Proc. of IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), 2018, Madrid, Spain. DOI: 10.1109/IROS.2018.8594459


# Robust Sensor Fusion with Self-tuning Mixture Models

Tim Pfeifer and Peter Protzel
Dept. of Electrical Engineering and Information Technology
TU Chemnitz, Germany
Email: {firstname.lastname}@etit.tu-chemnitz.de

***Abstract*— A fundamental problem of non-linear state estimation in robotics is the violation of assumptions about the sensors' error distribution. State of the art approaches reduce the impact of these violations with robust cost functions or predefined non-Gaussian error models. Both require extensive parameter tuning and fail if the sensors' error characteristic changes over time, due to environmental changes, ageing or sensor malfunctions. We demonstrate how the error distribution itself can be part of the state estimation process. Based on an efficient approximation of a Gaussian mixture, we optimize the sensor model simultaneously during the standard state estimation. Due to an implicit expectation-maximization approach, we achieve a fast convergence without prior knowledge of the true distribution parameters. We implement this self-tuning algorithm in a least-squares optimization framework and demonstrate its real time capability on a real world dataset for satellite localization of a driving vehicle. The resulting estimation quality is superior to previous robust algorithms.**

## I. Introduction

To estimate the current state of a robotic or autonomous system, almost every probabilistic sensor fusion algorithm is constructed by assuming specific error properties of one or multiple sensors. Regardless of the choice of the concrete model, a violation of model assumptions can have a strong impact on the estimation quality. Algorithms that are immune against these violations are often referred to as robust algorithms. Nevertheless, many of these robust algorithms also require a parametrization, which depends on characteristics of the sensors' distribution. Previous evaluations [1], [2] demonstrated, how small the parameter window for optimal performance can be. Even with the right set of parameters, the performance of these robust approaches can be exceeded if the exact non-Gaussian sensor model is used [3]. However, the non-Gaussian model parameters are even harder to determine.

Therefore we want to offer an alternative approach that does not rely on a set of fixed parameters to describe the sensor. Instead of specifying them in advance, we include the parameters in the optimization problem itself.

Our previously proposed Dynamic Covariance Estimation algorithm (DCE) [2] showed how to include the sensors' covariance to the estimation problem. In the current work, a Gaussian mixture model (GMM) is the probabilistic foundation to cover more complex error distributions. Inspired by the work of Olson and Agarwal on Gaussian mixtures for graph optimization [5], [4], we chose a Max-Mixture (MM) model as efficient approximation of a Gaussian mixture. With this concept, we were able to extend our self-tuning DCE algorithm to multimodal mixture distributions.

The project is funded by the "Bundesministerium für Wirtschaft und Energie" (German Federal Ministry for Economic Affairs and Energy).

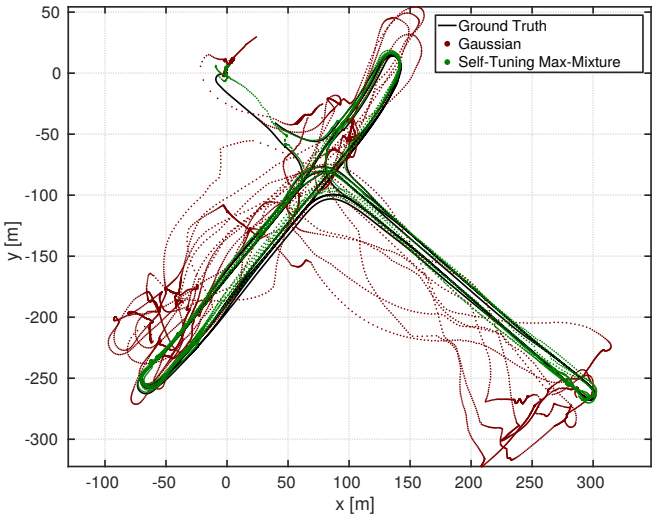


Fig. 1. Result of the online position estimation based on the Chemnitz City GNSS dataset. The top view of the estimated trajectory shows the performance of the proposed self-tuning algorithm (green) against the naive factor graph approach (red). Our approach is able to significantly reduce distortions that are caused by non-line-of-sight measurements.

In this work, we explain our choice of Max-Mixture as probabilistic model and its efficient self-tuning incorporation to the estimation process. We show how to handle the numerical stability issues that arise from this formulation and which prior knowledge about the sensors' properties is still required. A detailed validation is performed on the Chemnitz-City dataset used in [6].

## II. Prior Work

Initially motivated by the SLAM problem, optimization based algorithms have found their way to the field of general sensor fusion in the last decade [7], [8]. These class of algorithms is also known as factor graphs due to the corresponding graphical representation [9]. At first, we have a closer look to the principles of these technique. Based on the principle of maximum likelihood, (1) maximizes the probability of the system's state $\boldsymbol{X}$ for a set of measurements $\boldsymbol{Z}$.

$$\boldsymbol{X}^* = \underset{\boldsymbol{X}}{\operatorname{argmax}}\, \boldsymbol{P}(\boldsymbol{X}|\boldsymbol{Z}) \tag{1}$$

By assuming an uninformative prior, the Bayes' rule allows to rewrite this as maximum-likelihood inference of the measurements' conditional probability (2). The measurement vector $\mathbf{z}_\mathrm{i}$ and the state vector $\mathbf{x}_\mathrm{i}$ are subsets from the corresponding sets $\boldsymbol{Z}$ and $\boldsymbol{X}$.

$$\boldsymbol{P}(\boldsymbol{X}|\boldsymbol{Z}) \propto \prod_i \boldsymbol{P}(\mathbf{z}_\mathrm{i} \,|\, \mathbf{x}_\mathrm{i}) \tag{2}$$

Assuming Gaussian distributed measurement noise for each factor of (2), is the standard approach to implement almost any kind of sensor fusion algorithm. In case of factor graphs, the following least squares formulation can derived from (1) by applying the negative log-likelihood (3). For simplicity, we consider the one-dimensional case, but our statements also apply to multidimensional problems.

