Extension of an EKF-Based Localization Filter by Stochastic Cloning

Bachelor's Thesis of

Anton Schirg

at the

Fraunhofer Institute of Optronics, System Technologies
and Image Exploitation IOSB

Reviewer:	Prof. Dr.-Ing. habil. Jürgen Beyerer
Second reviewer:	Prof. Dr. rer. nat. Hartwig Steusloff
Advisors:	Dipl.-Ing. Thomas Emter, Dr.-Ing. Philipp Woock

01. October 2018 – 31. January 2019

I declare that I have developed and written the enclosed thesis completely by myself, and have not used sources or means without declaration in the text.

Karlsruhe, 31.01.2019

. . . .
(Anton Schirg)

Abstract

The Extended Kalman Filter is commonly used on mobile robots, as multiple sensors can be combined to provide an accurate and reliable localization. However in the general Kalman filter framework relative measurements relating states at different times can not be used. Stochastic cloning is a state augmentation method that allows such measurements to be integrated. The currently often used implicit form of stochastic cloning does not allow the fusion of other measurements while a relative measurement is in progress. This thesis advocates usage of the explicit form which allows the usage of relative sensors combined with absolute sensors with measurements in arbitrary order and compares results of explicit stochastic cloning with approximate methods.

Zusammenfassung

Der erweiterte Kalman-Filter wird häufig bei mobilen Robotern eingesetzt, da mehrere Sensoren kombiniert werden können, um eine genaue und zuverlässige Lokalisierung zu erhalten. Im allgemeinen Kalman-Filter Framework können jedoch relative Messungen, die von Zuständen an unterschiedlichen Zeitpunkten abhängig sind, nicht verwendet werden. Stochastic Cloning ist ein Zustandserweiterungsverfahren, das die Integration solcher Messungen ermöglicht. Die derzeit häufig verwendete implizite Form des Stochastic Clonings erlaubt es nicht, andere Messungen zu fusionieren, während eine relative Messung durchgeführt wird. Diese Arbeit befürwortet die Verwendung der expliziten Form, die die Verwendung von relativen Sensoren in Kombination mit absoluten Sensoren mit Messungen in beliebiger Reihenfolge ermöglicht. Die Ergebnisse des expliziten Stochastic Clonings werden mit ungefähren Methoden verglichen.

Contents

Table of Symbols

Symbol	Description
$\mathbf{x}$	State
$\hat{\mathbf{x}}$	Estimated state
$\check{\mathbf{x}}$	Augmented state
$\bar{\mathbf{z}}$	Measured value
$\mathbf{x}$	State
$\mathbf{P}$	State covariance
f	System model
$\mathbf{F}$	System model Jacobian
$\mathbf{u}$	Control input
$\mathbf{w}$	System noise
$\mathbf{G}$	System noise Jacobian
$\mathbf{Q}$	System noise covariance
h	Measurement model
$\mathbf{H}$	Measurement model Jacobian
$\mathbf{v}$	Measurement noise
$\mathbf{V}$	Measurement noise Jacobian
$\mathbf{R}$	Measurement noise covariance
$\mathbf{K}$	Kalman gain
$\hat{\mathbf{x}}_{k+m\|k}$	estimate of the state at time $k + m$ using measurements up to time k
$\mathbf{p}_b^n$	position of the body frame relative to the navigation frame
$\mathbf{q}_b^n$	orientation of the body frame relative to the navigation frame
$\mathbf{v}_{eb}^n$	velocity of the body frame relative to the earth frame given in the navigation frame
$(\cdots)_x$	x-component of a vector
$[\cdots]_\times$	skew symmetric operator, i. e., $[\mathbf{a}]_\times \mathbf{b} = \mathbf{a} \times \mathbf{b}$ where $\times$ is the cross product

1 Introduction

The Extended Kalman Filter (EKF) is commonly used in mobile robot localization. Multiple sensors (e.g. GNSS, IMU, Compass) can be combined to provide a more accurate and reliable pose.

The Kalman filter uses a mathematical model of the hidden state of the system and the dependencies of the sensor measurements on this state to estimate the state of the system using incoming measurements.

But in the general Kalman filter framework, because of the Markovian assumption [9, p. 51], measurements in the update step may only depend on the state at their respective timesteps. This does not hold true for relative sensors (e.g. odometry, laser scan matches) which measure a displacement between two timesteps.

One way to integrate these anyway is to convert them to a pseudo-velocity measurement [8]. But this may require changes to the system model, i. e., the velocity must be included in the state. Also it is not entirely accurate, because the converted velocity is an average velocity but is handled like an instantaneous velocity.

Stochastic cloning (SC) as introduced by Roumeliotis et al. [8] allows such sensors to be integrated into the state as relative measurements by cloning the state at the start of the measurement into a larger augmented state vector and state covariance matrix. This way the measurement model of the relative sensor has access to the states both at the start and at the end of the measurement as well as the covariances relating the two. Using this information the predicted measurement and its covariance can be calculated. The augmented state covariance matrix is only stored implicitly.

This implicit form of stochastic cloning does not allow the fusion of other measurements while a relative measurement is in progress.

This thesis uses a generalization which will be called explicit stochastic cloning. This form allows the usage of relative sensors combined with absolute sensors with measurements in arbitrary order.

Figure 1.1 shows examples of the dependencies between states $\mathbf{x}_i$ and measurements $\mathbf{z}_i$ that can be handled by the different methods.

The explicit SC-EKF is first demonstrated in a simulation using a simple system model for better understandability and compared to the pseudo-velocity method.

Later this approach is used to integrate odometry measurements into an existing EKF based localization filter used on an autonomous vehicle. Previously only the forward velocity

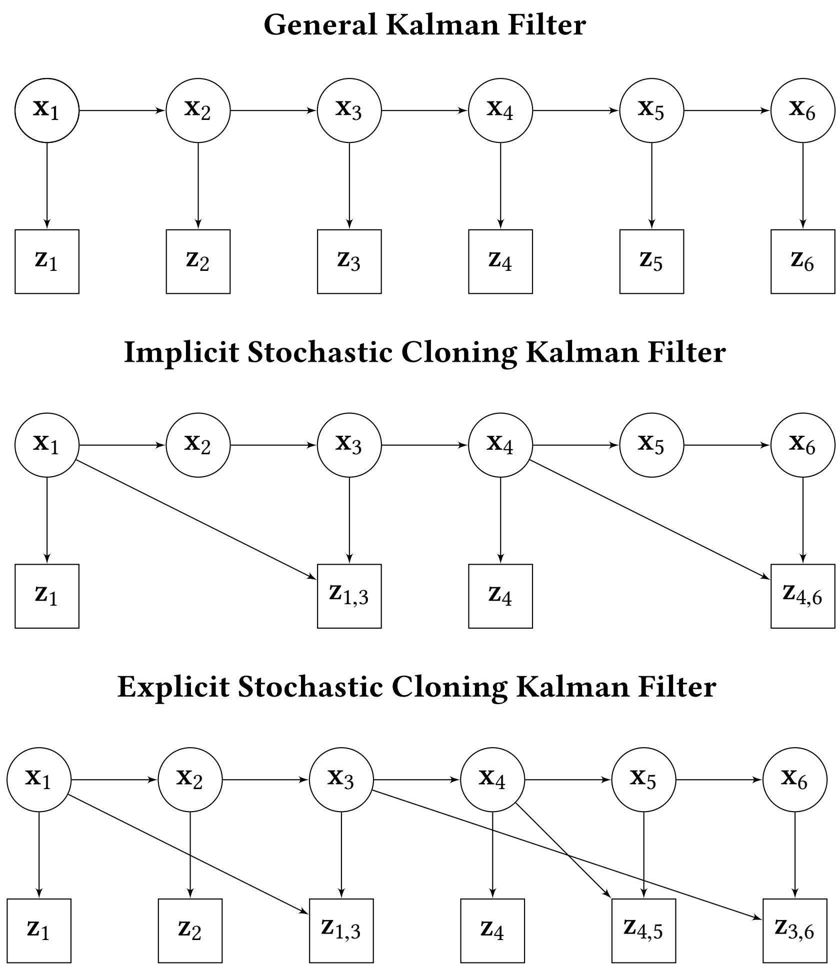


Figure 1.1: Comparison of the Bayesian networks handled by a general Kalman filter and the stochastic cloning Kalman filters

could be included, as angular velocity is not part of the state vector, so an improvement will be expected. The results obtained using the SC-EKF are compared with those of the existing conventional EKF.

