Self-supervised Calibration for Robotic Systems

Jérôme Maye, Paul Furgale, and Roland Siegwart
Autonomous Systems Lab, ETH Zurich, Switzerland
email: {jerome.maye, paul.furgale, roland.siegwart}@mavt.ethz.ch

***Abstract*— We present a generic algorithm for self calibration of robotic systems that utilizes two key innovations. First, it uses information theoretic measures to automatically identify and store novel measurement sequences. This keeps the computation tractable by discarding redundant information and allows the system to build a sparse but complete calibration dataset from data collected at different times. Second, as the full observability of the calibration parameters may not be guaranteed for an arbitrary measurement sequence, the algorithm detects and locks unobservable directions in parameter space using a truncated QR decomposition of the Gauss-Newton system. The result is an algorithm that listens to an incoming sensor stream, builds a minimal set of data for estimating the calibration parameters, and updates parameters as they become observable, leaving the others locked at their initial guess.**

Through an extensive set of simulated and real-world experiments, we demonstrate that our method outperforms state-of-the-art algorithms in terms of stability, accuracy, and computational efficiency.

I. Introduction

Every robotic system has some set of parameters—scale factors, sensor locations, link lengths, etc.—that are needed for state estimation, planning, and control. Despite best efforts during construction, some parameters will change over the lifetime of a robot due to normal wear and tear. In the best case, incorrect parameter values degrade performance. In the worst case, they cause critical safety issues.

We are interested in developing automated systems that are capable of robust long-term deployment in the hands of non-experts, so the automatic identification and update of these parameter values is highly important.

As an example, consider a camera-based collision avoidance system intended for deployment in a consumer automobile. For the vehicle to be able to avoid collisions, the pose of each camera with respect to the vehicle coordinate system must be known precisely so that obstacle positions can be transformed from camera coordinates into vehicle coordinates. However, a consumer vehicle will have no access to special calibration hardware or expert data analysis. In such a scenario, the vehicle must be capable of self-supervised recalibration. This problem is inherently difficult for a number of reasons that we will briefly discuss here.

1) Parameters change over time—although the vehicle may be factory calibrated, the transformations can change slowly over time due to vibration, thermal expansion, loose parts, or any number of other common problems that follow from normal usage.

2) Parameters must be inferred from the data—as the cameras may be installed in different places on the vehicle, their pose cannot be measured directly. Instead, the transformations must be inferred from the data produced by the full system. However, this is only possible if the motion of the vehicle renders the parameters *observable*[1].

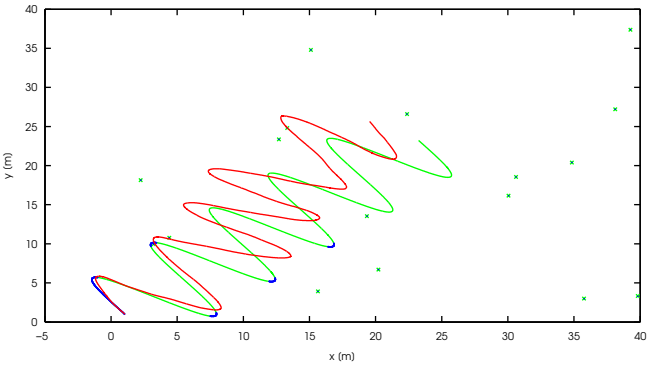

Fig. 1. Exemplary output of our calibration algorithm. The ground truth trajectory and landmark positions are shown in green, their respective estimates in blue, and the integrated odometry path in red. An information theoretic measure is used to automatically identify and store novel measurement sequences (shown in blue), discarding redundant information and building a sparse but complete calibration dataset from data collected at different times.

3) Normal operation may result in unobservable directions in parameter space—unfortunately, for this and many other practical problems normal operation may not render all directions in parameters space observable. In this example, when two cameras do not share an overlapping field of view, planar motion renders the calibration problem degenerate[2]; the transformation between cameras only becomes observable under general 3D motion.

4) Unobservable directions in parameter space may appear observable in the presence of noise—even if our hypothetical car is piloted only on a plane (a degenerate case), noise in the measurements can make unobservable parameters appear observable. We call these parameters *numerically unobservable* to mirror the related concept of numerically rank-deficient matrices.

Existing algorithms for self calibration generally handle issues (1) and (2). Issue (3) is dealt with by designing experiments that guarantee all parameters become observable. To the best of our knowledge, no published self-calibration

[1] Informally, observability means that the parameters can be inferred from some local batch of measurement data [1]

[2] See [2] for a handy table of degenerate cases or [3] for a more theoretical analysis.

algorithm is able to cope with issue (4). Therefore, unless we plan to require all vehicle owners to outfit their parking places with calibration patterns or regularly drive off-road, new advances for online system calibration are required.