$$\boldsymbol{X}^* = \operatorname{argmin} \sum_i -\ln(\boldsymbol{P}(\mathbf{z}_\mathrm{i} \,|\, \mathbf{x}_\mathrm{i})) \tag{3}$$

$$\boldsymbol{X}^* = \underset{\boldsymbol{X}}{\operatorname{argmin}} \sum_i \| \underbrace{\mathbf{e}(\mathbf{x}_\mathrm{i}, \mathbf{z}_\mathrm{i})}_{\mathbf{e}_\mathrm{i}} \|^2_\Sigma \tag{4}$$

By squaring the non-linear error function $\mathbf{e}_\mathrm{i}$, a convex error surface results which is easy to optimize. Drawback of the quadratic term is the high sensitivity to large errors, caused by erroneous measurements that do not belong to the assumed Gaussian. The so called "outliers" occur in SLAM problems as well as with satellite or radio based localization and other physical sensors. A broad variety of approaches exist to handle them, but the majority relies on extensive parameter tuning.

A natural solution, when the assumed sensor model is violated, lays in the choice of a more sophisticated sensor model. In [4], Olson and Agarwal introduce the Max-Mixture algorithm, which uses an approximation of a Gaussian mixture distribution as the sensor model. To solve the problem of parametrization, they present in [10] a simple learning based approach to characterize a sensor like a GPS Receiver in advance. This approach requires previously collected data for every sensor that should be addressed. Also, they do not address the possibility of new and unseen conditions. Violations of the sensor model are still possible and therefore the impact on the estimation quality too.

Rosen et al. propose with [11] a more accurate approach to incorporate arbitrary distributions into the least squares estimation including Gaussian mixtures. In our comparison [3], this approach showed leading performance in terms of the estimation error. On the downside, there is also no mechanism to handle violations of the sensor model.

Recently a non-parametric inference algorithm [12] appeared to lift the factor graph approach beyond the limitation of closed form parametric distributions. The remaining question how to get the required non-parametric sensor distribution and the impact on the inference if they do not match the real sensors' properties is still open for future research.

Another common way to handle outliers is to exclude them from the estimation process or weight them down. This requires no direct assumptions about the sensors' real distribution. The distinction between outliers and valid measurements can be done binary or continuous.

Binary assignments [13] can not guarantee to detect all outliers, especially for applications where no sharp border to the class of valid measurements exists.

M-estimators offer a broad range of different cost functions to weight outliers down. They are the de-facto standard solution when robustness against outliers is required during the estimation process. However, the right one is hard to select and extensive parameter tuning can be required to achieve the desired performance. Agamennoni et al. introduced a self-tuning reinterpretation of M-estimators which eliminates this burden [14]. They generalize a subset of the available M-estimators as self-tuning elliptical distribution and try to approximate the sensors' error distribution with them. Elliptical distributions cover many differently shaped probability distributions, but all of them are symmetric and unimodal. Therefore, multi-modal sensor data can only be approximated by this approach and violations of the model assumptions are still possible.

With Switchable Constraints (SC) [15], Sünderhauf and Protzel included a continuous weighting of each measurement in the optimization process itself. While the independent weights between 1 and 0 are more suitable for data association problems like SLAM, in [16] and [6] the performance was also shown for satellite based localization (GNSS). Agarwal et al. provided a closed form solution of SC, called Dynamic Covariance Scaling (DCS) [17]. Both algorithms share the burden of extensive parametrization, since their performance depends mainly on the choice of an arbitrary tuning parameter as shown in [1]. With this downside, an application for sensor distributions that change over time is difficult. We reformulated the idea of a dynamically changed weight with our currently proposed Dynamic Covariance Estimation (DCE) algorithm [2]. Instead of tuning arbitrary weights, we enabled the optimizer to estimate each measurement's covariance directly. This seems identical but has different mathematical consequences: We were able to remove the arbitrary tuning parameters of SC/DCS and demonstrated improved convergence properties. Nevertheless, this approach is still limited to unimodal and symmetric distributions. In the following section we show how these limitations can be overcome by more sophisticated sensor models.

## III. Gaussian Mixture Models

We chose a sum of multiple Gaussian distributions as probabilistic model of the proposed algorithm. Our decision is based on the well-behaved mathematical properties of Gaussian mixture distributions as well as our practical experience with outliers in localization problems. The algorithmic foundations to include a GMM in factor graphs already exists, so our algorithm can be build on top of them.

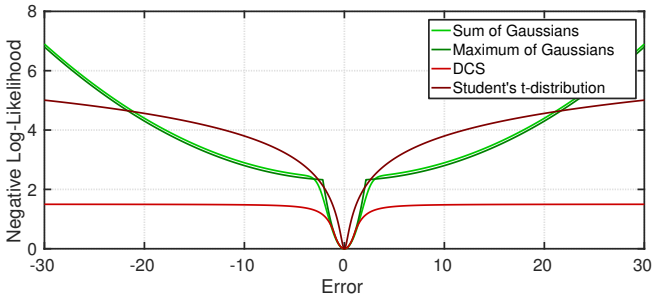


Fig. 2. The negative log-likelihood of a Gaussian mixture model (GMM) compared to the m-estimator Dynamic Covariance Scaling (DCS) [17] and a Student's t-distribution model from [14]. While the GMM has an easy to optimize convex surface, optimizing the m-estimator is more difficult due to the non-convex surface.

### A. Reasons for GMMs

Standard least squares optimization assumes a single zero-mean Gaussian $\boldsymbol{P}(\mathbf{z}_\mathrm{i} \,|\, \mathbf{x}_\mathrm{i}) \sim \mathcal{N}(0,\, \sigma^2)$ to describe the conditional probability

$$\boldsymbol{P}(\mathbf{z}_\mathrm{i} \,|\, \mathbf{x}_\mathrm{i}) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp\left(-\frac{{\mathbf{e}_\mathrm{i}}^2}{2\sigma^2}\right) \tag{5}$$

which leads to the mentioned least squares problem (4). To overcome the limitations of a single Gaussian, a sum of multiple ones can be used. With a weighted summation of multiple Gaussian $\mathcal{N}(\mu_j,\, \sigma_j^2)$, asymmetric distributions can be described as well as heavy-tailed and multimodal ones. The resulting error model is (6) with $w_j$ as weight, $\mu_j$ as mean and $\sigma_j^2$ as covariance of the $j$th component.

$$\boldsymbol{P}(\mathbf{z}_\mathrm{i} \,|\, \mathbf{x}_\mathrm{i}) \sim \sum_j w_j \cdot \mathcal{N}(\mu_j,\, \sigma_j^2) \tag{6}$$

Our comparison [3] revealed the potential of GMM-based approaches to outperform generic robust ones. We showed, that simulated NLOS measurements could easily be described with such a mixture. Later in Section VI-A we demonstrate on real GNSS data, that the non-Gaussian error characteristic of NLOS corrupted measurements can be described with a GMM. (See Fig. 4.)