1.1 Previous / relevant work

The first mention of stochastic cloning was probably in [8]. This method is used to process relative measurements in [5], [3]. The idea of explicit stochastic cloning is mentioned in [7] as a possible extension of this method, however no work is known where this was actually implemented.

2 The Stochastic Cloning Kalman Filter

A Kalman filter [4] estimates the probability distribution of the (unobservable) state of a discrete linear system given a model of the system and measurements that linearly depend on the state of the system. The probability distribution is modeled as a multivariate Gaussian distribution whose mean and covariance is propagated by the filter.

As a filter, in contrast to a smoother, the estimate is calculated using information from past measurements only. The Kalman filter is the optimal linear unbiased filter regarding minimum error variance for a linear system disturbed by white noise [6, p. 7].

The extended Kalman filter (EKF) is a generalization of the Kalman filter for non-linear systems. It essentially works by linearizing the system around the current estimated state at every timestep and applying a Kalman filter to the linearized system. The extended Kalman filter is in general non-optimal, but works well for nearly linear systems. It is relatively efficient even for systems with a state space of high dimensionality, in contrast to, e. g., a particle filter.

For robot localization the state to be estimated is the pose of the robot consisting of position and orientation relative to some reference frame. Estimates of other quantities needed to describe the system may be included in the state vector, e. g., the velocity of the robot, sensor biases, etc.

For a more detailed introduction of the Kalman filter see, e. g., [2]. A good overview of the concepts is also given in [6, Ch. 1.3].

2.1 EKF Equations

This section summarizes the EKF equations similar to how they are given in [2].

The Kalman filter is usually split into two alternating steps: The prediction step propagates the state to the next timestep using the system model and the update step fuses a measurement into the state (Fig. 2.1). It is valid to have multiple or no measurements during one cycle. For the sake of a more readable notation we will however assume that there is exactly one update for every prediction.

Figure 2.2 gives a high level overview of the extended Kalman filter equations. The dots and ellipses represent mean and covariance of a Gaussian distribution. The straight bold arrows represent vectors and the thin curved arrows represent mappings. The system model f is used to predict the state at timestep k using the state estimate at timestep

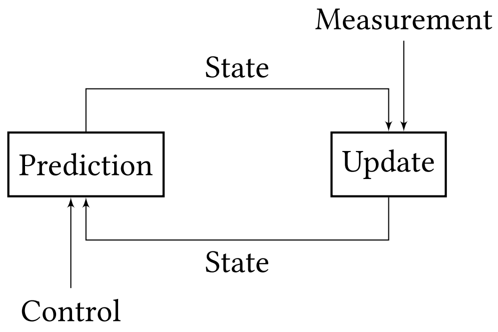


Figure 2.1: Predict - Update cycle of the Kalman filter

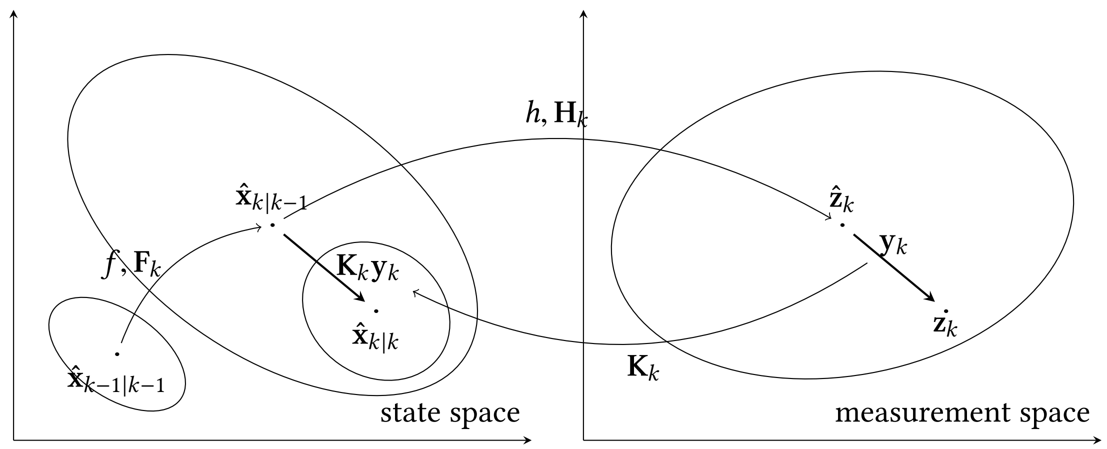


Figure 2.2: Visualization of the Kalman filter equations in state and measurement space

$k-1$. The system model is linearized and the resulting Jacobian $\mathbf{F}$ is used to propagate the covariance.

To apply a measurement first the estimated measurement $\hat{\mathbf{z}}$ is calculated from the state using the measurement model h. Again the measurement model is linearized and the Jacobian $\mathbf{H}$ is used to calculate the covariance of $\hat{\mathbf{z}}$. Then the error of the estimated measurement, called the residual $\mathbf{y}$, is mapped back to the state space using the Kalman gain $\mathbf{K}$ and applied to the state estimate.

2.1.1 Prediction

The system model is f given as

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_k, \mathbf{w}_k) \tag{2.1}$$

where $\mathbf{x}_{k-1}$ is the state at the previous timestep, $\mathbf{u}_k$ is the control input and $\mathbf{w}_k$ is the system noise.

Estimation of the state at time k using the estimate at time $k-1$ is performed using the system model by assuming the noise to be zero (as it is zero-mean):

$$\hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k, \mathbf{0}). \tag{2.2}$$

For estimation of the state covariance the system model is linearized with respect to the state and the noise.

$$\begin{aligned}
\mathbf{F}_k &= \left.\frac{\partial f(\mathbf{x}, \mathbf{u}, \mathbf{w})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}=\mathbf{u}_k, \mathbf{w}=\mathbf{0}} \\
\mathbf{G}_k &= \left.\frac{\partial f(\mathbf{x}, \mathbf{u}, \mathbf{w})}{\partial \mathbf{w}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}=\mathbf{u}_k, \mathbf{w}=\mathbf{0}} \\
f(\mathbf{x}, \mathbf{w}) &\approx f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k, \mathbf{0}) + \mathbf{F}_k(\mathbf{x} - \hat{\mathbf{x}}_{k-1|k-1}) + \mathbf{G}_k \mathbf{w}
\end{aligned}$$

The covariance is then propagated using this linearization:

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{G}_k \mathbf{Q}_k \mathbf{G}_k^T. \tag{2.3}$$

2.1.2 Update

The measurement model h is given as

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \tag{2.4}$$

where $\mathbf{x}_k$ is the current state and $\mathbf{v}_k$ is the measurement noise.

