter

(PACKF) with $N_p = 30$, the adaptive $\tilde{\mathbf{Q}}_k$ matrix of (9) and switching parameter $\boldsymbol{\mu}_k$. For both particle filters, we use simple multinomial resampling and the prior $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$ as proposal distribution. The development of the errors of the first state x_1 is given by Fig. 6.

2) Kernel density estimation evaluation: The model (14) is again simulated for 50.000 time steps. The simulation period is split up into five sectors (I to V) of 10.000 time steps, each with different statistics for the measurement noise $\mathbf{v}_k$. The noises are represented mathematically in the parameterized descriptions of Table I, with the parameter values listed in the table caption. These noises correspond to: sector I - Gaussian, sector II - Gaussian mixture, sector III - Random Walk, sector IV - Poisson noise and sector V - Gaussian.

Sector	Mathematical description
Sector I	$\mathbf{v}_k \sim N\left(0, \sigma_1^2\right)$
Sector II	$\mathbf{v}_k \sim (1-\alpha_2)\, N(\mu, \sigma_2^2) + \alpha_2 N\left(\mu, \beta_2 \sigma_2^2\right)$
Sector III	$\mathbf{v}_k = \alpha_3 \mathbf{v}_{k-1} + \beta_3, \quad \beta_3 \sim N\left(0, \sigma_3^2\right)$
Sector IV	$\mathbf{v}_k \sim \text{Poiss}\left(\lambda_4\right)$
Sector V	$\mathbf{v}_k \sim N\left(0, \sigma_5^2\right)$

TABLE I
THE PARAMETER VALUES THAT ARE USED: $\sigma_1 = 0.5$, $\alpha_2 = 0.5$, $\sigma_2 = 0.5$, $\beta_2 = 10$, $\alpha_3 = 1$, $\sigma_3 = 0.5$, $\lambda_4 = 1$ AND $\sigma_5 = 0.5$.

For both a standard particle filter and a particle filter that uses the proposed kernel density estimation algorithm, the development of the estimation error of the first state x_1 are presented in Fig. 7 below. Under Gaussian circumstances (sectors I and V), the filters perform similarly. In the sectors with non-Gaussian noise (II, III and IV), the standard particle filter exhibits considerably larger estimation errors.

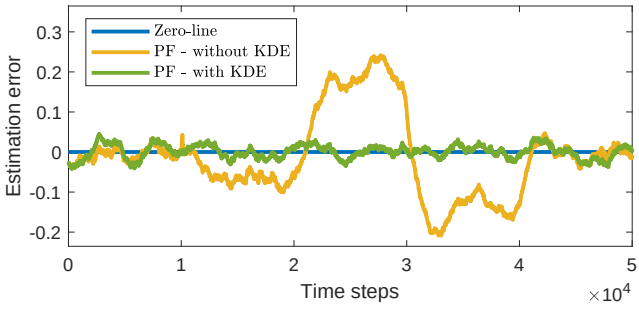

Fig. 7. Estimation error for the first state x_1 of model (14) for two particle filters: one with Gaussian measurement noise assumption (yellow) and a particle filter with the measurement likelihood approximated by the adaptive kernel density estimation algorithm (green). In the non-Gaussian sectors (time step 10.000-40.000), the proposed method performs better in terms of forcing the error towards the blue zero-line.

B. Loosely coupled GNSS/INS-integration

1) Process- and measurement model: In literature, the process model for loosely coupled integration is implemented in various ways, with different notations and assumptions. The model in this subsection contains elements from [15], [21], [22] and [37].

In our localization model, there is no change in the z-coordinate of the vehicle. We assume a vehicle is travelling on a flat surface, and all roll and pitch motions are neglected. These states are still included in the model, but are set to zero. The local navigation frame is denoted as the n-frame, with ENU (East-North-Up) directions. The global frames are the geocentric inertial frame (i-frame) and earth-centered earth-fixed frame (e-frame), for which more information can be found in [38]. The body frame (b-frame) is defined at the INS center, and is given in RFU (Right-Front-Up) convention.

The state vector is denoted as in follows:

$$\mathbf{x} = \begin{bmatrix} \delta\mathbf{p}^T & \delta\mathbf{V}^T & \boldsymbol{\phi}^T & \boldsymbol{\varepsilon}^{\mathrm{b}T} & \boldsymbol{\nabla}^{\mathrm{b}T} \end{bmatrix}^T \tag{16}$$

in which $\delta\mathbf{p} = \begin{bmatrix} \delta L & \delta\lambda & \delta h \end{bmatrix}^T$ is the INS position error in latitude L, longitude λ and height h in the n-frame, $\delta\mathbf{V} = \begin{bmatrix} V_{\mathrm{E}} & \delta V_{\mathrm{N}} & \delta V_{\mathrm{U}} \end{bmatrix}^T$ is the velocity error in the n-frame, and $\boldsymbol{\phi} = \begin{bmatrix} \phi_{\mathrm{E}} & \phi_{\mathrm{N}} & \phi_{\mathrm{U}} \end{bmatrix}^T$ denotes the orientation error in the n-frame. $\boldsymbol{\varepsilon}^{\mathrm{b}} = \begin{bmatrix} \varepsilon_x & \varepsilon_y & \varepsilon_z \end{bmatrix}^T$ is the bias of the gyroscope, given in the b-frame. The accelerometer specific force measurements in the b-frame are defined as $\mathbf{f}^{\mathrm{b}} = \begin{bmatrix} f_x & f_y & f_z \end{bmatrix}^T$, with its bias $\boldsymbol{\nabla}^{\mathrm{b}} = \begin{bmatrix} \nabla_x^{\mathrm{b}} & \nabla_y^{\mathrm{b}} & \nabla_z^{\mathrm{b}} \end{bmatrix}^T$. The biases are both modelled as first-order Markov processes:

$$\begin{aligned} \dot{\boldsymbol{\nabla}}^{\mathrm{b}} &= \boldsymbol{\tau}_a \boldsymbol{\nabla}^{\mathrm{b}} + \mathbf{w}_a \\ \dot{\boldsymbol{\varepsilon}}^{\mathrm{b}} &= \boldsymbol{\tau}_\omega\, \boldsymbol{\varepsilon}^{\mathrm{b}} + \mathbf{w}_\omega \end{aligned} \tag{17}$$

with $\boldsymbol{\tau}_a$ and $\boldsymbol{\tau}_\omega$ as time constants of the accelerator and gyroscope bias, respectively. The time derivative of the orientation error $\dot{\boldsymbol{\phi}}$ is given as:

$$\dot{\boldsymbol{\phi}} = -\boldsymbol{\omega}_{\mathrm{in}}^{\mathrm{n}} \times \boldsymbol{\phi} + \delta\boldsymbol{\omega}_{\mathrm{in}}^{\mathrm{n}} - \mathbf{C}_{\mathrm{b}}^{\mathrm{n}} \boldsymbol{\varepsilon}_{\mathrm{b}}^{\mathrm{b}} \tag{18}$$

in which $\boldsymbol{\omega}_{\mathrm{in}}^{\mathrm{n}}$ denotes the true angle rate and $\delta\boldsymbol{\omega}_{\mathrm{in}}^{\mathrm{n}}$ is its error vector. The term $\mathbf{C}_{\mathrm{b}}^{\mathrm{n}}$ is the rotation matrix from b-frame to n-frame:

$$\mathbf{C}_{\mathrm{b}}^{\mathrm{n}} = \begin{bmatrix} \cos\phi_z & \sin\phi_z & 0 \\ -\sin\phi_z & \cos\phi_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{19}$$

Note that a simplification is made with respect to the rotation matrix in, for example, [39], since roll and pitch motions are neglected in our case. Logically, this holds for both the b-frame and n-frame, and it leads to $\sin\phi_x = 1$, $\cos\phi_y = 1$, $\sin\phi_x = 0$ and $\sin\phi_y = 0$.