Unlike other robust approaches, initial convergence after poor initialization is not necessarily a problem. At the start of the optimization, when the optimizer's state is far away from the local minimum, one Gaussian component usually dominates the cost function as shown in Fig. 2. Therefore, the log-likelihood is still quadratic and a fast large-scale convergence is possible similar to the single Gaussian case. Drawback of the quadratic large-scale error surface is the impact of unexpected outliers. Hence, the Gaussian mixture is only robust if it describes the correct error distribution. This is difficult, if the distribution changes over time due to environmental conditions or aging.

### B. Gaussian Mixtures in Factor Graphs

Gaussian mixtures can be incorporated into the estimation problem in two different ways. Olson and Agarwal showed an efficient approximation that replace sum operator $\Sigma$ of (6) with a maximum operator, which leads to formulation (7).

$$\boldsymbol{P}(\mathbf{z}_\mathrm{i} \,|\, \mathbf{x}_\mathrm{i}) \sim \max_j w_j \cdot \mathcal{N}(\mu_j,\, \sigma_j^2) \tag{7}$$

Rosen et al. proposed a more general approach to include any distribution, that is bounded in range, as the error model of the estimation problem. The implementation of their algorithm for sum of Gaussians leads to a difficult first derivation [3] which can cause convergence issues. Therefore, a variation of the Max-Mixture approach is used in this work which we want to explain in detail. For a single Gaussian component, the conditional probability is defined as

$$\boldsymbol{P}(\mathbf{z}_\mathrm{i} \,|\, \mathbf{x}_\mathrm{i}, \mu_j, \sigma_j, w_j) = \frac{w_j}{\sqrt{2\pi\sigma_j^2}} \cdot \exp\left(-\frac{(\mathbf{e}_\mathrm{i} - \mu_j)^2}{2\sigma_j^2}\right) \tag{8}$$

To obtain the required least squares form, the negative log-likelihood of (7) has to be formulated.

$$-\ln(\boldsymbol{P}) = -\ln\left(\max_j \frac{w_j}{\sqrt{2\pi\sigma_j^2}} \cdot \exp\left(-\frac{(\mathbf{e}_\mathrm{i} - \mu_j)^2}{2\sigma_j^2}\right)\right) \tag{9}$$

Since it is strictly monotonically increasing, the natural logarithm can be moved inside the maximum operator. Due to the negative sign, the maximum operator becomes a minimum operator.

$$-\ln(\boldsymbol{P}) = \min_j -\ln\left(\frac{w_j}{\sqrt{2\pi\sigma_j^2}} \cdot \exp\left(-\frac{(\mathbf{e}_\mathrm{i} - \mu_j)^2}{2\sigma_j^2}\right)\right) \tag{10}$$

The logarithm is applied to the single Gaussian inside the minimum operator and eliminates the exponential function.

$$-\ln(\boldsymbol{P}) = \min_j \left[\ln\left(\frac{\sqrt{2\pi\sigma_j^2}}{w_j}\right) + \frac{(\mathbf{e}_\mathrm{i} - \mu_j)^2}{2\sigma_j^2}\right] \tag{11}$$

Since constant it is over all $j$, $\ln\left(\sqrt{2\pi}\right)$ is pushed outside the minimum and squared error is written as Mahalanobis distance with $\Sigma_j = \sigma_j^2$.

$$-\ln(\boldsymbol{P}) = \underbrace{\ln\left(\sqrt{2\pi}\right)}_{\text{const.}} + \min_j \left[\ln\left(\frac{\sigma_j}{w_j}\right) + \frac{1}{2}\| \mathbf{e}_\mathrm{i} - \mu_j \|^2_{\Sigma_j}\right] \tag{12}$$

While the constant first part is neglected, a new regularization term $\gamma$ has to be added to guarantee $-\ln(\boldsymbol{P}) > 0$ [11]. We set this term to be greater than $-\min_j \ln\left(\frac{\sigma_j}{w_j}\right)$ which is constant if the distribution's parameters $\sigma_j$ and $w_j$ are constant.

$$-\ln(\boldsymbol{P}) \propto \min_j \left[\underbrace{\ln\left(\frac{\sigma_j}{w_j}\right)}_{\text{const.}} + \frac{1}{2}\| \mathbf{e}_\mathrm{i} - \mu_j \|^2_{\Sigma_j}\right] + \gamma \tag{13}$$

$$\text{with } \gamma > -\min_j \ln\left(\frac{\sigma_j}{w_j}\right) = \text{const.} \Leftarrow \sigma_j, w_j = \text{const.} \tag{14}$$

Note that our formulation of Max-Mixtures (13) differs from the original one in [5], due to the constant part and the conversion to a minimum operator. We prefer this formulation, because it approximates the error surface of a real sum of Gaussians, while the original formulation describes only a set of independent Gaussians. In the second case, the absolute value of the log-likelihood is inconsistent when the error model switches between two components. This becomes even more important if the distribution's parameter are not constant any more.