The following steps are performed to integrate a measurement into the estimate: The measurement model is used to calculate the residual $\mathbf{y}_k$:

$$\mathbf{y}_k = \mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1}). \tag{2.5}$$

For the following equations the measurement model is linearized with respect to the state and the noise.

$$\begin{aligned}
\mathbf{H}_k &= \left.\frac{\partial h(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}} \\
h(\mathbf{x}) &\approx h(\hat{\mathbf{x}}_{k|k-1}) + \mathbf{H}_k(\mathbf{x} - \hat{\mathbf{x}}_{k-1|k-1})
\end{aligned}$$

The optimal Kalman gain $\mathbf{K}_k$ is computed from the estimated covariance, the measurement Jacobian and the measurement covariance:

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k)^{-1}. \tag{2.6}$$

As the name suggests, the Kalman gain is used to weight the residual into the state estimate:

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \mathbf{y}_k. \tag{2.7}$$

The state covariance is propagated as following

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k|k-1}(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T. \tag{2.8}$$

Equation 2.8 is called the Joseph form of the Kalman update.

For the optimal Kalman gain (2.6) it can be simplified to

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k|k-1}. \tag{2.9}$$

The Joseph form is however numerically more robust [11, p. 132] and can also be used with a non-optimal Kalman gain. For a derivation of both forms see [10].

2.2 Implicit Stochastic Cloning

As shown in Equation 2.4, a measurement in the conventional EKF depends only on the state at the current time. This is not the case for relative measurements, e. g., laser scan matches which measure a pose displacement between two time steps. Roumeliotis at al. [8] introduce stochastic cloning as a method to handle such measurements.

The state the relative measurement relates to is remembered by including it in an augmented state:

$$\breve{\mathbf{x}}_k = \begin{pmatrix} \hat{\mathbf{x}}_{k,s} \\ \hat{\mathbf{x}}_k \end{pmatrix} \tag{2.10}$$

where $\hat{\mathbf{x}}_{k,s}$ is the estimate of the cloned state and $\hat{\mathbf{x}}_k$ that of the current "evolving" state. This augmented state is estimated by the filter. Consequently the filter also estimates the covariance relating the two states. This way the measurement model has access to all required quantities.

2.2.1 Cloning

Instead of storing the full state and covariance matrix, the cloned state and the covariance of the cloned state are stored separately. Initially they are set equal to their evolving counterparts. Additionally the accumulated state transfer matrix $\mathcal{F}$, which relates the cloned and the evolving state, is stored. It is initialized with the identity matrix.

2.2.2 Prediction

As only the evolving state shall be affected by the prediction, $\breve{\mathbf{F}}$ and $\breve{\mathbf{G}}$ are chosen as follows [8]:

$$\begin{pmatrix} \mathbf{x_s} \\ \mathbf{x_e} \end{pmatrix}_{k|k-1} = \underbrace{\begin{pmatrix} \mathbf{I} & 0 \\ 0 & \mathbf{F}_k \end{pmatrix}}_{=: \breve{\mathbf{F}}_k} \begin{pmatrix} \mathbf{x_s} \\ \mathbf{x_e} \end{pmatrix}_{k-1|k-1} + \underbrace{\begin{pmatrix} 0 \\ \mathbf{G}_k \end{pmatrix}}_{=: \breve{\mathbf{G}}_k} \mathbf{w}. \tag{2.11}$$

Substituting $\breve{\mathbf{F}}_k$ and $\breve{\mathbf{G}}_k$ into Eq. 2.3 results in

$$\breve{\mathbf{P}}_{k|k-1} = \begin{pmatrix} \mathbf{P}_{k-1|k-1} & \mathbf{P}_{k-1|k-1}\mathbf{F}_k^T \\ \mathbf{F}_k\mathbf{P}_{k-1|k-1} & \mathbf{F}_k\mathbf{P}_{k-1|k-1}\mathbf{F}_k^T + \mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^T \end{pmatrix}. \tag{2.12}$$

Applying this equation for $m+1$ steps and substituting $\mathbf{F}_{k+m} \cdots \mathbf{F}_{k+1}\mathbf{F}_k =: \mathcal{F}$ results in

$$\breve{\mathbf{P}}_{k+m|k-1} = \begin{pmatrix} \mathbf{P}_{k-1|k-1} & \mathbf{P}_{k-1|k-1}\mathcal{F}^T \\ \mathcal{F}\mathbf{P}_{k-1|k-1} & \mathbf{P}_{k+m|k-1} \end{pmatrix}. \tag{2.13}$$

2.2.3 Update

Equation 2.13 can be used to obtain the full covariance matrix from the static and evolving covariance matrices and the accumulated state transfer matrix. It can then be used to execute the update using the conventional EKF equations.

The estimate of the cloned state and its covariances are only needed to update the evolving state and are discarded once the relative measurement is done. This means only the evolving part of the state and covariance need to be updated. By exploiting this fact the update can be brought into a more specific form depending on the cloned covariance, the evolving covariance and the accumulated state transfer matrix [8]. This way the full covariance matrix does not have to be constructed.

However this method is only usable if no other updates need to be processed between the cloning of a state and the update using this cloned state, i. e., if the relative sensor is the only sensor or if the sensors are synchronized in a way that no update has to happen while a relative measurement is in progress.

2.3 Explicit Stochastic Cloning

Being able to use only one sensor is quite limiting. In this thesis a more general form of stochastic cloning is used where this limitation does not apply. Instead of splitting the covariance into the form shown in Eq. 2.13 the full augmented state and covariance are used in the general EKF equations.

This also allows more than one clone to be used at the same time. Arguably it is also more understandable as the equations are the same as those of the regular EKF and not hidden behind calculations specific to the case of exactly one relative sensor.

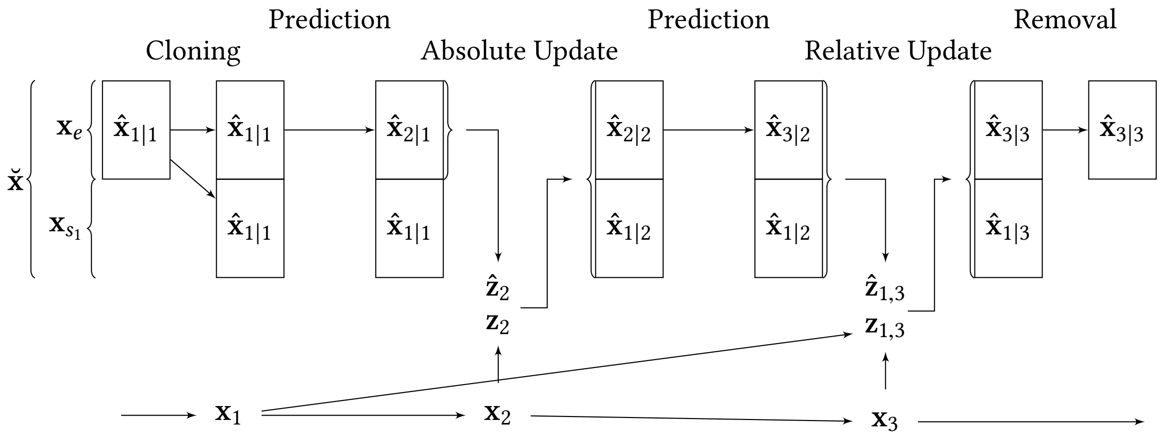


Figure 2.3: The steps of handling a relative measurement using stochastic cloning

The state the relative measurement relates to is cloned and included in a larger augmented state. Consequently the cloned state, its covariance and the covariances relating the cloned state to the other states are also continuously estimated by the filter. The cloned state is not affected by the predict step as it always represents the same time step but later absolute measurements may change the estimate of the cloned state. Once the relative measurement has been processed the clone is removed or replaced by a new clone. Figure 2.3 shows this sequence.

The state clones are also called static states $\mathbf{x}_{s_i}$ and the state that is changed by the prediction step is called evolving state $\mathbf{x}_e$. The whole state vector containing state clones and the evolving state is called the augmented state $\breve{\mathbf{x}}$.