The change in the velocity error in the navigation frame can be expressed as:

$$\begin{aligned} \delta\dot{\mathbf{V}}^{\mathrm{n}} = \mathbf{C}_{\mathrm{b}}^{\mathrm{n}}\mathbf{f}^{\mathrm{b}} \times \boldsymbol{\phi} - (2\boldsymbol{\omega}_{\mathrm{ie}}^{\mathrm{n}} + \boldsymbol{\omega}_{\mathrm{en}}^{\mathrm{n}}) \times \delta\boldsymbol{V}^{\mathrm{n}} \\ - (2\delta\boldsymbol{\omega}_{\mathrm{ie}}^{\mathrm{n}} + \delta\boldsymbol{\omega}_{\mathrm{en}}^{\mathrm{n}}) \times \mathbf{V}^{\mathrm{n}} + \mathbf{C}_{\mathrm{b}}^{\mathrm{n}} \nabla^{\mathrm{b}} \end{aligned} \tag{20}$$

In this equation, $\boldsymbol{\omega}_{\mathrm{en}}^{\mathrm{n}}$ is the rotation velocity vector from e-frame to n-frame and $\boldsymbol{\omega}_{\mathrm{ie}}^{\mathrm{n}}$ is the rotation of the earth in the n-frame, with $\delta\boldsymbol{\omega}_{\mathrm{en}}^{\mathrm{n}}$ and $\delta\boldsymbol{\omega}_{\mathrm{ie}}^{\mathrm{n}}$ as the corresponding error vectors. The time derivative of the position error is defined as follows:

$$\delta\dot{\mathbf{p}} = \begin{bmatrix} \frac{\delta V_{\mathrm{N}}}{R_{\mathrm{n}}+h} - \delta h \frac{V_{\mathrm{N}}}{(R_{\mathrm{n}}+h)^2} \\ \frac{\delta V_{\mathrm{E}}}{R_{\mathrm{e}}+h} \sec L + \delta L \frac{V_{\mathrm{E}}}{R_{\mathrm{e}}+h} \tan L \sec L - \delta h \frac{V_{\mathrm{E}} \sec L}{(R_{\mathrm{e}}+h)^2} \\ \delta V_{\mathrm{U}} \end{bmatrix} \tag{21}$$

In this set of equations, R_{n} and R_{e} are the radii of the curvatures in the meridian and prime vertical axis of the earth, respectively. Following [22], $\boldsymbol{\omega}^{\mathrm{n}}_{\mathrm{in}}$ and $\delta\boldsymbol{\omega}^{\mathrm{n}}_{\mathrm{in}}$ can now be described by:

$$\boldsymbol{\omega}^{\mathrm{n}}_{\mathrm{in}} = \begin{bmatrix} -\frac{\delta V_{\mathrm{N}}}{R_{\mathrm{n}}+h} & \omega_{\mathrm{ie}}\cos L + \frac{V_{\mathrm{E}}}{R_{\mathrm{e}}+h}\omega_{\mathrm{ie}}\sin L + \frac{V_{\mathrm{E}}}{R_{\mathrm{e}}+h}\tan L \end{bmatrix}^T$$

$$\delta\boldsymbol{\omega}^{\mathrm{n}}_{\mathrm{in}} = \mathbf{F}_{\mathrm{av}}\delta\mathbf{V}^{\mathrm{n}} + \mathbf{F}_{\mathrm{ap}}\delta\mathbf{p} \tag{22}$$

The matrices $\mathbf{F}_{\mathrm{av}}$ and $\mathbf{F}_{\mathrm{ap}}$ are included in Appendix B. Substituting (22) into (20) leads to (23) below:

$$\begin{aligned}\delta\dot{\mathbf{V}}^{\mathrm{n}} =& \boldsymbol{C}^{\mathrm{n}}_{\mathrm{b}}\mathbf{f}^{\mathrm{b}} \times \boldsymbol{\phi} - (2\boldsymbol{\omega}^{\mathrm{n}}_{\mathrm{ie}} + \boldsymbol{\omega}^{\mathrm{n}}_{\mathrm{en}}) \times \delta\mathbf{V}^{\mathrm{n}} - \\ & (\delta\boldsymbol{\omega}^{\mathrm{n}}_{\mathrm{ie}} + \mathbf{F}_{\mathrm{av}}\delta\mathbf{V}^{\mathrm{n}} + \mathbf{F}_{\mathrm{ap}}\delta\mathbf{p}) \times \mathbf{V}^{\mathrm{n}} + \mathbf{C}^{\mathrm{n}}_{\mathrm{b}}\nabla^{\mathrm{b}}\end{aligned} \tag{23}$$

For each of the states in the vector $\mathbf{x}$ of (16), we have now described the continuous dynamics. With the forward Euler discretization method [40], the set of equations is discretized. From now on, we will proceed with this discrete version.

GNSS- and INS location measurements are generated in the local n-frame. The velocities $v_{x,k}$ and $v_{y,k}$ and the yaw rate $\omega_{z,k}$ are expressed in the b-frame. The INS measurements in the n-frame can now be modelled as:

$$\begin{aligned}\mathbf{p}^{\mathrm{INS}}_k &= \begin{bmatrix} x^{\mathrm{INS}}_k & y^{\mathrm{INS}}_k & z^{\mathrm{INS}}_k \end{bmatrix}^T \\ \mathbf{V}^{\mathrm{INS}}_k &= \mathbf{C}^{\mathrm{n}}_{\mathrm{b},k}\begin{bmatrix} v^{\mathrm{INS}}_{x,k} & v^{\mathrm{INS}}_{y,k} & v^{\mathrm{INS}}_{z,k} \end{bmatrix}^T\end{aligned} \tag{24}$$

The same holds for the GNSS measurements, they can be described in the local n-frame by:

$$\begin{aligned}\mathbf{p}^{\mathrm{GNSS}}_k &= \begin{bmatrix} x^{\mathrm{GNSS}}_k & y^{\mathrm{GNSS}}_k & z^{\mathrm{GNSS}}_k \end{bmatrix}^T \\ \mathbf{V}^{\mathrm{GNSS}}_k &= \mathbf{C}^{\mathrm{n}}_{\mathrm{b},k}\begin{bmatrix} v^{\mathrm{GNSS}}_{x,k} & v^{\mathrm{GNSS}}_{y,k} & v^{\mathrm{GNSS}}_{z,k} \end{bmatrix}^T\end{aligned} \tag{25}$$

In the loosely coupled GNSS/INS-integration architecture, we attempt to find the error of the INS with respect to the true states. Later on, this error is then subtracted from the INS measurements to determine a final state estimate. The observations for our filter are then, logically, obtained by subtracting the GNSS observations from the INS observations:

$$\mathbf{z}_k = \begin{bmatrix}\delta\mathbf{p} \\ \delta\mathbf{V}\end{bmatrix} + \mathbf{v}_k = \begin{bmatrix}\mathbf{p}^{\mathrm{INS}}_k - \mathbf{p}^{\mathrm{GNSS}}_k \\ \mathbf{V}^{\mathrm{INS}}_k - \mathbf{V}^{\mathrm{GNSS}}_k\end{bmatrix} \tag{26}$$

The fact that the measurement equation of (26) calculates the errors between GNSS and INS has no influence on the uncertainty of the measurements. That is the case because the INS errors over one time step are very small compared to the GNSS errors. In other words, we assume that the standard errors of the GNSS device are equal to the standard errors that we can use for constructing the measurement noise covariance $\mathbf{R}$ of our filtering algorithm.

2) Ground truth- and measurement data acquisition: Ground-truth data is generated with a 10-dimensional model based on configuration 1 of [41]. The process noise is $\mathbf{w} = 0$ for all time steps k. We simulate 4 minutes of driving in east direction, with both the GNSS- and INS update rate at 10Hz. In each simulation, we randomly generate three different inputs, which each will be used for a third of the simulation time. In each of the three periods, the acceleration is set such that the longitudinal speed ranges from 50 km/h to 110 km/h. An arbitrary, mild steering angle ($\delta < 0.004$ rad) is used to resemble highway driving conditions.

The GNSS measurement data is obtained by artificially adding measurement noise to the ground-truth data. A second simulation with the 10-dimensional model will be performed to obtain INS measurements with bias. Logically, the process noise $\mathbf{w}_k$ is now nonzero. From this second simulation, the INS states are extracted. These will be used as INS measurement data. This process of noise injection is a broadly adopted way of simulating noise in order to validate localization algorithms [42], [43], [44].