## IV. Self-tuning Mixtures

To overcome the limitations of static sensor models, we add each weight, mean and standard deviation of the GMM to the optimization problem. The maximum likelihood formulation (15) can be written straightforward.

$$\boldsymbol{X}^*, \boldsymbol{\sigma}^*, \boldsymbol{\mu}^*, \boldsymbol{w}^* = \underset{\boldsymbol{X},\boldsymbol{\sigma},\boldsymbol{\mu},\boldsymbol{w}}{\operatorname{argmax}} \prod_i \boldsymbol{P}(\mathbf{z}_\mathrm{i} \,|\, \mathbf{x}_\mathrm{i}, \boldsymbol{\sigma}, \boldsymbol{\mu}, \boldsymbol{w}) \tag{15}$$

To describe the negative log-likelihood of (15), (13) can be reused. However, the previous proposed regularization term (14) would lead to the trivial solution of $\sigma \to \infty$. To construct a more general regularization, we consider the special case $\mathbf{e}_\mathrm{i} = \mu_j$ where the log-likelihood has its global minimum.

$$-\ln(\boldsymbol{P}(\mathbf{z}_\mathrm{i} \,|\, \mathbf{e}_\mathrm{i} = \mu_j)) \propto \min_j \left[\ln\left(\frac{\sigma_j}{w_j}\right)\right] + \gamma \tag{16}$$

To guarantee $-\ln(\boldsymbol{P}) > 0$, the normalization term $\gamma$ has to satisfy (17).

$$\gamma > -\ln \min_j \left(\frac{\sigma_j}{w_j}\right) \tag{17}$$

While $\boldsymbol{P}$ is a normalized sum of Gaussians, the sum of $w_j$ is 1 and $w_j \geq 0$. So $\min(1/w_j) = 1$ reduces the regularization problem to (18).

$$\gamma > -\ln\left(\min_j \sigma_j\right) \tag{18}$$

There exists no $\gamma$ that guarantees $-\ln(\boldsymbol{P}) > 0$ for arbitrary $\sigma_j$. So as proposed for our DCE approach [2], a lower limit for the standard deviation $\sigma_{min}$ is necessary. This limit is set for all Gaussian components and depends on the sensors' properties under optimal conditions. In our experience, the general performance of the estimator is not affected by this regularization, since it is a constant offset. Therefore, the value can be chosen very conservatively, for example a tenth of the usual standard deviation. The log-likelihood (13) can be applied for the variable distribution parameter in form of (19).

$$-\ln(\boldsymbol{P}) \propto \min_j \left[\ln\left(\frac{\sigma_j}{w_j}\right) + \frac{1}{2}\| \mathbf{e}_\mathrm{i} - \mu_j \|^2_{\Sigma_j}\right] - \ln(\sigma_{min}) \tag{19}$$

The least squares problem (3) with (19) as log-likelihood is solved by iterative least-squares optimization. The proposed solution results implicitly in a variation of the Expectation-Maximization (EM) algorithm [18, p. 435]. This iterative algorithm can be applied to solve the estimation problem with some not fully observable variables, so called hidden states. At the beginning, an expectation step, where the most likely hidden variables are estimated based on the current visible states, is performed. In the following maximization step the observable variables are estimated while keeping the hidden ones fixed. Both states alternate until convergence is reached.

For GMMs, the assignment of each error function to one of the Gaussian components is the hidden variable and the components parameters are the visible ones. The minimum selector in (19) performs the expectation and the least-squares optimizer the maximization step. Due to the hard assignment through the minimum, the algorithm is a "winner-take-all" variant [19] of EM. Therefore we expect a fast initial convergence, while the final estimate is not the exact maximum likelihood solution for a GMM. Nevertheless, it should be accurate enough for our sensor model.

## V. GNSS Localization as Factor Graph

In robotics, factor graphs are used as graphical models to show the probabilistic connections between the state variables of an estimation problem. In the following section we describe the factors and variables that we use to solve the problem of satellite based localization for a ground vehicle. The overall structure of the graph is shown in Fig. 3.

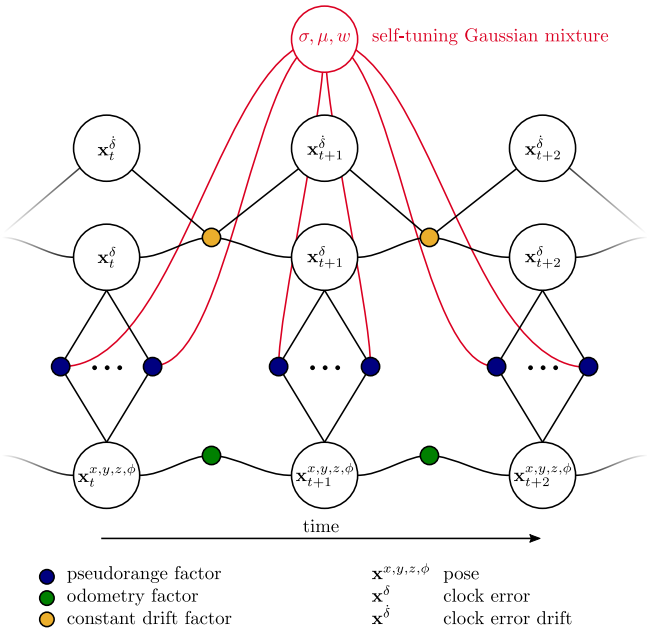


Fig. 3. The resulting factor graph representation of the GNSS localization problem. The small dots are error functions (factors) that define the least-squares problem. The big circles are the corresponding state variables. Highlighted with red is the main contribution of this paper, the self-tuning mixture model that is applied in the pseudorange factor.

### A. State Vector

We estimate the three-dimensional position of our system $\mathbf{x}_{\mathrm{t}}^{\mathrm{x,y,z}}$ in the Cartesian earth-centered, earth-fixed coordinate system (ECEF). However, the full three-dimensional rotations is hard to estimate due to an observability problem. Therefore, we only estimate the rotation $\mathbf{x}_{\mathrm{t}}^{\phi}$ around its upright body axis. We also estimate the GNSS receiver's clock error $\mathbf{x}_{\mathrm{t}}^{\delta}$ and its drift $\mathbf{x}_{\mathrm{t}}^{\dot{\delta}}$ to solve the GNSS problem.