2.3.1 Initialization

Initially the augmented state vector contains only the evolving state.

$$\begin{aligned} \breve{\mathbf{x}}_0 &= \mathbf{x}_{e,0} \\ \breve{\mathbf{P}}_0 &= \mathbf{P}_{ee,0} \end{aligned} \tag{2.14}$$

2.3.2 Cloning

At the start of a relative measurement the current evolving state needs to be cloned to a new static state.

A new clone of the evolving state is added to the estimated state vector. At this point the new clone is equal to the evolving state and all covariances involving the new clone are set as if it was the evolving state:

$$\breve{\mathbf{x}} = \begin{pmatrix} \mathbf{x}_{s_1} \\ \vdots \\ \mathbf{x}_{s_n} \\ \mathbf{x}_e \end{pmatrix} \qquad \breve{\mathbf{P}} = \begin{pmatrix} \mathbf{P}_{s_1 s_1} & \cdots & \mathbf{P}_{s_1 s_n} & \mathbf{P}_{s_1 e} \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{P}_{s_n s_1} & \cdots & \mathbf{P}_{s_n s_n} & \mathbf{P}_{s_n e} \\ \mathbf{P}_{e s_1} & \cdots & \mathbf{P}_{e s_n} & \mathbf{P}_{ee} \end{pmatrix}$$

becomes

$$\breve{\mathbf{x}} = \begin{pmatrix} \mathbf{x}_{s_1} \\ \vdots \\ \mathbf{x}_{s_n} \\ \mathbf{x}_{s_{n+1}} = \mathbf{x}_e \\ \mathbf{x}_e \end{pmatrix} \quad \breve{\mathbf{P}} = \begin{pmatrix} \mathbf{P}_{s_1 s_1} & \cdots & \mathbf{P}_{s_1 s_n} & \mathbf{P}_{s_1 e} & \mathbf{P}_{s_1 e} \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ \mathbf{P}_{s_n s_1} & \cdots & \mathbf{P}_{s_n s_n} & \mathbf{P}_{s_n e} & \mathbf{P}_{s_n e} \\ \mathbf{P}_{e s_1} & \cdots & \mathbf{P}_{e s_n} & \mathbf{P}_{ee} & \mathbf{P}_{ee} \\ \mathbf{P}_{e s_1} & \cdots & \mathbf{P}_{e s_n} & \mathbf{P}_{ee} & \mathbf{P}_{ee} \end{pmatrix}. \tag{2.15}$$

2.3.3 Prediction

As only the evolving state shall be affected by the prediction step, $\breve{\mathbf{F}}$ and $\breve{\mathbf{G}}$ are chosen as follows [8]:

$$\begin{pmatrix} \mathbf{x}_{s_1} \\ \vdots \\ \mathbf{x}_{s_n} \\ \mathbf{x}_e \end{pmatrix}_{k|k-1} = \underbrace{\begin{pmatrix} \mathbf{I} & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & \mathbf{I} & 0 \\ 0 & \cdots & 0 & \mathbf{F}_k \end{pmatrix}}_{=:\,\breve{\mathbf{F}}_k} \begin{pmatrix} \mathbf{x}_{s_1} \\ \vdots \\ \mathbf{x}_{s_n} \\ \mathbf{x}_e \end{pmatrix}_{k-1|k-1} + \underbrace{\begin{pmatrix} 0 \\ \vdots \\ 0 \\ \mathbf{G}_k \end{pmatrix}}_{=:\,\breve{\mathbf{G}}_k} \mathbf{w}. \tag{2.16}$$

The covariance is then propagated correspondingly by substituting the augmented matrices into Eq. 2.3:

$$\breve{\mathbf{P}}_{k|k-1} = \breve{\mathbf{F}}_k \breve{\mathbf{P}}_{k-1|k-1} \breve{\mathbf{F}}_k^T + \breve{\mathbf{G}}_k \breve{\mathbf{Q}}_k \breve{\mathbf{G}}_k^T. \tag{2.17}$$

2.3.4 Absolute Sensor Update

The measurement Jacobian for the update of the augmented state $\breve{\mathbf{H}}$ for a given Jacobian $\mathbf{H}$ of an absolute sensor is

$$\breve{\mathbf{H}} = \begin{pmatrix} 0 & \cdots & 0 & \mathbf{H} \end{pmatrix} \tag{2.18}$$

since the absolute measurement only depends on the evolving state. Using $\breve{\mathbf{H}}$ the regular EKF update equations 2.6 to 2.8 can be used on the augmented state.

It is important to note that the absolute update affects not only the evolving state but also the cloned static states. Directly after cloning a state the $\breve{\mathbf{P}}$ matrix is the same in all four blocks concerning the cloned state, which leads to the Kalman gain being the same for the static and evolving part. This makes sense as directly after a clone the evolving and the cloned state represent the same timestep and should thus be updated the same. After some time the static and evolving states will be decorrelated by the prediction step and absolute measurements will affect the evolving state more than the static state.

The handling of absolute updates is very similar to the handling of updates in a fixed point smoother [1, p. 171].

2.3.5 Relative Sensor Update

When a relative sensor measurement is finished it is applied to the state estimate. In contrast to the absolute update the measurement model has access to the cloned state too.

$$\breve{\mathbf{z}}_k = h(\mathbf{x}_{s_i,k}, \mathbf{x}_{e,k}, \mathbf{v}_k) \tag{2.19}$$

The measurement model is linearized with respect to both the static and the evolving state leading to the measurement Jacobians $\mathbf{H}_{s_i}$ and $\mathbf{H}_e$ which are assembled into the full measurement Jacobian $\breve{\mathbf{H}}$.

$$\breve{\mathbf{H}} = \begin{pmatrix} 0 & \cdots & 0 & \mathbf{H}_{s_i} & 0 & \cdots & 0 & \mathbf{H}_e \end{pmatrix} \tag{2.20}$$

Using $\breve{\mathbf{z}}_k$ and $\breve{\mathbf{H}}$ the regular EKF update equations can be applied.

At the end of the relative sensor update the cloned state is not needed anymore and can be removed.

2.3.6 Clone removal

When a clone is not needed anymore it can be removed by removing the corresponding static state from the state vector and removing all rows and columns concerning the static state from the state covariance matrix. I.e., when removing clone i, $\mathbf{x}_{s_i}$ is removed from the state vector and $\mathbf{P}_{s_i s_1}$ through $\mathbf{P}_{s_i s_e}$ and $\mathbf{P}_{s_1 s_i}$ through $\mathbf{P}_{s_e s_i}$ are removed from the state covariance matrix.

3 2D Simulation

To demonstrate the SC-EKF in a simple and understandable environment where ground truth is available, a 2D simulation was implemented. In this simulation the SC-EKF is compared to a regular EKF which incorporates the relative measurements as velocities.

For simplicity and to allow fusion of relative measurements as velocities, a constant velocity system model was chosen. The state vector thus looks like this:

$$\mathbf{x} = (\underbrace{\mathbf{p}_x \quad \mathbf{p}_y \quad \mathbf{p}_\theta}_{\text{position}} \quad \underbrace{\mathbf{v}_x \quad \mathbf{v}_y \quad \mathbf{v}_\theta}_{\text{velocity}})^T . \tag{3.1}$$

The position is given in world coordinates and the velocity is given in robot body coordinates.

The length of one timestep τ is set to 1 s. The simulation runs for $N = 500$ timesteps.