This noise injection method requires some fixed values for the matrices $\mathbf{Q}$ and $\mathbf{R}$. The process noise covariance matrix $\mathbf{Q}$ can be chosen at the hand of sensor specifications. It is dynamic because of the rotation matrix $\mathbf{C}^{\mathrm{n}}_{\mathrm{b}}$, that depends on the yaw angle $\phi_{z,k}$ at time k. Both the accelerometer and the gyroscope experience noise- and bias errors, which are included as standard errors $\boldsymbol{\sigma}_{\mathrm{a}}$, $\boldsymbol{\sigma}_{\omega}$, $\boldsymbol{\sigma}_{\mathrm{b,a}}$ and $\boldsymbol{\sigma}_{\mathrm{b},\omega}$, respectively. For the state vector of (16) the process noise model, with $\mathbf{w}_k \sim N(0, \mathbf{Q}_k)$, then has the following covariance matrix:

$$\mathbf{Q}_k = \begin{bmatrix} 0_3 & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & \mathbf{C}^{\mathrm{n}}_{\mathrm{b},k} & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & \mathbf{C}^{\mathrm{n}}_{\mathrm{b},k} & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & \mathbf{I}_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} 0 \\ \boldsymbol{\sigma}^2_{\mathrm{a}} \\ \boldsymbol{\sigma}^2_{\omega} \\ \boldsymbol{\sigma}^2_{\mathrm{b,a}} \\ \boldsymbol{\sigma}^2_{\mathrm{b},\omega} \end{bmatrix} \tag{27}$$

For the measurement noise covariance matrix $\mathbf{R}$, the specifications of a considered GNSS device can be used as follows:

$$\mathbf{R} = \begin{bmatrix} \mathbf{I}_3\,\boldsymbol{\sigma}^2_{\mathrm{pos}} & 0_3 \\ 0_3 & \mathbf{I}_3\,\boldsymbol{\sigma}^2_{\mathrm{vel}} \end{bmatrix} \tag{28}$$

The positioning and velocity standard error vectors are denoted as $\boldsymbol{\sigma}_{\mathrm{pos}}$ and $\boldsymbol{\sigma}_{\mathrm{vel}}$, respectively. Their values, together with the entries for matrix $\mathbf{Q}$, can be found in Table II.

Standard error type	Symbol	Value	Unit
GNSS position	$\boldsymbol{\sigma}_{\mathrm{pos}}$	4	m
GNSS velocity	$\boldsymbol{\sigma}_{\mathrm{vel}}$	1	m/s
Accelerometer noise	$\boldsymbol{\sigma}_{\mathrm{a}}$	0.03	$\mathrm{mg}\sqrt{\mathrm{Hz}}$
Accelerometer bias	$\boldsymbol{\sigma}_{\mathrm{b,a}}$	0.015	mg
Gyroscope noise	$\boldsymbol{\sigma}_{\omega}$	0.006	$/\mathrm{s}\sqrt{\mathrm{Hz}}$
Gyroscope bias	$\boldsymbol{\sigma}_{\mathrm{b},\omega}$	3	/h

TABLE II
STANDARD ERRORS BASED ON THE SENSOR SPECIFICATIONS OF [45].

Data from a GNSS device is generated by adding the noise sequence of Table III to the ground truth data. For the GNSS noise, the simulation period is split up into 7 sectors, each with a different type of measurement noise. These noise types, and the environments in which they might occur, are as follows: Gaussian (sector I, open sky environments), Gaussian mixture (sector II, outlier measurements due to change in number of receivable satellites), Gaussian with varying variance (sector III, urban environments), flicker noise (sector IV, changing reception conditions), Gaussian with extremely high variance (sector V, GNSS outages), Random Walk (sector VI, multipath phenomena) and again Gaussian (sector VII). The occurrence of these noise types in specific environments has been studied in [41], [46], [47], [48]. A mathematical description of these noise types can also be found in Table III.

Sector	Time period	Mathematical description
Sector I	0s-40s	$\mathbf{v}_k \sim N(0, \mathbf{R})$
Sector II	40s-80s	$\mathbf{v}_k \sim 0.9N(0, \mathbf{R}) + 0.1N(0, 300\,\mathbf{R})$
Sector III	80s-120s	$\mathbf{v}_k \sim N(0, \alpha\mathbf{R}), \quad \alpha \in [1, 5]$
Sector IV	120s-160s	$S(f) \propto \frac{1}{f}$
Sector V	160s-170s	$\mathbf{v}_k \sim N(0, 100\,\mathbf{R})$,
Sector VI	170s-210s	$\mathbf{v}_k = 0.6\,\mathbf{v}_{k-1} + \beta, \quad \beta \sim N(0, \mathbf{R})$
Sector VII	210s-240s	$\mathbf{v}_k \sim N(0, \mathbf{R})$

TABLE III
SETTINGS FOR THE GNSS MEASUREMENT NOISE SEQUENCE.

3) Results in terms of localization errors: The goal of the GNSS/INS-integration is to find an accurate location at the hand of a set of GNSS and INS measurements. As a performance metric in achieving this goal, the RMSE between the true position and the position that is estimated by the filters, is used. The localization RMSE is calculated in the Cartesian, local navigation frame (n-frame) as follows:

$$\mathrm{RMSE}_{\mathrm{pos}} = \frac{1}{N}\sum_{k=1}^{N}\sqrt{\left((x_k - \hat{x}_k)^2 + (y_k - \hat{y}_k)^2\right)} \tag{29}$$

For two simulations, the GNSS location data, as well as the INS location data, are given in Fig. 8 below. As can be seen in the x,y-plane of the graphs, the INS drifts away from the GNSS track. The period of GNSS outage (noise with extremely high variance) can be spot easily.

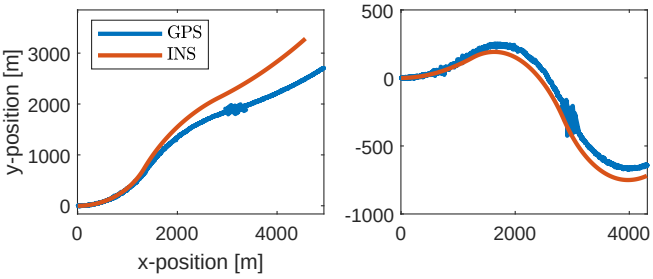

Fig. 8. In the x,y-plane; GNSS and INS location measurements of two randomly generated paths in east direction.

A total of 25 separate Monte Carlo runs are performed. The filters that will be used for evaluation are: a standard cubature Kalman filter (CKF), a particle filter (PF), a cubature particle filter (CKFAPF), an innovation-based adaptive estimation unscented Kalman filter (IBAE-UKF) and the proposed particle-aided cubature Kalman filter (PACKF). The cubature particle filter has a proposal distribution that is provided by a cubature Kalman filter output. Thus, in this case, the proposal distribution also includes information on the latest observation. The particle filter has the exact same settings as the one in the proposed method; the proposal distribution of (10), a multinomial resampling strategy and the same number of particles ($N_p = 50$). The resampling process usually starts when a threshold N_T for the effective sample size is exceeded:

$$\hat{N}_{\mathrm{eff}} = \frac{1}{\sum_{i=1}^{N_p}\left(w_k^i\right)^2} \leq N_T \tag{30}$$

with w_k^i as particle weights. Regular particle filters resample or re-use particles from the previous time step. This would be a good approach for systems with large process uncertainty. In our specific case, however, it makes more sense to intensively resample around the predictions, since these are already very accurate. Re-using old particles would lead to very high resampling rates, since they quickly become inaccurate compared to the predictions. One might then as well just resample all particles around the predictions at every time step. This is basically a sequential-importance-resampling (SIR) particle filter with a high threshold $N_T = N_p$.