### B. Pseudorange Factor

In satellite navigation, the time-of-flight of a radio signal is used to calculate the distance between the position of the $i$th satellite $\mathbf{s}_{\mathrm{t,i}}^{\mathrm{x,y,z}}$ and the receiver $\mathbf{x}_{\mathrm{t}}^{\mathrm{x,y,z}}$. The pseudorange measurements $\mathbf{z}_{\mathrm{t,i}}^{\mathrm{pr}}$ obtained from a GNSS receiver are biased for multiple reasons.

Largest component is the receiver clock error $\mathbf{x}_{\mathrm{t}}^{\delta}$, the difference between the receivers internal clock and the GNSS reference time. It varies over time and has to be estimated together with the receivers position. We estimate the offset directly as a metric distance instead of a time difference.

Each satellite also suffers from a clock error $\delta^{sat}$ which is typically provided by the satellite system.

Signal delays, caused by ionosphere and troposphere, add an offset of several meter. Both atmospheric error terms are also provided by the satellite system, we sum them to $\delta^{atm}$.

The Sagnac effect, caused by the earth rotation is also corrected using (20) from [16, p. 153].

$$\delta^{sag} = \tfrac{\omega^{earth}}{c} \cdot \left( \mathbf{s}_{\mathrm{t,i}}^{\mathrm{x}} \cdot \mathbf{x}_{\mathrm{t}}^{\mathrm{y}} - \mathbf{s}_{\mathrm{t,i}}^{\mathrm{y}} \cdot \mathbf{x}_{\mathrm{t}}^{\mathrm{x}} \right) \tag{20}$$

The constant $\omega^{earth}$ it the earth rotation rate and $c$ the speed of light.

All together results in (21) which is included in the self-tuning mixture formulation (19).

$$\mathbf{e}_{\mathrm{t,i}}^{\mathrm{pr}} = \sqrt{\left\| \mathbf{s}_{\mathrm{t,i}}^{\mathrm{x,y,z}} - \mathbf{x}_{\mathrm{t}}^{\mathrm{x,y,z}} \right\|^2} + \mathbf{x}_{\mathrm{t}}^{\delta} + \delta^{atm} + \delta^{sag} + \delta^{sat} - \mathbf{z}_{\mathrm{t,i}}^{\mathrm{pr}} \tag{21}$$

### C. Clock drift factor

The receivers clock offset is not static, since its clock source (typically a crystal) has a frequency error. Hence, it drifts at a certain velocity compared to the GNSS reference clock which is stabilized by an atomic clock. We apply a constant clock error drift (CCED) model (22) from [8] to estimate the clock error $\mathbf{x}_{\mathrm{t}}^{\delta}$ together with its derivation $\mathbf{x}_{\mathrm{t}}^{\dot{\delta}}$. These model approximates a linear frequency drift of the receivers clock source and allows a random walk of this drift.

$$\left\| \mathbf{e}_{\mathrm{t}}^{\mathrm{CCED}} \right\|_{\Sigma^{CCED}}^2 = \left\| \begin{pmatrix} \mathbf{x}_{\mathrm{t}}^{\delta} + \mathbf{x}_{\mathrm{t}}^{\dot{\delta}} \cdot \Delta t \\ \mathbf{x}_{\mathrm{t}}^{\dot{\delta}} \end{pmatrix} - \begin{pmatrix} \mathbf{x}_{\mathrm{t+1}}^{\delta} \\ \mathbf{x}_{\mathrm{t+1}}^{\dot{\delta}} \end{pmatrix} \right\|_{\Sigma^{CCED}}^2 \tag{22}$$

### D. Odometry Factor for Ground Vehicles

The ground vehicle's wheel odometry measures a velocity $\mathbf{z}_{\mathrm{t}}^{\mathrm{v}}$ and yaw rate $\mathbf{z}_{\mathrm{t}}^{\omega}$. Since the odometry contains only the velocity in the local x-direction $\tilde{x}$, we assume the velocity in the other dimensions to be zero. Its full 3D pose with three degree of freedom for the orientation is only partially observable. Then, if the vehicle moves in a straight line, the rotation around its forward axis can not be observed. Therefore, we estimate only one rotation around the vehicles vertical axis $\mathbf{x}_{\mathrm{t}}^{\phi}$ along with its global 3D position $\mathbf{x}_{\mathrm{t}}^{\mathrm{x,y,z}}$. We assume that the vehicles upright axis is perpendicular to the earth surface. This is a approximation of the real motion, although it should be accurate enough for the localization problem and it solves the observability problem. The required rotation from global coordinates to the local body frame is implemented with the quaternion

$$\mathbf{q}_{rot} = \left( \frac{\mathbf{q}_z \cdot \mathbf{q}_\phi}{\left\| \mathbf{q}_z \cdot \mathbf{q}_\phi \right\|} \right)^{-1} \tag{23}$$

$\mathbf{q}_{rot}$ is composed from the angle between the global z-axis $\mathbf{v}^z = [0,0,1]^\intercal$ and the position $\mathbf{x}_{\mathrm{t}}^{\mathrm{x,y,z}}$ which is

$$\mathbf{q}_z = \angle \mathbf{v}^z \, \mathbf{x}_{\mathrm{t}}^{\mathrm{x,y,z}} = \begin{pmatrix} \mathbf{v}^z \times \mathbf{x}_{\mathrm{t}}^{\mathrm{x,y,z}} \\ \mathbf{v}^z \cdot \mathbf{x}_{\mathrm{t}}^{\mathrm{x,y,z}} + \left\| \mathbf{v}^z \right\| + \left\| \mathbf{x}_{\mathrm{t}}^{\mathrm{x,y,z}} \right\| \end{pmatrix} \tag{24}$$

and the angle $\mathbf{x}_{\mathrm{t}}^{\phi}$. Therefore, $\mathbf{q}_\phi$ (25) rotates the system around the position vector (which is assumed to be the vehicles z-axis) by $\mathbf{x}_{\mathrm{t}}^{\phi}$.

$$\mathbf{q}_\phi = \begin{pmatrix} \sin(\mathbf{x}_{\mathrm{t}}^{\phi}/2) \\ 0 \\ 0 \\ \cos(\mathbf{x}_{\mathrm{t}}^{\phi}/2) \end{pmatrix} \tag{25}$$