The system model is defined as such:

$$f(\mathbf{x}, \mathbf{w}) = \begin{pmatrix} \mathbf{p}_x + \cos(\mathbf{p}_\theta + \mathbf{v}_\theta T)\mathbf{v}_x T - \sin(\mathbf{p}_\theta + \mathbf{v}_\theta T)\mathbf{v}_y T + \mathbf{w}_0 \\ \mathbf{p}_y + \sin(\mathbf{p}_\theta + \mathbf{v}_\theta T)\mathbf{v}_x T + \cos(\mathbf{p}_\theta + \mathbf{v}_\theta T)\mathbf{v}_y T + \mathbf{w}_1 \\ \mathbf{p}_\theta + \mathbf{v}_\theta T + \mathbf{w}_2 \\ \mathbf{v}_x \\ \mathbf{v}_y \\ \mathbf{v}_\theta \end{pmatrix} . \tag{3.2}$$

First the rotational velocity is applied to the orientation of the robot, then the linear velocity is applied to the position.

For propagating the real state the velocities are set to

$$\begin{aligned} \mathbf{v}_x &= 1 \\ \mathbf{v}_y &= 0 \\ \mathbf{v}_\theta &= \begin{cases} \sin(2\pi/N), & \text{if } k < N/2 \\ -\sin(2\pi/N), & \text{otherwise} \end{cases} \end{aligned} \tag{3.3}$$

leading to an s-shaped trajectory. This input is not known to the filters. The filters are however initialized with the correct position and velocity.

The simulation contains two sensors. In the following paragraphs the measurement models of these sensors are given. The measurement noise is modeled to be additive as defined in Equation 2.4.

The first sensor measures the absolute value of θ (similar to a compass) every timestep using the following measurement model:

$$h(\mathbf{x}) = \begin{pmatrix}\theta\end{pmatrix} \tag{3.4}$$

The second sensor measures the robot's pose relative to an earlier timestep (similar to laser scan matching) every 10th timestep using the following measurement model:

$$h(\begin{pmatrix}\mathbf{x}_s \\ \mathbf{x}_e\end{pmatrix}) = \begin{pmatrix}\cos(-\mathbf{p}_{\theta,s})(\mathbf{p}_{x,e} - \mathbf{p}_{x,s}) - \sin(-\mathbf{p}_{\theta,s})(\mathbf{p}_{y,e} - \mathbf{p}_{y,s}) \\ \sin(-\mathbf{p}_{\theta,s})(\mathbf{p}_{x,e} - \mathbf{p}_{x,s}) + \cos(-\mathbf{p}_{\theta,s})(\mathbf{p}_{y,e} - \mathbf{p}_{y,s}) \\ \mathbf{p}_{\theta,e} - \mathbf{p}_{\theta,s}\end{pmatrix} \tag{3.5}$$

For the conventional EKF the measurement is converted to a velocity and the following measurement model is used for the update step:

$$h(\mathbf{x}) = \begin{pmatrix}\mathbf{v}_x & \mathbf{v}_y & \mathbf{v}_\theta\end{pmatrix}^T . \tag{3.6}$$

The conversion can be done by separately converting each component to a velocity:

$$\begin{pmatrix}\mathbf{v}_x & \mathbf{v}_y & \mathbf{v}_\theta\end{pmatrix}^T = \frac{\begin{pmatrix}\Delta x & \Delta y & \Delta\theta\end{pmatrix}^T}{10\tau}. \tag{3.7}$$

Likewise the variance of each component is converted by dividing it by $(10\tau)^2$:

$$\begin{aligned}\mathrm{Var}(\mathbf{v}_x) &= \left(\frac{1}{10\tau}\right)^2 \mathrm{Var}(\Delta x) \\ \mathrm{Var}(\mathbf{v}_y) &= \left(\frac{1}{10\tau}\right)^2 \mathrm{Var}(\Delta y) \\ \mathrm{Var}(\mathbf{v}_\theta) &= \left(\frac{1}{10\tau}\right)^2 \mathrm{Var}(\Delta\theta).\end{aligned} \tag{3.8}$$

For comparison a conversion is used which uses the knowledge that the robot does not move sideways and approximates the driven distance as a straight line:

$$\begin{pmatrix}\mathbf{v}_x & \mathbf{v}_y & \mathbf{v}_\theta\end{pmatrix}^T = \frac{\begin{pmatrix}\sqrt{\Delta x^2 + \Delta y^2} & 0 & \Delta\theta\end{pmatrix}^T}{10\tau}. \tag{3.9}$$

This would of course not work for an omnidirectional robot that is able to move sideways. Assuming the variances of Δx and Δy are the same they can be transformed as following:

$$\begin{aligned}\mathrm{Var}(\mathbf{v}_x) &= \left(\frac{\sqrt{2}}{10\tau}\right)^2 \mathrm{Var}(\Delta x) \\ \mathrm{Var}(\mathbf{v}_y) &= 0 \\ \mathrm{Var}(\mathbf{v}_\theta) &= \left(\frac{1}{10\tau}\right)^2 \mathrm{Var}(\Delta\theta).\end{aligned} \tag{3.10}$$

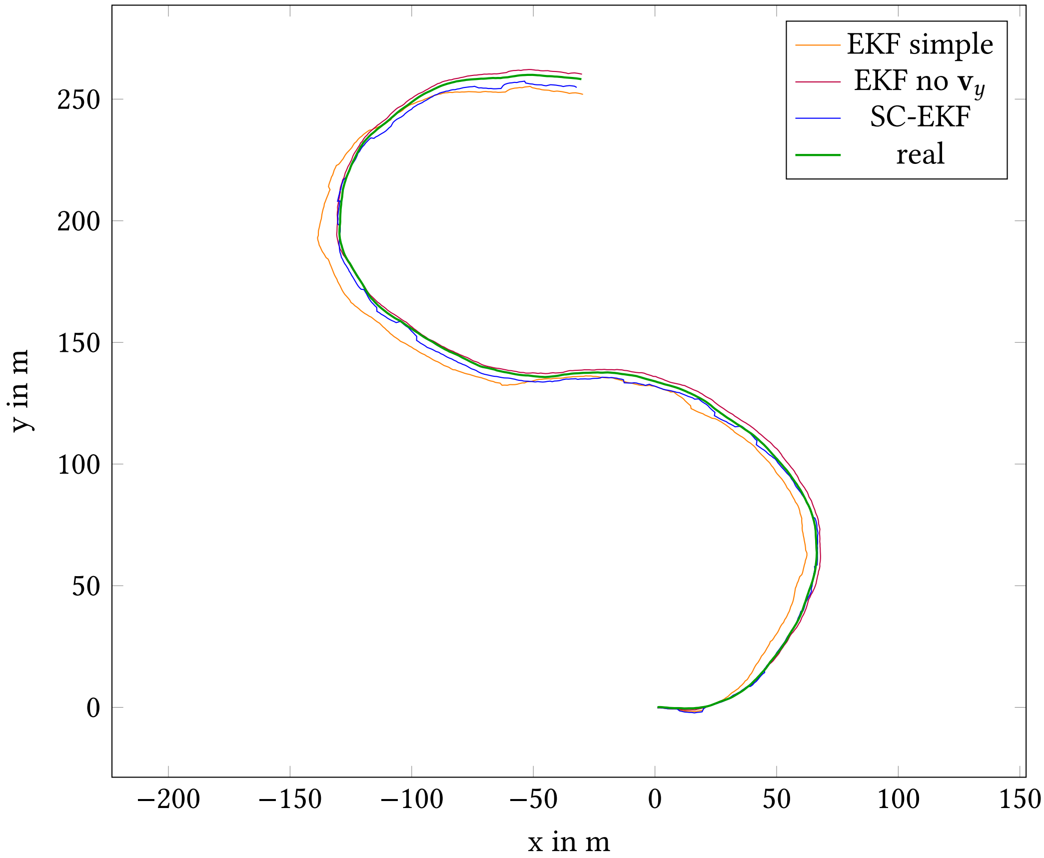


Figure 3.1: Exemplary localization results

3.1 Implementation

The filter was implemented as an explicit stochastic cloning EKF as described in section 2.3. The implementation was done in C++ using the Eigen linear algebra template library. Compile time index calculations using templates allowed flexible composition of the augmented state vector and covariance matrix without runtime overhead. This also made it possible to easily add a second relative sensor.