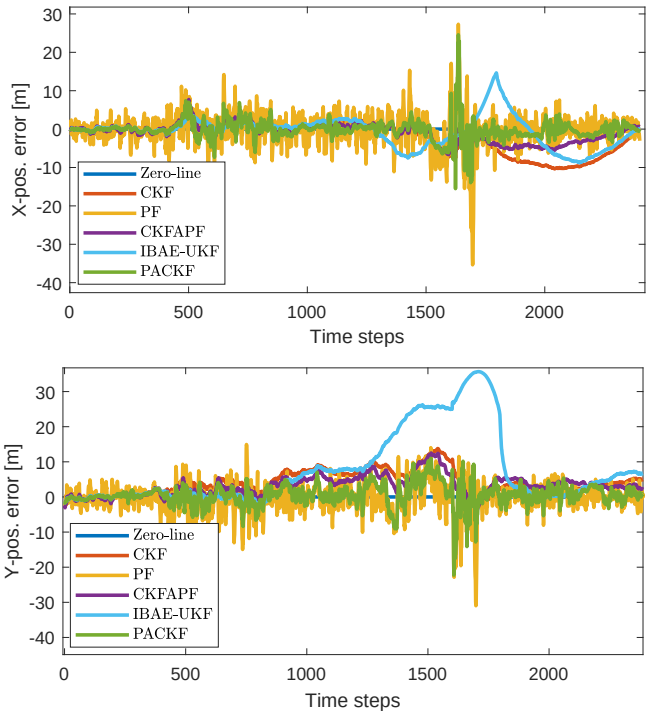

Fig. 9. The x-position estimation errors for several filtering methods. Though the algorithms perform similarly in earlier sectors, there is a clear difference in performance in the later sectors. The sector with GNSS outage seems, however, still problematic for the proposed filtering method.

For sectors I-VI, the estimation errors of the x-position and y-position of the local n-frame are presented in Fig. 9. The particle filter estimate and, to a lesser extent, the particle-aided cubature Kalman filter estimate stand out by their noisy character. It is suspected that this is a result of our choice for the proposal distribution and resampling strategy. After all, we do not re-use any knowledge from the last time step. So the final estimate is sensitive to the current measurements. We see, however, that lower thresholds N_T lead to higher inaccuracy, since particle degeneracy happens rather quickly. The estimation errors turn out to be the lowest with these settings.

Filter	RMSE
CKF	6.42
PF	3.20
CKFAPF	5.49
IBAE-UKF	10.93
PACKF	2.26

TABLE IV
AVERAGE RMSE VALUES OVER THE ENTIRE SIMULATION PERIOD.

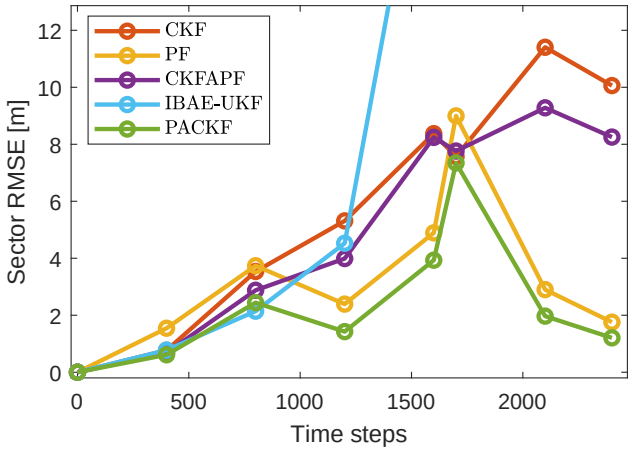

Fig. 10. The average RMSE of 25 Monte Carlo runs at the end of each sector. The proposed filter has the lowest errors in all but one sector.

Let it be noted that the standard particle filter performs adequate in the non-Gaussian domain. Under Gaussian measurement noise, however, it performs much worse than the other algorithms. Since it is expected that the majority of the measurement noise will be (close to) Gaussian, particle filters would not be that competitive under normal circumstances. Furthermore, the innovation-based adaptive estimation unscented Kalman filter experiences problems with most non-Gaussian noise types. It adjusts its setting for $\mathbf{R}$ to reduce the estimation errors, but the innovation does not become much smaller after a while, due to the non-Gaussian nature that can not be captured. It seems as if the standard algorithm suffers from this feature, rather than that it is serving it.

VI. Conclusion

GNSS/INS-based vehicle localization is considered to be a key element of future automated driving systems. One of the problems of this approach, is that the GNSS measurement noise statistics are highly dynamic and, depending on the specific environment, might follow a non-Gaussian distribution. To overcome this problem, a hybrid filtering architecture has been proposed. The proposed particle-aided cubature Kalman filter exploits a soft switching strategy, an adaptive, instrumental matrix $\tilde{\mathbf{Q}}$ and a kernel density estimation algorithm. After generating ground-truth data and injecting common non-Gaussian noise sequences in the measurement data, the proposed particle-aided cubature Kalman filter is implemented in a loosely coupled GNSS/INS-integration architecture. The results in terms of localization errors are reviewed, and the proposed method shows to beat traditional filtering methods. It is suspected that the noisy character of the filtered estimates is the results of the choice for our proposal distribution and resampling strategy. Future research could focus on revising these filter settings, applications of the hybrid filter is other domains, or could concentrate on learning approaches to mitigate the effects of GPS outages.

Appendix A: Full Algorithm Description

A square-root version of the cubature Kalman filter is used [12]. This implementation has a better numerical stability. The Tria-operator denotes any arbitrary triangularization algorithm, such as the Cholesky factorization [49].

Cubature Kalman filter prediction

1. Decompose last covariance

$$\mathbf{S}_{k-1|k-1} = \mathrm{Tria}\left(\mathbf{P}_{k-1|k-1}\right)$$

2. Calculate cubature points

$$\boldsymbol{X}_{i,k-1|k-1} = \mathbf{S}_{k-1|k-1}\xi_i + \hat{\mathbf{x}}_{k-1|k-1}$$

3. Propagate through f

$$\boldsymbol{X}^*_{i,k|k-1} = f\left(\boldsymbol{X}_{i,k-1|k-1}\right)$$

4. Predict state

$$\hat{\mathbf{x}}_{k|k-1} = \frac{1}{m}\sum_{i=1}^{m}\boldsymbol{X}^*_{i,k|k-1}$$

5. Calculate centered matrix

$$\mathcal{X}^*_{k|k-1} = \frac{1}{\sqrt{m}}\left[\boldsymbol{X}^*_{1,k|k-1} - \hat{\mathbf{x}}_{k|k-1}\cdots\ \boldsymbol{X}^*_{m,k|k-1} - \hat{\mathbf{x}}_{k|k-1}\right]$$

6. Calculate $\tilde{\mathbf{Q}}_k$

$$\mathbf{Q}_k = \rho\tilde{\mathbf{y}}^T_{k-1|k-1}\tilde{\mathbf{y}}_{k-1|k-1}\mathbf{I}_n + \eta\mathbf{I}_n$$

7. Decompose $\tilde{\mathbf{Q}}_k$

$$\mathbf{S}_{Q,k} = \mathrm{Tria}\left(\tilde{\mathbf{Q}}_k\right)$$

8. Square-root of covariance

$$\mathbf{S}_{k|k-1} = \mathrm{Tria}\left(\begin{bmatrix}\mathcal{X}^*_{k|k-1} & \mathbf{S}_{Q,k}\end{bmatrix}\right)$$

9. Calculate covariance

$$\mathbf{P}_{k|k-1} = \mathbf{S}_{k|k-1}\mathbf{S}^T_{k|k-1}$$

Particle filter

10. Draw particles from proposal distribution

$$\mathbf{x}^i_k \sim q(\mathbf{x}_k \mid \mathbf{z}_k, \mathbf{x}_{k-1})$$

11. Update particle weights

$$w^i_k \propto w^j_{k-1}\cdot\frac{p\left(\mathbf{z}_k \mid \mathbf{x}^i_k\right)p\left(\mathbf{x}^i_k \mid \mathbf{x}^j_{k-1}\right)}{q\left(\mathbf{x}^i_k \mid \mathbf{x}^j_{k-1}, \mathbf{z}_k\right)}$$

12. Calculate particle filter estimate

$$\hat{\mathbf{x}}^{\mathrm{pf}}_{k|k} \approx \sum_{i=1}^{N} w^i_{k|k}\mathbf{x}^i_k$$

13. Propagate through h

$$\mathbf{z}^{\mathrm{pf}}_k = h\left(\hat{\mathbf{x}}^{\mathrm{pf}}_{k|k}\right)$$