The complete error function for our ground vehicle odometry factor is $\mathbf{e}_{\mathrm{t}}^{\mathrm{odo}}$, with (27) as relative position $\Delta\, \mathbf{x}_{\mathrm{t}}^{\tilde{\mathrm{x}},\tilde{\mathrm{y}},\tilde{\mathrm{z}}}$ in the local body coordinates $\tilde{x}, \tilde{y}, \tilde{z}$.

$$\left\| \mathbf{e}_{\mathrm{t}}^{\mathrm{odo}} \right\|_{\Sigma^{odo}}^2 = \left\| \begin{pmatrix} \Delta\, \mathbf{x}_{\mathrm{t}}^{\tilde{\mathrm{x}},\tilde{\mathrm{y}},\tilde{\mathrm{z}}} \\ \left( \mathbf{x}_{\mathrm{t+1}}^{\phi} - \mathbf{x}_{\mathrm{t}}^{\phi} \right) \end{pmatrix} \cdot \Delta t^{-1} - \begin{pmatrix} \mathbf{z}_t^v \\ 0 \\ 0 \\ \mathbf{z}_{\mathrm{t}}^{\omega} \end{pmatrix} \right\|_{\Sigma^{odo}}^2 \tag{26}$$

$$\Delta\, \mathbf{x}_{\mathrm{t}}^{\tilde{\mathrm{x}},\tilde{\mathrm{y}},\tilde{\mathrm{z}}} = \mathbf{q}_{rot} \left( \mathbf{x}_{\mathrm{t+1}}^{\mathrm{x,y,z}} - \mathbf{x}_{\mathrm{t}}^{\mathrm{x,y,z}} \right) \mathbf{q}_{rot}^{-1} \tag{27}$$

## VI. Real World Experiment

Motivation of our work is the problem of precise GNSS localization under urban multipath conditions. Hence, we evaluate the proposed self-tuning Mixtures optimization on a dataset of real world GNSS measurements. In this section, we want to give a brief summary of the used dataset, how we implemented the factor graph to solve it and how to parametrize the sensor models.

### A. Chemnitz City Dataset

Already known from the publications of Sünderhauf [6], [8], [16], the Chemnitz City dataset contains a synchronized set of pseudorange and odometry measurements, as well as a precise ground truth. Due to the urban environment, the

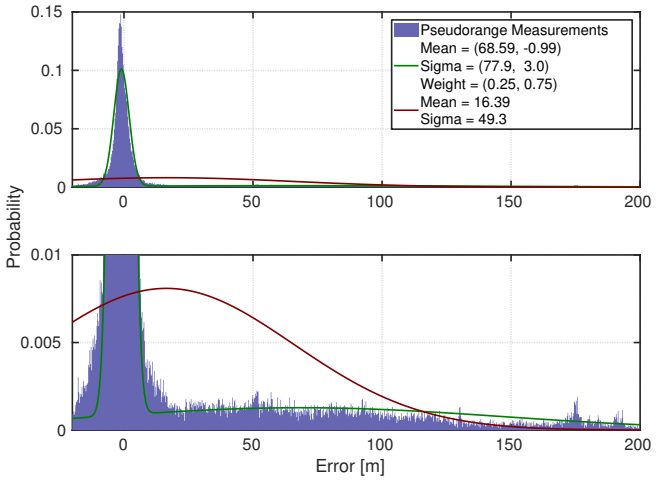


Fig. 4. The histogram of the pseudorange error over the complete Chemnitz City Dataset (blue) with a fitted 2-component Gaussian mixture model (green) and a single Gaussian (red). Both models are fitted with the `fitgmdist` function of Matlab. The lower plot is an magnification of the upper plot. It is clearly visible that the single Gaussian distribution is not sufficient to describe the error distribution.

direct line-of-sight to some of the satellites is blocked by buildings or other obstacles. Signals from a part of these, still reach the receiver by reflections on the walls. While the high precision reference receiver can suppress them with its advanced antenna and algorithms, these non-line-of-sight (NLOS) measurements are recorded by the consumer grade receiver. Fig. 4 shows the error histogram of all measurements, which displays the heavy-tailed error distribution. We estimated the corresponding Gaussian mixture with the Matlab function `fitgmdist`. The distribution's parameters indicate a proportion of over 22% NLOS for the whole dataset which reflects the challenging conditions. To exploit the changing conditions during the test drive, we repeated this with a sliding window. Fig. 5 in Section VII-B shows, that the characteristics as well as the amount of the NLOS errors are variable over time. Note that in practical applications this evaluation is not possible without a centimetre-level precise reference system. From now on, the dataset is online available[1].

### B. Graph Optimization

The proposed factor graph with a self-tuning GMM as sensor model is implemented inside the Ceres framework [20]. Even if a recorded dataset is used, we process the data under online conditions. We update the graph with new factors and variables each time step and use a Dogleg optimizer to solve it. To keep the runtime of our estimation problem bounded, we apply a simple sliding window filter. The filter length is set to $60\,\mathrm{s}$ to include enough data to estimate the GMM, while ensuring real-time capability. This trade-off is set empirically, but the impact to the performance is low, even for shorter windows.

[1] `www.mytuc.org/rxvw`

### C. Parametrization

Every factor of our factor graph has to be weighted by a covariance. These parametrization is crucial for the performance of our estimator. Since only the covariance of the pseudorange factor is estimated during the optimization, the odometry and clock drift factors have to be parametrized a priori. These noise parameters depend on the hardware used for the dataset and since it was recorded in 2011, the original hardware is not longer available. Regardless the empirical chosen values, provided by [6] and [16], we estimated them from the dataset to ensure a correct parametrization. The precise ground truth allows to calculate the errors of the odometry and clock drift. Based on these errors, we estimated the noise and drift properties shown in Table I. The covariance of the pseudorange is taken from [6] and it is only used for the non-robust graph.