The size of the state vector and covariance matrix is not changed at runtime. Instead of adding and removing clones, space for one clone is reserved for every relative sensor. Once a relative measurement is finished the clone is immediately overwritten with a new clone as all relative measurements start immediately after the previous one.

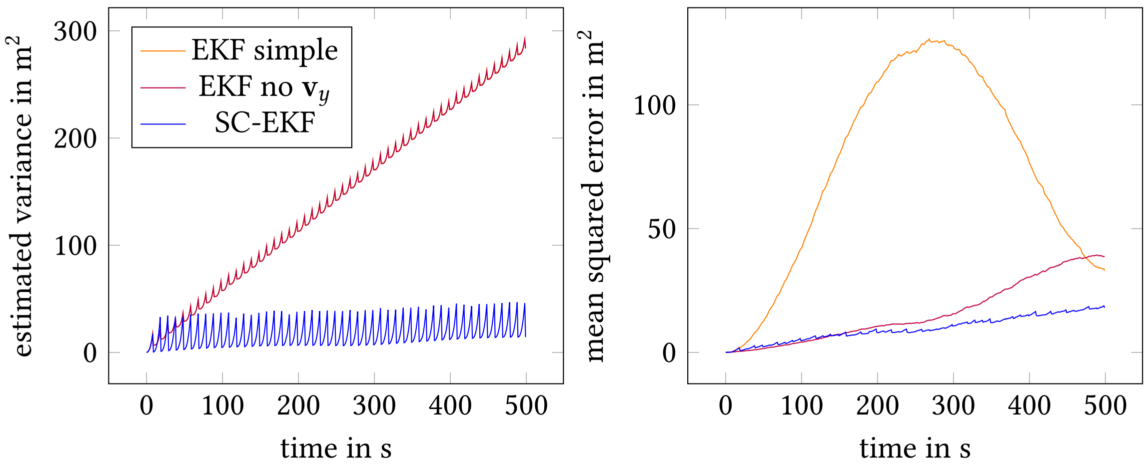


Figure 3.2: Estimated variance and mean squared error in the xy-plane

3.2 Results

To assess the errors of the different methods the results from 100 runs of the simulation were used (Fig. 3.2). From these the mean squared error is calculated as

$$\mathrm{MSE}_k = \frac{1}{100} \sum_{r=1}^{100} \left\| (\hat{\mathbf{p}}_{k,r} - \mathbf{p}_{k,r}) \right\|_2^2 \tag{3.11}$$

where again $\mathbf{p}_{k,r}$ is the xy-position at timestep k in run r.

When using the direct conversion from relative pose (Eq. 3.7) to velocity the EKF shows a large error. This is due to the fact that an average velocity is handled as an instantaneous velocity. The curvature of the path leads to a non-zero velocity being measured in y direction and the velocity in x direction being underestimated.

For a robot that can not move sideways this effect can be reduced by using the approximation shown in Equation 3.9. This however still slightly underestimates the x velocity.

The SC-EKF has a much lower error which shows that it models the actual system more closely. Its estimated variance also reflects the real error more accurately.

4 3D Filter

To evaluate the performance in a realistic setting stochastic cloning was used to incorporate odometric information into an existing EKF used on an autonomous vehicle. Previously only the pseudo-forward-velocity of the odometry was used.

4.1 The Vehicle

The vehicle (Fig. 4.1) is equipped with the following sensors: An XSens MTi-G-700 IMU and GNSS combo, providing accelerations and turning rates at 100 Hz and GNSS measurements at 4 Hz. The wheels are equipped with encoders from which a relative position is calculated every 10 Hz. Lastly a 3D laser scanner is mounted on top of the vehicle, which is however not used for localization in this thesis.

4.2 Filter Setup

The localization filter used on the vehicle is a Closed Loop Error State Space Extended Kalman Filter. This means the filter does not estimate the state directly but instead estimates the error of the prediction (error state space) and then subtracts this error from the state (closed loop) (Figure 4.2). The error vector can be expressed differently from the state vector. This localization filter expresses the orientation as a quaternion in the state vector but as roll pitch yaw in the error vector. This makes the measurement matrices easier to build since the rotation reference axes are the same as the position and velocity reference axes. As the rotation deltas in the error are always close to zero (since it is a closed loop filter) discontinuities in the Euler representation are not a problem.

Instead of a vehicle specific system model the IMU is used for prediction using a strapdown algorithm. This allows the filter to be used on many different robots since most robots have an IMU and the only thing that needs to be adapted to the specific robot are the variances. The strapdown algorithm is explained in [11, p. 45].

4.2.1 EKF

This section explains the existing EKF.

Figure 4.1: The Vehicle

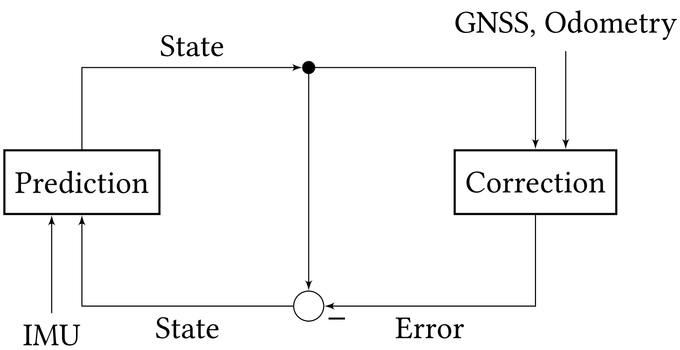


Figure 4.2: Schema of a closed loop error state space Kalman filter

State variables and measurements are expressed in the following coordinate systems as defined by [11, p. 28]:

- **b**ody - Fixed to the vehicle. The x axis points forward, the z axis points down.
- **n**avigation - The origin coincides with the body frame. The x axis points north, the y axis points east and the z axis points down (NED).
- **e**arth - Centered in and fixed to the earth ellipsoid. The x axis points to the intersection of the zero meridian and the equator. The z axis points to the north pole.

Turning rates are not included in the state and can not be added without changes to the system model. For this reason the conventional EKF only uses the pseudo-forward-velocity of the odometry. This is an advantage of the SC-EKF which can also use the change in yaw measured by the odometry since it incorporates them as a relative measurement between two orientations.

The state vector looks like this:

$$\mathbf{x} = \begin{pmatrix} \mathbf{q}_b^n & \mathbf{v}_{eb}^n & \text{LLA} & \mathbf{b}_a & \mathbf{b}_\omega & s_o \end{pmatrix}^T . \tag{4.1}$$

The entries are:

- Orientation of the body frame in the navigation frame as a quaternion
- Velocity of the body frame relative to the earth frame expressed in the navigation frame
- Latitude, Longitude, Altitude (LLA)
- Biases for IMU accelerations and turning rates
- Scale factor of the odometry

The Error vector is very similar, with the difference that the orientation error is expressed in Euler angles instead of a quaternion and the position error is expressed in Cartesian instead of geodetic coordinates.

$$\mathbf{e} = \begin{pmatrix} \Delta \text{rpy}_b^n & \Delta \mathbf{v}_{eb}^n & \Delta \mathbf{p}_b^n & \Delta \mathbf{b}_a & \Delta \mathbf{b}_\omega & \Delta s_o \end{pmatrix}^T \tag{4.2}$$

The measurement vector of the odometry contains only the forward velocity of the vehicle:

$$\mathbf{z} = \begin{pmatrix} (\bar{\mathbf{v}}_{eb}^b)_x \end{pmatrix} . \tag{4.3}$$

It is fused into the state using the following measurement model:

$$h(\mathbf{x}) = \left(\mathbf{C}_n^b \mathbf{v}_{eb}^n \right)_x s_o . \tag{4.4}$$

The model converts the estimated velocity to body coordinates and applies the estimated odometry scale factor.