Cubature Kalman filter update

14. Evaluate cubature points

$$\boldsymbol{X}_{i,k|k-1} = \mathbf{S}_{k|k-1}\xi_i + \hat{\mathbf{x}}_{k|k-1}$$

15. Propagate through h

$$\boldsymbol{Z}_{i,k|k-1} = h\left(\boldsymbol{X}_{i,k|k-1}\right)$$

16. Predict measurement

$$\hat{\mathbf{z}}_{k|k-1} = \frac{1}{m}\sum_{i=1}^{m}\boldsymbol{Z}_{i,k|k-1}$$

17. Calculate centered matrix

$$\mathcal{Z}_{k|k-1} = \frac{1}{\sqrt{m}}\left[\boldsymbol{Z}_{1,k|k-1} - \hat{\mathbf{z}}_{k|k-1}\cdots\ \boldsymbol{Z}_{m,k|k-1} - \hat{\mathbf{z}}_{k|k-1}\right]$$

18. Decompose $\tilde{\mathbf{R}}_{\mathbf{k}}$

$$\mathbf{S}_{R,k} = \mathrm{Tria}\left(\left(1-\boldsymbol{\mu}_k\right)\mathbf{R}\right)$$

19. Decompose $\tilde{\mathbf{y}}$ covariance

$$\mathbf{S}_{zz,k|k-1} = \mathrm{Tria}\left(\begin{bmatrix}\mathcal{Z}_{k|k-1} & \mathbf{S}_{R,k}\end{bmatrix}\right)$$

20. Calculate centered matrix

$$\mathcal{X}_{k|k-1} = \frac{1}{\sqrt{m}}\left[\boldsymbol{X}_{1,k|k-1} - \hat{\mathbf{x}}_{k|k-1}\cdots\ \boldsymbol{X}_{m,k|k-1} - \hat{\mathbf{x}}_{k|k-1}\right]$$

21. Calculate cross-covariance

$$\mathbf{P}_{xz,k|k-1} = \mathcal{X}_{k|k-1}\mathcal{Z}_{k|k-1}^{T}$$

22. Determine Kalman gain

$$\mathbf{K}_k = \left(\mathbf{P}_{xz,k|k-1}/\mathbf{S}_{zz,k|k-1}^{T}\right)/\mathbf{S}_{zz,k|k-1}$$

23. Calculate observation

$$\tilde{\mathbf{z}}_{k|k} = \left(1-\boldsymbol{\mu}_k\right)\mathbf{z}_k + \boldsymbol{\mu}_k\,\mathbf{z}_k^{\mathrm{pf}}$$

24. Calculate state estimate

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\tilde{\mathbf{z}}_{k|k} - \hat{\mathbf{z}}_{k|k-1}\right)$$

25. Square-root of covariance

$$\mathbf{S}_{k|k} = \mathrm{Tria}\left(\begin{bmatrix}\mathcal{X}_{k|k-1} - \mathbf{K}_k\mathcal{Z}_{k|k-1} & \mathbf{K}_k\,\mathrm{Tria}(\mathbf{R})\end{bmatrix}\right)$$

APPENDIX B: TRUE ANGLE RATE DESCRIPTION ($\delta\boldsymbol{\omega}_{\mathrm{IN}}^{\mathrm{N}}$)

$$\delta\boldsymbol{\omega}_{\mathrm{in}}^{\mathrm{n}} = \mathbf{F}_{\mathrm{av}}\delta\mathbf{V}^{\mathrm{n}} + \mathbf{F}_{\mathrm{ap}}\delta\mathbf{p} =$$

$$\begin{bmatrix} -\frac{\delta V_{\mathrm{N}}}{R_{\mathrm{n}}+h} + \delta h\frac{V_{\mathrm{N}}}{(R_{\mathrm{n}}+h)^2} \\ -\delta L\omega_{\mathrm{ie}}\sin L + \frac{\delta V_{\mathrm{E}}}{R_{\mathrm{e}}+h} - \delta h\frac{V_{\mathrm{E}}}{(R_{\mathrm{e}}+h)^2} \\ \delta L\omega_{\mathrm{ie}}\cos L + \frac{\delta V_{\mathrm{E}}}{R_{\mathrm{e}}+h}\tan L + \delta L\frac{V_{\mathrm{E}}}{R_{\mathrm{e}}+h}\sec^2 L - \delta h\frac{V_{\mathrm{E}}\tan L}{(R_{\mathrm{e}}+h)^2} \end{bmatrix}$$

APPENDIX C: LOCALIZATION RMSE PER SECTOR

Sector	I	II	III	IV	V	VI	VII
CKF	0.77	3.54	5.31	8.37	7.57	11.40	10.06
PF	1.54	3.74	2.39	4.89	9.00	2.90	1.76
CKFAPF	0.68	2.88	3.99	8.24	7.76	9.28	8.25
IBAE-UKF	0.78	2.14	4.53	21.44	31.72	18.28	15.19
PACKF	0.61	2.44	1.43	3.93	7.34	1.97	1.21

TABLE V
AVERAGE RMSE VALUES OF THE GNSS/INS INTEGRATION PROCESS FOR 25 MONTE CARLO RUNS.

REFERENCES

[1] R. C. Shit, "Precise localization for achieving next-generation autonomous navigation: State-of-the-art, taxonomy and future prospects," *Computer Communications*, 2020.

[2] K.-W. Chiang, G.-J. Tsai, H.-J. Chu, and N. El-Sheimy, "Performance enhancement of ins/gnss/refreshed-slam integration for acceptable lane-level navigation accuracy," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 2463–2476, 2020.

[3] D. Gruyer, V. Magnier, K. Hamdi, L. Claussmann, O. Orfila, and A. Rakotonirainy, "Perception, information processing and modeling: Critical stages for autonomous driving applications," *Annual Reviews in Control*, vol. 44, pp. 323–341, 2017.

[4] S. Kuutti, S. Fallah, K. Katsaros, M. Dianati, F. Mccullough, and A. Mouzakitis, "A survey of the state-of-the-art localization techniques and their potentials for autonomous vehicle applications," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 829–846, 2018.

[5] D. Li, X. Jia, and J. Zhao, "A novel hybrid fusion algorithm for low-cost gps/ins integrated navigation system during gps outages," *IEEE Access*, vol. 8, pp. 53 984–53 996, 2020.

[6] C. Kim, S. Cho, M. Sunwoo, P. Resende, B. Bradaï, and K. Jo, "Updating point cloud layer of high definition (hd) map based on crowd-sourcing of multiple vehicles installed lidar," *IEEE Access*, vol. 9, pp. 8028–8046, 2021.

[7] W.-W. Kao, "Integration of gps and dead-reckoning navigation systems," in *Vehicle Navigation and Information Systems Conference, 1991*, vol. 2. IEEE, 1991, pp. 635–643.

[8] R. E. Kalman, "A new approach to linear filtering and prediction problems," 1960.

[9] G. L. Smith, S. F. Schmidt, and L. A. McGee, *Application of statistical filter theory to the optimal estimation of position and velocity on board a circumlunar vehicle*. National Aeronautics and Space Administration, 1962.

[10] S. J. Julier and J. K. Uhlmann, "New extension of the kalman filter to nonlinear systems," in *Signal processing, sensor fusion, and target recognition VI*, vol. 3068. International Society for Optics and Photonics, 1997, pp. 182–193.

[11] E. A. Wan and R. Van Der Merwe, "The unscented kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*. Ieee, 2000, pp. 153–158.

[12] I. Arasaratnam and S. Haykin, "Cubature kalman filters," *IEEE Transactions on automatic control*, vol. 54, no. 6, pp. 1254–1269, 2009.

[13] T. T. NGOC, A. KHENCHAF, and F. COMBLET, "Evaluating process and measurement noise in extended kalman filter for gnss position accuracy," in *2019 13th European Conference on Antennas and Propagation (EuCAP)*. IEEE, 2019, pp. 1–5.

[14] A. Almagbile, J. Wang, and W. Ding, "Evaluating the performances of adaptive kalman filter methods in gps/ins integration," *Journal of Global Positioning Systems*, vol. 9, no. 1, pp. 33–40, 2010.