TABLE I
COVARIANCES OF THE ESTIMATION PROBLEM

| sensor | error function | covariance |
|---|---|---|
| pseudorange | $\mathbf{e}_{\mathrm{t,i}}^{\mathrm{pr}}$ | $\Sigma_{pr} = (10\,\mathrm{m})^2$ |
| odometry | $\mathbf{e}_{\mathrm{t}}^{\mathrm{odo}}$ | $\Sigma_{odo} = \mathrm{diag}\begin{pmatrix} 0.05\,\mathrm{m\,s^{-1}} \\ 0.03\,\mathrm{m\,s^{-1}} \\ 0.03\,\mathrm{m\,s^{-1}} \\ 0.006\,\mathrm{rad\,s^{-1}} \end{pmatrix}^2$ |
| CCED model | $\mathbf{e}_{\mathrm{t}}^{\mathrm{CCED}}$ | $\Sigma_{CCED} = \begin{pmatrix} 0.1\,\mathrm{m} \\ 0.009\,\mathrm{m\,s^{-1}} \end{pmatrix}^2$ |

Before the optimization, initial values for the covariances, means and weights of the self-tuning mixture model have to be defined. They are, in difference to the other noise parameters, not static and are only used during the first iterations. To demonstrate the usability in practical applications, we do not use any information from the dataset to determine these values. Instead, we use a generic robust mixture of a Gaussian with a covariance of $10\,\mathrm{m}$ corrupted by $25\,\%$ outliers with a covariance of $100\,\mathrm{m}$. Table II summarize the GMM initial parametrization. We also added lower and upper limits to each value. This is required, due to the clock error as common offset of the pseudoranges. It prevents the unbounded estimation of the mean values $(\mu_1, \mu_2)$. Therefore, we fixed $\mu_1$ to 0 and limited $\mu_2$ to be positive. With limits for the weight parameters, we also forbid the optimizer to discard a Gaussian component completely.

TABLE II
LIMITATIONS AND INITIAL VALUES OF THE GMM

| Parameter | Start | Min | Max |
|---|---|---|---|
| $(\sigma_1, \sigma_2)$ [m] | (10, 100) | (1, 20) | (10, $\infty$) |
| $(\mu_1, \mu_2)$ [m] | (0, 0) | (0, 0) | (0, $\infty$) |
| $(w_1, w_2)$ | (0.75, 0.25) | (0.1, 0.1) | (0.9, 0.9) |

## VII. Results

### A. Position Accuracy

We plot the online position estimates along with the ground truth coordinates as top view in Fig. 1. To allow comparisons to prior work [6], we use the trajectory error in the UTM xy-plane as error metric and summarize it in Table III. The result of a non-robust implementation is in the first row and demonstrates how the NLOS measurements affect the position accuracy even with odometry information and the clock drift model. To show the difference between a static GMM and our self-tuning model, we included a Max-Mixture implementation with fixed parameters in the second row. The third row contains the error of our proposed self-tuning mixture algorithm. In addition to our own work, the results of Sünderhauf et al. are also listed in the last two rows. Please notice that the result from [16] is achieved by batch processing and [6] without odometry information.

TABLE III
Results of the final run

| Algorithm | Trajectory Error [m] | | | Time [s] |
|---|---|---|---|---|
| | Median | Mean | Max | |
| Non-robust | 27.15 | 30.01 | 98.47 | 125.1 |
| Static GMM | 3.57 | 4.52 | 31.75 | 91.3 |
| Self-tuning GMM | 1.99 | 2.45 | 13.91 | 185.0 |
| SC (no odometry) [6] | 2.55 | 3.21 | 21.04 | – |
| SC (batch) [16] | 2.56 | 2.86 | 9.35 | – |

The results show the advantages of a dynamic error model estimation over static parametrization. The significantly reduced estimation errors demonstrate the robustness of our self-tuning mixtures approach. Compared to previous results of Switchable Constraints, it is clear that particularly the mean and median errors are improved. This suggests that the dynamic error model is able to improve the estimation quality compared to generic robust approaches.

### B. Estimated Error Model

To validate the estimated parameters of the Gaussian mixture model, we compare them to an external estimate as ground truth. With a sliding window of the same length as our factor graph, we use again Matlab's `fitgmdist` function to calculate the parameter. Fig. 5 displays the parameters (in color) with their respective external estimate (in black). Obviously, the mean of the first component is fixed to zero, as explained in section VI-C. The overall estimation quality of our approach is good, considering the winner-take-all behaviour of the EM algorithm. Binary assignment of each measurement to one Gaussian component causes the noisy weight estimate, since small parameter changes can lead to many reassignments. The systematic underestimation of the first component's standard deviation is probably caused by the hard assignments too. This has to be investigated in further research.

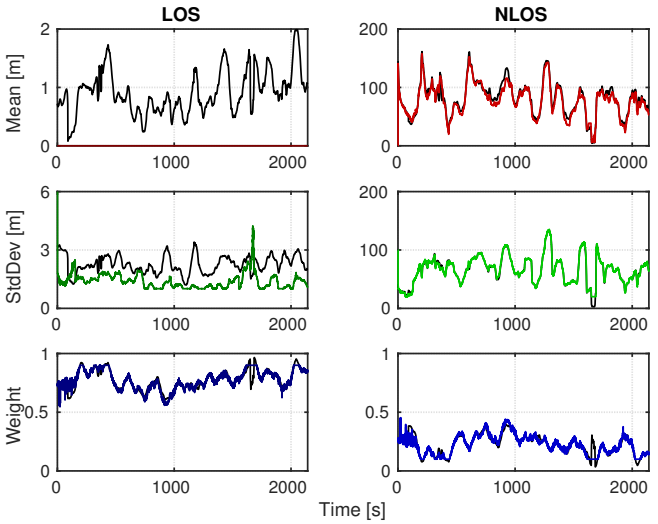


Fig. 5. Comparison of the mixture model parameters estimated in the proposed factor graph (in color) and the externally estimated ground truth (in black). The ground truth is based on a sliding window of 60s as well as the proposed algorithm. In the left column are the mean, standard deviation and weight of the line-of-sight (LOS) component of the estimated Gaussian mixture. Its mean is fixed to zero. The right column contains the none-line-of-sight (NLOS) component.