From this model the following measurement Jacobian was derived:

$$\mathbf{H} = \begin{pmatrix} (\mathbf{C}_n^b)_x[\mathbf{v}_{eb}^n]_\times s_o & (\mathbf{C}_n^b)_x s_o & 0 & 0 & 0 & (\mathbf{C}_n^b \mathbf{v}_{eb}^n)_x \end{pmatrix}. \tag{4.5}$$

Note that while the function h takes a state vector as an argument, the matrix $\mathbf{H}$ is defined in the error space.

4.2.2 SC-EKF

For the SC-EKF the state vector and covariance matrix are doubled in size to contain static and evolving state and their covariances. While this of course increases computational cost significantly, the filter is still able to run multiple times faster than real time.

Also an additional scale factor and bias is introduced into the state for the angular part of the odometry.

So the non-augmented state and error vectors of the SC-EKF look like this:

$$\mathbf{x} = \begin{pmatrix} \mathbf{q}_b^n & \mathbf{v}_{eb}^n & \text{lat} & \text{lon} & \text{down} & \mathbf{b}_a & \mathbf{b}_\omega & s_{o_{\text{lin}}} & s_{o_{\text{ang}}} & b_{o_{\text{ang}}} \end{pmatrix}^T \tag{4.6}$$

$$\mathbf{e} = \begin{pmatrix} \Delta\text{rpy}_b^n & \Delta\mathbf{v}_{eb}^n & \Delta\mathbf{p}_b^n & \Delta\mathbf{b}_a & \Delta\mathbf{b}_\omega & \Delta s_{o_{\text{lin}}} & \Delta s_{o_{\text{ang}}} & \Delta b_{o_{\text{ang}}} \end{pmatrix}^T. \tag{4.7}$$

The measurement vector of the relative odometry contains the 2D position and orientation of the evolved body frame relative to the static body frame:

$$\mathbf{z} = \begin{pmatrix} (\bar{\mathbf{p}}_{b_{\text{ev}}}^{b_{\text{st}}})_{xy} \\ (\bar{\mathbf{q}}_{b_{\text{ev}}}^{b_{\text{st}}})_{\text{yaw}} \end{pmatrix}. \tag{4.8}$$

The following measurement model is used to predict the relative measurement from the state:

$$h\left(\begin{pmatrix} \mathbf{x}_{st} \\ \mathbf{x}_{ev} \end{pmatrix}\right) = \begin{pmatrix} (\mathbf{C}_b^n \Delta\mathbf{p}_n)_{xy} s_{o_{lin}} \\ (\mathbf{q}_{b_{st}}^{n*} \mathbf{q}_{b_{ev}}^n)_{\text{yaw}} s_{o_{ang}} + b_{o_{ang}} \end{pmatrix} \tag{4.9}$$

where $\Delta\mathbf{p}_n$ is the difference between the evolving and static geodetic coordinates converted to a Cartesian offset in meters expressed in the navigation frame.

4.3 Results

Since no ground truth is available for the recorded data no quantitative statements can be made about the localization results. Instead some qualitative statements are made with regard to the resulting poses, maps generated from these poses and the robustness against sensor failure.

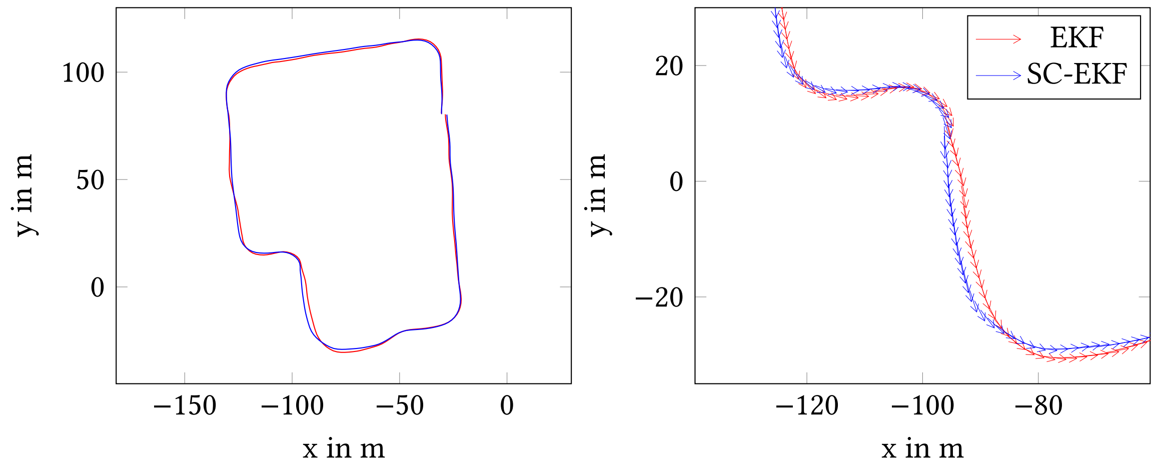


Figure 4.3: Comparison of localization results with an overview on the left and a detailed view on the right.

4.3.1 Comparison of resulting poses

Figure 4.3 shows the poses resulting from the SC-EKF and the original EKF. It can be seen that the SC-EKF produces straighter paths while still reaching the original location after closing the circle. It can be seen that local accuracy can be improved without loss of global accuracy.

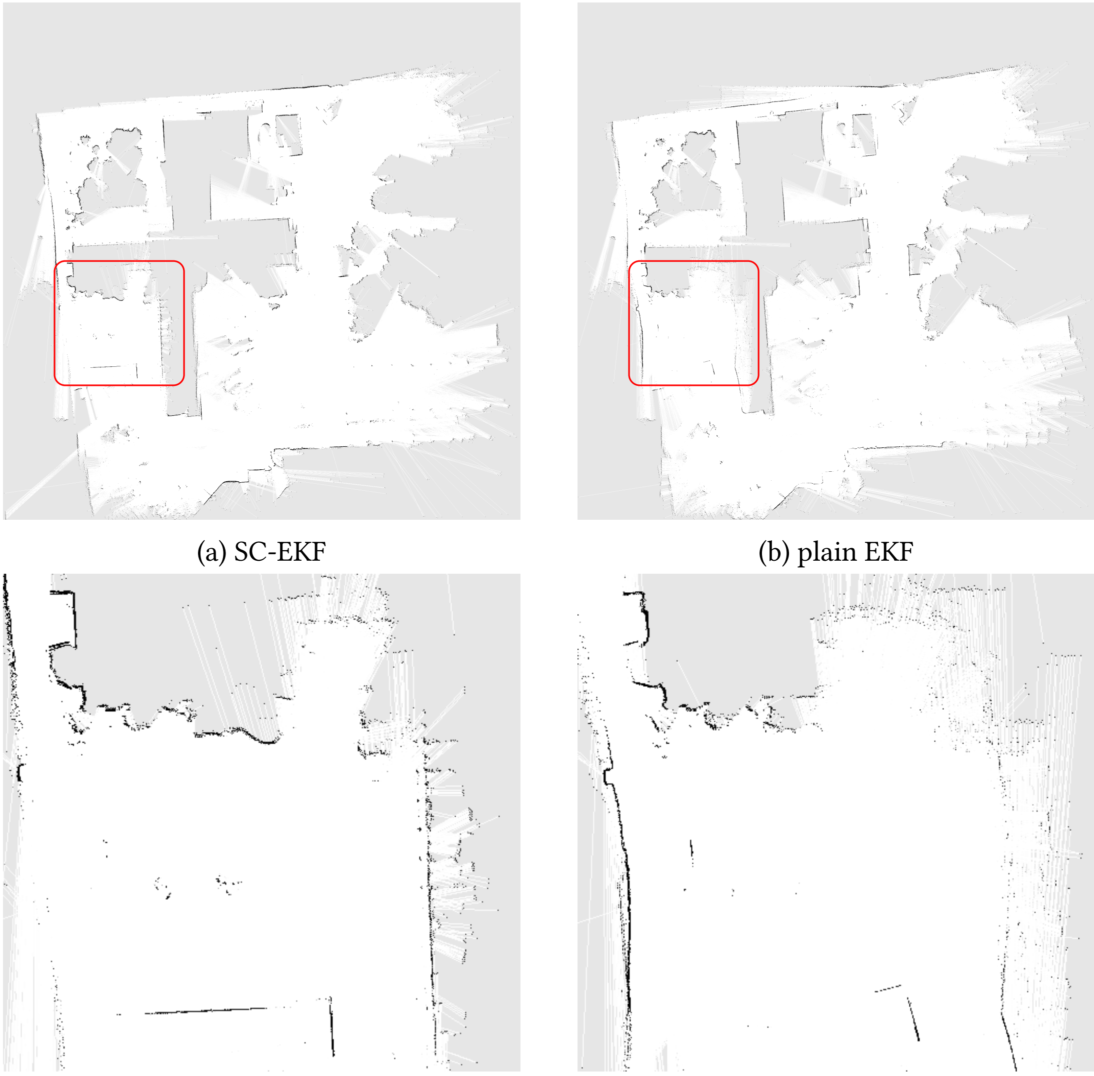
4.3.2 Comparison of maps generated using (SC-)EKF