[15] Y. Liu, X. Fan, C. Lv, J. Wu, L. Li, and D. Ding, "An innovative information fusion method with adaptive kalman filter for integrated ins/gps navigation of autonomous vehicles," *Mechanical Systems and Signal Processing*, vol. 100, pp. 605–616, 2018.

[16] K. Feng, J. Li, X. Zhang, X. Zhang, C. Shen, H. Cao, Y. Yang, and J. Liu, "An improved strong tracking cubature kalman filter for gps/ins integrated navigation systems," *Sensors*, vol. 18, no. 6, p. 1919, 2018.

[17] G. Hu, W. Wang, Y. Zhong, B. Gao, and C. Gu, "A new direct filtering approach to ins/gnss integration," *Aerospace Science and Technology*, vol. 77, pp. 755–764, 2018.

[18] Q. Xu, X. Li, and C.-Y. Chan, "Enhancing localization accuracy of mems-ins/gps/in-vehicle sensors integration during gps outages," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 8, pp. 1966–1978, 2018.

[19] H. Jiang, C. Shi, T. Li, Y. Dong, Y. Li, and G. Jing, "Low-cost gps/ins integration with accurate measurement modeling using an extended state observer," *GPS Solutions*, vol. 25, no. 1, pp. 1–15, 2021.

[20] H. Xiong, Z. Mai, J. Tang, and F. He, "Robust gps/ins/dvl navigation and positioning method using adaptive federated strong tracking filter based on weighted least square principle," *IEEE Access*, vol. 7, pp. 26 168–26 178, 2019.

[21] G. Hu, B. Gao, Y. Zhong, and C. Gu, "Unscented kalman filter with process noise covariance estimation for vehicular ins/gps integration system," *Information Fusion*, vol. 64, pp. 194–204, 2020.

[22] R. Song, X. Chen, Y. Fang, and H. Huang, "Integrated navigation of gps/ins based on fusion of recursive maximum likelihood imm and square-root cubature kalman filter," *ISA transactions*, vol. 105, pp. 387–395, 2020.

[23] F. Wu, H. Luo, H. Jia, F. Zhao, Y. Xiao, and X. Gao, "Predicting the noise covariance with a multitask learning model for kalman filter-based gnss/ins integrated navigation," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–13, 2020.

[24] S. Chakravorty *et al.*, "An unscented kalman-particle hybrid filter for space object tracking," *The Journal of the Astronautical Sciences*, vol. 65, no. 1, pp. 111–134, 2018.

[25] R. Van Der Merwe, A. Doucet, N. De Freitas, and E. Wan, "The unscented particle filter," *Advances in neural information processing systems*, vol. 13, pp. 584–590, 2000.

[26] M. Lin and B. Kim, "Discriminative parameter training of the extended particle-aided unscented kalman filter for vehicle localization," *Applied Sciences*, vol. 10, no. 18, p. 6260, 2020.

[27] ——, "Extended particle-aided unscented kalman filter based on self-driving car localization," *Applied Sciences*, vol. 10, no. 15, p. 5045, 2020.

[28] M. Lin, J. Yoon, and B. Kim, "Self-driving car location estimation based on a particle-aided unscented kalman filter," *Sensors*, vol. 20, no. 9, p. 2544, 2020.

[29] N. J. Gordon, D. J. Salmond, and A. F. Smith, "Novel approach to nonlinear/non-gaussian bayesian state estimation," in *IEE Proceedings F-radar and signal processing*, vol. 140, no. 2. IET, 1993, pp. 107–113.

[30] F. J. Massey Jr, "The kolmogorov-smirnov test for goodness of fit," *Journal of the American statistical Association*, vol. 46, no. 253, pp. 68–78, 1951.

[31] R. Mansuy and M. Yor, *Aspects of Brownian motion*. Springer Science & Business Media, 2008.

[32] M. Boutayeb and D. Aubry, "A strong tracking extended kalman observer for nonlinear discrete-time systems," *IEEE Transactions on Automatic Control*, vol. 44, no. 8, pp. 1550–1556, 1999.

[33] J. Zarei and E. Shokri, "Convergence analysis of non-linear filtering based on cubature kalman filter," *IET Science, Measurement & Technology*, vol. 9, no. 3, pp. 294–305, 2015.

[34] R. A. Davis, K.-S. Lii, and D. N. Politis, "Remarks on some nonparametric estimates of a density function," in *Selected Works of Murray Rosenblatt*. Springer, 2011, pp. 95–100.

[35] E. Parzen, "On estimation of a probability density function and mode," *The annals of mathematical statistics*, vol. 33, no. 3, pp. 1065–1076, 1962.

[36] V. A. Epanechnikov, "Non-parametric estimation of a multivariate probability density," *Theory of Probability & Its Applications*, vol. 14, no. 1, pp. 153–158, 1969.

[37] M. Aftatah, A. Lahrech, A. Abounada, and A. Soulhi, "Gps/ins/odometer data fusion for land vehicle localization in gps denied environment," *Modern applied science*, vol. 11, no. 1, p. 62, 2016.

[38] A. Budiyono, "Principles of gnss, inertial, and multi-sensor integrated navigation systems," *Industrial Robot: An International Journal*, 2012.

[39] Y. Zhang, C. Shen, J. Tang, and J. Liu, "Hybrid algorithm based on mdf-ckf and rf for gps/ins system during gps outages (april 2018)," *IEEE Access*, vol. 6, pp. 35 343–35 354, 2018.

[40] L. Euler, *Institutiones calculi integralis*. Academia Imperialis Scientiarum, 1794, vol. 4.

[41] J. Huang and H.-S. Tan, "A low-order dgps-based vehicle positioning system under urban environment," *IEEE/ASME Transactions on mechatronics*, vol. 11, no. 5, pp. 567–575, 2006.

[42] N. M. Drawil and O. Basir, "Intervehicle-communication-assisted localization," *IEEE transactions on intelligent transportation systems*, vol. 11, no. 3, pp. 678–691, 2010.

[43] K. Liu, H. B. Lim, E. Frazzoli, H. Ji, and V. C. Lee, "Improving positioning accuracy using gps pseudorange measurements for cooperative vehicular localization," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 6, pp. 2544–2556, 2013.

[44] R. Parker and S. Valaee, "Vehicular node localization using received-signal-strength indicator," *IEEE Transactions on Vehicular Technology*, vol. 56, no. 6, pp. 3371–3380, 2007.

[45] "Inertial sensors: Imu, gps-ins, ahrs, mru for uavs & robotics," https://www.unmannedsystemstechnology.com/company/inertial-labs/, (Accessed on 10/01/2021).

[46] C. Ordonez, J. Martínez, J. Rodríguez-Pérez, and A. Reyes, "Detection of outliers in gps measurements by using functional-data analysis," *Journal of Surveying Engineering*, vol. 137, no. 4, pp. 150–155, 2011.

[47] F. A. Ghaleb, A. Zainal, M. A. Rassam, and A. Abraham, "Improved vehicle positioning algorithm using enhanced innovation-based adaptive kalman filter," *Pervasive and Mobile Computing*, vol. 40, pp. 139–155, 2017.

[48] A. Santamaría-Gómez, M.-N. Bouin, X. Collilieux, and G. Wöppelmann, "Correlated errors in gps position time series: Implications for velocity estimates," *Journal of Geophysical Research: Solid Earth*, vol. 116, no. B1, 2011.

[49] C. Brezinski, "La méthode de cholesky," *Revue d'histoire des mathématiques*, vol. 11, no. 2, pp. 205–238, 2005.

Bibliography

[1] Automated vehicles for safety | nhtsa. https://www.nhtsa.gov/technology-innovation/automated-vehicles#resources. (Accessed on 03/02/2021).

[2] Does your device really need machine learning? - ceva's experts blog. https://www.ceva-dsp.com/ourblog/to-learn-or-not-to-learn-does-your-device-really-need-machine-learning/. (Accessed on 05/20/2021).

[3] Inertial sensors: Imu, gps-ins, ahrs, mru for uavs & robotics. https://www.unmannedsystemstechnology.com/company/inertial-labs/. (Accessed on 10/01/2021).