### C. Runtime

The time measurements in Table III are achieved on a Intel i7-4770 System. While the average computation time per time step is about 2.4 ms, Fig. 6 shows the strong variation in the individual steps. The maximal time per step is 22.6 ms. Since dataset is recorded with 25 ms per step, our approach is real-time capable without limitations. We do not include the runtime of [6] and [16] because they were run on different machines with older frameworks.

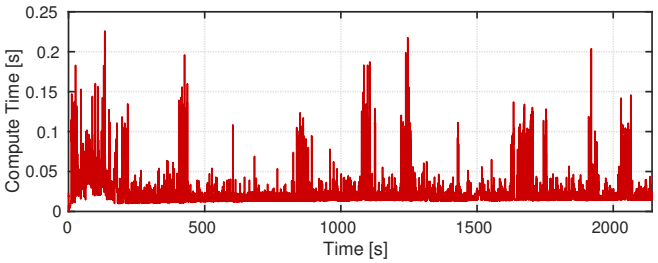


Fig. 6. Computation time required by the proposed self-tuning algorithm between two consecutive time steps. Despite the large fluctuations, the maximum available time of 25 ms is never exceeded.

## VIII. Conclusion

We introduced our concept of a self-tuning Gaussian mixture distribution and demonstrated its application to a online GNSS localization problem.

Through combination of our previously proposed DCE algorithm [2] with Olsons efficient Max-Mixture representation [5] we formulated the problem of state estimation and sensor characterization jointly as maximum-likelihood problem. The evaluation on a GNSS localization dataset showed how the dynamic estimation of the pseudorange error

model can improve the overall estimation quality. In our opinion, dynamic estimation of the sensor model is just a necessary step to handle sensors like GNSS, wireless point-to-point ranging or radar. Also non-physical sensors like feature detectors for vision based systems that are affected by changing lighting and weather conditions could be addressed. Therefore, we want to investigate in our future work, how these sensors can be described with a dynamic mixture model.

Even if our approach looks promising, there are still some limitations. The self-tuning sensor model is variable in its parameters, but fixed in its structure. Hence, it can only adapt to the sensors' distribution, if the number of Gaussian components are enough to describe the distribution. Because of the hard assignments in the mixture model, we also expect suboptimal performance if the number of components is to high. If a component has no measurement assigned, it can simply diverge. So, a model that is variable in its structure should be topic of future research.

The presented connections to the EM-algorithm and the factor graph representation allow to extend the proposed approach in many different directions to overcome some of the limitations. Also, we can imagine a combination with machine learning techniques to learn the sensors' behaviour over time.

## REFERENCES


[1] Y. Latif, C. Cadena, and J. Neira, "Robust graph SLAM back-ends: A comparative analysis," in *Proc. of Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2014.

[2] T. Pfeifer, S. Lange, and P. Protzel, "Dynamic covariance estimation – a parameter free approach to robust sensor fusion," in *Proc. of Intl. Conf. on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, Nov. 2017, pp. 359–365.

[3] T. Pfeifer, P. Weissig, S. Lange, and P. Protzel, "Robust factor graph optimization – a comparison for sensor fusion applications," in *Proc. of Intl. Conf. on Emerging Technologies and Factory Automation (ETFA)*, 2016.

[4] E. Olson and P. Agarwal, "Inference on networks of mixtures for robust robot mapping," *Intl. Journal of Robotics Research*, 2013.

[5] ——, "Inference on networks of mixtures for robust robot mapping," in *Proc. of Robotics: Science and Systems (RSS)*. Sydney, Australia: Robotics: Science and Systems Foundation, Jul. 2012.

[6] N. Sünderhauf, M. Obst, S. Lange, G. Wanielik, and P. Protzel, "Switchable constraints and incremental smoothing for online mitigation of non-line-of-sight and multipath effects," in *Proc. of Intelligent Vehicles Symposium*, 2013.

[7] S. Lange, N. Sünderhauf, and P. Protzel, "Incremental smoothing vs. filtering for sensor fusion on an indoor UAV," in *Proc. of Intl. Conf. on Robotics and Automation (ICRA)*, 2013.

[8] N. Sünderhauf, M. Obst, G. Wanielik, and P. Protzel, "Multipath mitigation in GNSS-based localization using robust optimization," in *Proc. of Intelligent Vehicles Symposium*, Jun. 2012, pp. 784–789.

[9] F. Dellaert, M. Kaess *et al.*, "Factor graphs for robot perception," *Foundations and Trends in Robotics*, vol. 6, no. 1-2, pp. 1–139, 2017.

[10] R. Morton and E. Olson, "Robust sensor characterization via max-mixture models: GPS sensors," in *Proc. of Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2013.

[11] D. M. Rosen, M. Kaess, and J. J. Leonard, "Robust incremental online inference over sparse factor graphs: Beyond the Gaussian case," in *Proc. of Intl. Conf. on Robotics and Automation (ICRA)*, 2013.

[12] D. Fourie, J. Leonard, and M. Kaess, "A nonparametric belief solution to the bayes tree," in *Proc. of Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 2189–2196.

[13] Y. Latif, C. Cadena, and J. Neira, "Robust loop closing over time for pose graph SLAM," *Intl. Journal of Robotics Research*, 2013.

[14] G. Agamennoni, P. Furgale, and R. Siegwart, "Self-tuning m-estimators," in *Proc. of Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 4628–4635.

[15] N. Sünderhauf and P. Protzel, "Switchable constraints for robust pose graph slam," in *Proc. of Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2012.

[16] N. Sünderhauf, "Robust optimization for simultaneous localization and mapping," Ph.D. dissertation, TU Chemnitz, 2012.

[17] P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard, "Robust map optimization using dynamic covariance scaling," in *Proc. of Intl. Conf. on Robotics and Automation (ICRA)*, 2013.

[18] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer: New York, 2006.

[19] R. M. Neal and G. E. Hinton, "A view of the em algorithm that justifies incremental, sparse, and other variants," in *Learning in graphical models*. Springer, 1998, pp. 355–368.

[20] S. Agarwal, K. Mierle, and Others, "Ceres Solver," http://ceres-solver.org.