To make a bit more reliable statements about the local accuracy of the localization poses grid maps were generated using a horizontal 2D cross section from the 3D laser scanner mounted on top of the vehicle. These show the relative consistency of the localization more clearly (Figure 4.4). It can be seen that the SC-EKF produces a sharper map with straighter lines, e. g., at the side of the building. This shows it provides better (relative) information, especially about the orientation of the vehicle, which is critical for producing consistent maps.

4.3.3 Comparison of robustness against sensor failure

Another important criterion for a good localization filter is robustness to sensor degradation or failure. Especially GNSS sensors are prone to temporary failures due to occlusion by, e. g., buildings or trees. To compare the robustness of the two filters each filter is run two times, once with the full dataset and once with a period of the GNSS measurements removed. The xy-distance and yaw-difference between the two runs is shown in figure 4.5.

Up until the sensor failure the two runs of course lead to the same result. During the failure both filters provide similar results regarding xy. The SC-EKF is however more stable



(a) SC-EKF

(b) plain EKF

(c) SC-EKF detail

(d) plain EKF detail

Figure 4.4: Comparison of maps generated from localization poses

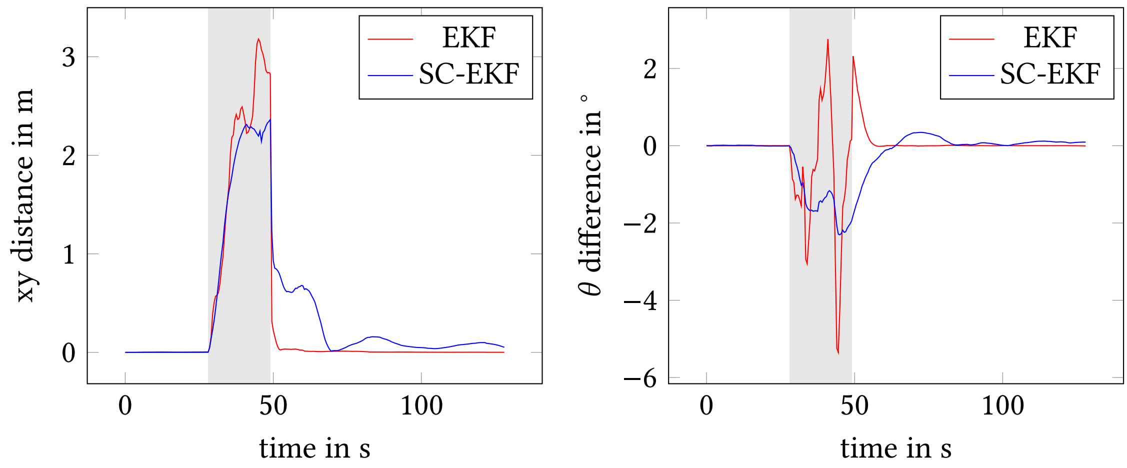


Figure 4.5: Comparison of robustness against sensor failure. The duration of the outage is marked in gray.

in the yaw angle, which would be expected as the SC-EKF is able to use the rotational information of the odometry.

After the sensor failure for both filters the differences become very small again. For the SC-EKF it takes a bit longer, which might be caused by the fact that the uncertainty in the position does not grow as much as for the EKF, thus the arriving GNSS measuremens do not have such a strong influence. But since both filters are compared to themselves and no real reference is available no certain statement can be made about which filter is more accurate.

5 Conclusion

Stochastic cloning is interesting especially in combination with an absolute sensor for position like GNSS as this allows to refrain from using complex SLAM solutions without steadily increasing localization error. For this reason the proposed method of using absolute sensors in combination with stochastic cloning is very relevant.

In Chapter 3 it was shown that stochastic cloning provides an advantage over approximate solutions. Chapter 4 showed that the method also works for a system of realistic complexity.

Linearization errors are especially relevant with stochastic cloning. It could furthermore be analyzed whether an iterated EKF provides an advantage here.

Bibliography

[1] Brian DO Anderson and John B Moore. *Optimal filtering*. Englewood Cliffs, 1979.

[2] Gary Bishop, Greg Welch, et al. “An introduction to the Kalman filter”. In: *Proc of SIGGRAPH, Course* 8.27599-3175 (2001), p. 59.

[3] Thomas Emter and Janko Petereit. “Stochastic Cloning and Smoothing for Fusion of Multiple Relative and Absolute Measurements for Localization and Mapping”. In: *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. IEEE. 2018, pp. 1508–1513.

[4] Rudolph Emil Kalman. “A new approach to linear filtering and prediction problems”. In: *Journal of basic Engineering* 82.1 (1960), pp. 35–45.

[5] Simon Lynen et al. “A robust and modular multi-sensor fusion approach applied to mav navigation”. In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE. 2013, pp. 3923–3929.

[6] Peter S Maybeck. *Stochastic models, estimation, and control*. Vol. 1. Academic press, 1979.

[7] Anastasios I Mourikis, Stergios I Roumeliotis, and Joel W Burdick. “SC-KF mobile robot localization: A stochastic cloning Kalman filter for processing relative-state measurements”. In: *IEEE Transactions on Robotics* 23.4 (2007), pp. 717–730.

[8] Stergios I Roumeliotis and Joel W Burdick. “Stochastic cloning: A generalized framework for processing relative state measurements”. In: *Robotics and Automation, 2002. Proceedings. ICRA’02. IEEE International Conference on*. Vol. 2. IEEE. 2002, pp. 1788–1795.

[9] Simo Särkkä. *Bayesian filtering and smoothing*. Vol. 3. Cambridge University Press, 2013.

[10] NA Thacker and AJ Lacey. “Tutorial: The Kalman filter”. In: ().

[11] Jan Wendel. *Integrierte Navigationssysteme: Sensordatenfusion, GPS und Inertiale Navigation*. Oldenbourg Verlag, 2009.