[4] Optimal estimation algorithms: Kalman and particle filters - kdnuggets. https://www.kdnuggets.com/2020/02/optimal-estimation-algorithms-kalman-particle-filters.html. (Accessed on 05/20/2021).

[5] Hamed H Afshari, S Andrew Gadsden, and S Habibi. Gaussian filters for parameter and state estimation: A general review of theory and recent trends. *Signal Processing*, 135:218–238, 2017.

[6] Mohammed Aftatah, Abdelkabir Lahrech, Abdelouahed Abounada, and Aziz Soulhi. Gps/ins/odometer data fusion for land vehicle localization in gps denied environment. *Modern applied science*, 11(1):62, 2016.

[7] Ali Almagbile, Jinling Wang, and Weidong Ding. Evaluating the performances of adaptive kalman filter methods in gps/ins integration. *Journal of Global Positioning Systems*, 9(1):33–40, 2010.

[8] Ienkaran Arasaratnam and Simon Haykin. Cubature kalman filters. *IEEE Transactions on automatic control*, 54(6):1254–1269, 2009.

[9] Ravindra Babu and Jinling Wang. Real-time data analysis of ultra-tight gps/ins integration. In *Proceedings of the International Global Navigation Satellite Systems Society IGNSS Symposium, The University of New South Wales, Sydney, Australia*, pages 1–12, 2007.

[10] T. Bayes. An essay towards solving a problem in the doctrine of chances. *Phil. Trans. of the Royal Soc. of London*, 53:370–418, 1763.

[11] Carlo Berzuini, Nicola G Best, Walter R Gilks, and Cristiana Larizza. Dynamic conditional independence models and markov chain monte carlo methods. *Journal of the American Statistical Association*, 92(440):1403–1412, 1997.

[12] Robert Bos, Stijn De Waele, and Piet MT Broersen. Autoregressive spectral estimation by application of the burg algorithm to irregularly sampled data. *IEEE Transactions on Instrumentation and Measurement*, 51(6):1289–1294, 2002.

[13] Mohamed Boutayeb and Didier Aubry. A strong tracking extended kalman observer for nonlinear discrete-time systems. *IEEE Transactions on Automatic Control*, 44(8):1550–1556, 1999.

[14] Claude Brezinski. La méthode de cholesky. *Revue d'histoire des mathématiques*, 11(2):205–238, 2005.

[15] Agus Budiyono. Principles of gnss, inertial, and multi-sensor integrated navigation systems. *Industrial Robot: An International Journal*, 2012.

[16] Suman Chakravorty et al. An unscented kalman-particle hybrid filter for space object tracking. *The Journal of the Astronautical Sciences*, 65(1):111–134, 2018.

[17] Zhe Chen et al. Bayesian filtering: From kalman filters to particle filters, and beyond. *Statistics*, 182(1):1–69, 2003.

[18] Kai-Wei Chiang, Guang-Je Tsai, Hone-Jay Chu, and Naser El-Sheimy. Performance enhancement of ins/gnss/refreshed-slam integration for acceptable lane-level navigation accuracy. *IEEE Transactions on Vehicular Technology*, 69(3):2463–2476, 2020.

[19] Richard A Davis, Keh-Shin Lii, and Dimitris N Politis. Remarks on some nonparametric estimates of a density function. In *Selected Works of Murray Rosenblatt*, pages 95–100. Springer, 2011.

[20] Nabil Mohamed Drawil and Otman Basir. Intervehicle-communication-assisted localization. *IEEE transactions on intelligent transportation systems*, 11(3):678–691, 2010.

[21] Vassiliy A Epanechnikov. Non-parametric estimation of a multivariate probability density. *Theory of Probability & Its Applications*, 14(1):153–158, 1969.

[22] Leonhard Euler. *Institutiones calculi integralis*, volume 4. Academia Imperialis Scientiarum, 1794.

[23] Gianluca Falco, Garry A Einicke, John T Malos, and Fabio Dovis. Performance analysis of constrained loosely coupled gps/ins integration solutions. *Sensors*, 12(11):15983–16007, 2012.

[24] Kaiqiang Feng, Jie Li, Xi Zhang, Xiaoming Zhang, Chong Shen, Huiliang Cao, Yanyu Yang, and Jun Liu. An improved strong tracking cubature kalman filter for gps/ins integrated navigation systems. *Sensors*, 18(6):1919, 2018.

[25] Fuad A Ghaleb, Anazida Zainal, Murad A Rassam, and Ajith Abraham. Improved vehicle positioning algorithm using enhanced innovation-based adaptive kalman filter. *Pervasive and Mobile Computing*, 40:139–155, 2017.

[26] Neil J Gordon, David J Salmond, and Adrian FM Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEE Proceedings F-radar and signal processing*, volume 140, pages 107–113. IET, 1993.

[27] Dominique Gruyer, Valentin Magnier, Karima Hamdi, Laurene Claussmann, Olivier Orfila, and Andry Rakotonirainy. Perception, information processing and modeling: Critical stages for autonomous driving applications. *Annual Reviews in Control*, 44:323–341, 2017.

[28] Gaoge Hu, Bingbing Gao, Yongmin Zhong, and Chengfan Gu. Unscented kalman filter with process noise covariance estimation for vehicular ins/gps integration system. *Information Fusion*, 64:194–204, 2020.

[29] Gaoge Hu, Wei Wang, Yongmin Zhong, Bingbing Gao, and Chengfan Gu. A new direct filtering approach to ins/gnss integration. *Aerospace Science and Technology*, 77:755–764, 2018.

[30] Jihua Huang and H-S Tan. A low-order dgps-based vehicle positioning system under urban environment. *IEEE/ASME Transactions on mechatronics*, 11(5):567–575, 2006.

[31] Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics*, pages 492–518. Springer, 1992.

[32] Haitao Jiang, Chuang Shi, Tuan Li, Yitong Dong, Yuhang Li, and Guifei Jing. Low-cost gps/ins integration with accurate measurement modeling using an extended state observer. *GPS Solutions*, 25(1):1–15, 2021.

[33] Simon J Julier and Jeffrey K Uhlmann. New extension of the kalman filter to nonlinear systems. In *Signal processing, sensor fusion, and target recognition VI*, volume 3068, pages 182–193. International Society for Optics and Photonics, 1997.

[34] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

[35] Paul Kaminski, A Bryson, and Stanley Schmidt. Discrete square root filtering: A survey of current techniques. *IEEE Transactions on automatic control*, 16(6):727–736, 1971.

[36] Raaga Kannan and Ronald C Lasky. Autonomous vehicles still decades away: 2019. In *2020 Pan Pacific Microelectronics Symposium (Pan Pacific)*, pages 1–6. IEEE, 2020.

[37] Wei-Wen Kao. Integration of gps and dead-reckoning navigation systems. In *Vehicle Navigation and Information Systems Conference, 1991*, volume 2, pages 635–643. IEEE, 1991.

[38] Rickard Karlsson and Fredrik Gustafsson. The future of automotive localization algorithms: Available, reliable, and scalable localization: Anywhere and anytime. *IEEE signal processing magazine*, 34(2):60–69, 2017.

[39] Chansoo Kim, Sungjin Cho, Myoungho Sunwoo, Paulo Resende, Benazouz Bradaï, and Kichun Jo. Updating point cloud layer of high definition (hd) map based on crowd-sourcing of multiple vehicles installed lidar. *IEEE Access*, 9:8028–8046, 2021.

[40] Sampo Kuutti, Saber Fallah, Konstantinos Katsaros, Mehrdad Dianati, Francis Mccullough, and Alexandros Mouzakitis. A survey of the state-of-the-art localization techniques and their potentials for autonomous vehicle applications. *IEEE Internet of Things Journal*, 5(2):829–846, 2018.

[41] Dengao Li, Xuan Jia, and Jumin Zhao. A novel hybrid fusion algorithm for low-cost gps/ins integrated navigation system during gps outages. *IEEE Access*, 8:53984–53996, 2020.

[42] Tiancheng Li, Miodrag Bolic, and Petar M Djuric. Resampling methods for particle filtering: classification, implementation, and strategies. *IEEE Signal processing magazine*, 32(3):70–86, 2015.

[43] Yong Li, Jinling Wang, Chris Rizos, Peter Mumford, and Weidong Ding. Low-cost tightly coupled gps/ins integration based on a nonlinear kalman filtering design. In *Proceedings of the 2006 National Technical Meeting of The Institute of Navigation*, pages 958–966, 2006.

[44] Ming Lin and Byeongwoo Kim. Discriminative parameter training of the extended particle-aided unscented kalman filter for vehicle localization. *Applied Sciences*, 10(18):6260, 2020.

[45] Ming Lin and Byeongwoo Kim. Extended particle-aided unscented kalman filter based on self-driving car localization. *Applied Sciences*, 10(15):5045, 2020.

[46] Ming Lin, Jaewoo Yoon, and Byeongwoo Kim. Self-driving car location estimation based on a particle-aided unscented kalman filter. *Sensors*, 20(9):2544, 2020.

[47] Jiageng Liu and Ge Guo. Vehicle localization during gps outages with extended kalman filter and deep learning. *IEEE Transactions on Instrumentation and Measurement*, 70:1–10, 2021.

[48] Kai Liu, Hock Beng Lim, Emilio Frazzoli, Houling Ji, and Victor CS Lee. Improving positioning accuracy using gps pseudorange measurements for cooperative vehicular localization. *IEEE Transactions on Vehicular Technology*, 63(6):2544–2556, 2013.

[49] Yahui Liu, Xiaoqian Fan, Chen Lv, Jian Wu, Liang Li, and Dawei Ding. An innovative information fusion method with adaptive kalman filter for integrated ins/gps navigation of autonomous vehicles. *Mechanical Systems and Signal Processing*, 100:605–616, 2018.

[50] Yong Luo, Ravindra Babu, Wen-qi Wu, and Xiao-feng He. Double-filter model with modified kalman filter for baseband signal pre-processing with application to ultra-tight gps/ins integration. *GPS solutions*, 16(4):463–476, 2012.

[51] Roger Mansuy and Marc Yor. *Aspects of Brownian motion*. Springer Science & Business Media, 2008.

[52] Frank J Massey Jr. The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253):68–78, 1951.

[53] Arash Olia, Hossam Abdelgawad, Baher Abdulhai, and Saiedeh N Razavi. Assessing the potential impacts of connected vehicles: mobility, environmental, and safety perspectives. *Journal of Intelligent Transportation Systems*, 20(3):229–243, 2016.

[54] Celestino Ordonez, J Martínez, JR Rodríguez-Pérez, and Andria Reyes. Detection of outliers in gps measurements by using functional-data analysis. *Journal of Surveying Engineering*, 137(4):150–155, 2011.

[55] Ryan Parker and Shahrokh Valaee. Vehicular node localization using received-signal-strength indicator. *IEEE Transactions on Vehicular Technology*, 56(6):3371–3380, 2007.

[56] Emanuel Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962.

[57] Matti Raitoharju, Robert Piché, and Henri Nurminen. A systematic approach for kalman-type filtering with non-gaussian noises. In *2016 19th International Conference on Information Fusion (FUSION)*, pages 1853–1858. IEEE, 2016.

[58] Alvaro Santamaría-Gómez, Marie-Noëlle Bouin, Xavier Collilieux, and Guy Wöppelmann. Correlated errors in gps position time series: Implications for velocity estimates. *Journal of Geophysical Research: Solid Earth*, 116(B1), 2011.

[59] Abraham Savitzky and Marcel JE Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8):1627–1639, 1964.

[60] Robin Schubert, Christian Adam, Marcus Obst, Norman Mattern, Veit Leonhardt, and Gerd Wanielik. Empirical evaluation of vehicular models for ego motion estimation. In *2011 IEEE intelligent vehicles symposium (IV)*, pages 534–539. IEEE, 2011.

[61] Christopher Schwarzlmüller, Fadi Al Machot, Alireza Fasih, and Kyandoghere Kyamakya. Adaptive contrast enhancement involving cnn-based processing for foggy weather conditions & non-uniform lighting conditions. In *Proceedings of the Joint INDS'11 & ISTET'11*, pages 1–10. IEEE, 2011.

[62] Joga Dharma Setiawan, Mochamad Safarudin, and Amrik Singh. Modeling, simulation and validation of 14 dof full vehicle model. In *International Conference on Instrumentation, Communication, Information Technology, and Biomedical Engineering 2009*, pages 1–6. IEEE, 2009.

[63] Rathin Chandra Shit. Precise localization for achieving next-generation autonomous navigation: State-of-the-art, taxonomy and future prospects. *Computer Communications*, 2020.

[64] Gerald L Smith, Stanley F Schmidt, and Leonard A McGee. *Application of statistical filter theory to the optimal estimation of position and velocity on board a circumlunar vehicle*. National Aeronautics and Space Administration, 1962.

[65] Adriano Solimeno. Low-cost ins/gps data fusion with extended kalman filter for airborne applications. *Masters of Science, Universidade Technica de Lisboa*, 2007.

[66] Rui Song, Xiyuan Chen, Yongchun Fang, and Haoqian Huang. Integrated navigation of gps/ins based on fusion of recursive maximum likelihood imm and square-root cubature kalman filter. *ISA transactions*, 105:387–395, 2020.

[67] Ines V Stelzer, Julian Kager, and Christoph Herwig. Comparison of particle filter and extended kalman filter algorithms for monitoring of bioprocesses. In *Computer Aided Chemical Engineering*, volume 40, pages 1483–1488. Elsevier, 2017.

[68] Guillaume Thenaisie, Cheol-Heui Park, and Sang-Gug Lee. A real-time entropy estimation algorithm for lithium batteries based on a combination of kalman filter and nonlinear observer. *IEEE Transactions on Industrial Electronics*, 67(9):8034–8043, 2019.

[69] David Titterton, John L Weston, and John Weston. *Strapdown inertial navigation technology*, volume 17. IET, 2004.

[70] Nicolae Tudoroiu, Kash Khorasani, Mohammed Zaheeruddin, Eshan Sobhani-Tehrani, Dumitru Burdescu, and Elena-Roxana Tudoroiu. Application of the unscented kalman filter (ukf) estimation techniques for fault detection diagnosis and isolation (fddi) in attitude control (ac) and heating ventilation air conditioning (hvac) systems. *Kalman Filter: Recent Advances and Applications*, page 461, 2009.

[71] Rudolph Van Der Merwe, Arnaud Doucet, Nando De Freitas, and Eric Wan. The unscented particle filter. *Advances in neural information processing systems*, 13:584–590, 2000.

[72] Michel Verhaegen and Vincent Verdult. *Filtering and system identification: a least squares approach*. Cambridge university press, 2007.

[73] Eric A Wan and Rudolph Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*, pages 153–158. Ieee, 2000.

[74] Fan Wu, Haiyong Luo, Hongwei Jia, Fang Zhao, Yimin Xiao, and Xile Gao. Predicting the noise covariance with a multitask learning model for kalman filter-based gnss/ins integrated navigation. *IEEE Transactions on Instrumentation and Measurement*, 70:1–13, 2020.

[75] Hailiang Xiong, Zhenzhen Mai, Juan Tang, and Fen He. Robust gps/ins/dvl navigation and positioning method using adaptive federated strong tracking filter based on weighted least square principle. *IEEE Access*, 7:26168–26178, 2019.

[76] Qimin Xu, Xu Li, and Ching-Yao Chan. Enhancing localization accuracy of mems-ins/gps/in-vehicle sensors integration during gps outages. *IEEE Transactions on Instrumentation and Measurement*, 67(8):1966–1978, 2018.

[77] Jafar Zarei and Ehsan Shokri. Convergence analysis of non-linear filtering based on cubature kalman filter. *IET Science, Measurement & Technology*, 9(3):294–305, 2015.

[78] Jinghe Zhang. *Uncertainty-driven adaptive estimation with applications in electrical power systems*. PhD thesis, The University of North Carolina at Chapel Hill, 2013.

[79] Yu Zhang, Chong Shen, Jun Tang, and Jun Liu. Hybrid algorithm based on mdf-ckf and rf for gps/ins system during gps outages (april 2018). *IEEE Access*, 6:35343–35354, 2018.