In this paper, we propose an algorithm to deal with *all* of these difficulties. Our approach exploits the algebraic links between the Gauss-Newton algorithm, the Fisher Information Matrix, and nonlinear observability analysis to automatically detect directions in parameter space that are numerically unobservable and avoid updating our parameters in these directions; at any given time, directions in the parameter space that are observable will be updated based on the latest information, while unobservable directions will remain at their initial guess. Novel sets of measurements are detected using a mutual information test, and added to a working set that is used to estimate the parameters. The result is an algorithm that listens to an incoming stream of data, automatically accumulating a batch of data that may be used to calibrate a full robot system. The only requirements are that (i) the parameters are theoretically observable given some ideal set of data, (ii) it is possible to implement a batch Gauss-Newton estimator for the system state and calibration parameters based on a set of measurements, and (iii) we have some reasonable initial guess for the calibration parameters (e.g. from factory calibration). Fig. 1 shows a typical calibration run of our algorithm.

The remainder of the paper is structured as follows. Section II will give a brief overview over related approaches. Section III is dedicated to the mathematical grounding of our method. Section IV demonstrates the validity of our approach through extensive experiments and evaluation. Section V will conclude the paper.

II. Related Work

The problem of sensor calibration has been a recurring one in the history of robotics and computer vision. Thereby, it has been addressed using a variety of sensor setups and algorithms. A calibration process may involve recovering *intrinsic*, e.g. focal length for a camera, and *extrinsic* parameters, i.e. the rigid transformation between the sensor's coordinate system and a reference coordinate system. For the former, one could devise a naive approach where the parameters are accurately determined during the manufacturing process. Similarly, the transformation could be retrieved by means of some measuring instrument. However, the disadvantages of such purely engineered methods are manifold. Apart from their impracticality, it can be nearly impossible to reach a satisfying accuracy and thus hinder the proper use of the sensor in a robotic system. Furthermore, external factors such as temperature variations or mechanical shocks may seriously bias a factory calibration. Therefore, much efforts have been dedicated over the years to develop algorithms for calibration of systems in the field.

The use of a known calibration pattern such as a checkerboard coupled with nonlinear regression has become the most popular method in computer vision during the last decade. It has been deployed both for intrinsic camera calibration [4] and extrinsic calibration between heterogeneous sensors [5]. While being relatively efficient, this procedure still requires expert knowledge to reach a good level of accuracy. It can also be quite inconvenient on a mobile platform requiring frequent recalibration.

In an effort to automate the process in the context of mobile robotics, several authors have included the calibration problem in a state-space estimation framework, either with filtering [6] or smoothing [7] techniques. Filtering techniques based on the Kalman filter are appealing due to their inherently online nature. However, in case of nonlinear systems, smoothing techniques based on iterative optimization are usually superior in terms of accuracy. While building on this latter method, our approach attempts to reduce its computational load and copes with degenerate cases.

More recently, Brookshire and Teller [8] have carried out a formal observability analysis in order to identify degenerate paths of the calibration run. In contrast to the method presented in this paper, their approach still expects that the robot travels along non-degenerate paths.

A last class of methods relies on an energy function to be minimized. For instance, Levinson and Thrun [9] have defined an energy function based on surfaces and Sheehan *et al.* [10] on an information theoretic quantity measuring point cloud quality.

To the best of the authors' knowledge, little research has been devoted to efficiently deal with degenerate cases frequently occurring during calibration. The majority of the authors indeed assumes that their optimization routine runs on well-behaved data. As demonstrated in the next sections, this can be a critical point in real-world scenarios. In contrast, our approach does not require any of these assumptions and is suitable for online, autonomous, and long-term operations on various platforms and sensors.

III. Model

In this section, we will first state our problem in a probabilistic manner and present the mathematical groundings of our method. A practical algorithmic sketch will conclude the presentation.

A. Problem Formulation

In the following, we borrow the formalism of the probabilistic discrete-time Simultaneous Localization and Mapping (SLAM) model [11]. For the sake of clarity, we consider here a robot with a single sensor observing a known number of landmarks at each timestep. Furthermore, we assume the correspondences between sensor's measurements and landmarks are known. While this latter assumption involves data association techniques which go beyond the scope of this paper, the extension to multiple sensors is straightforward.

Let $\mathcal{X} = \{\mathbf{x}_{0:K}\}$ be a set of *latent* random variables (LRV) representing robot states up to timestep K, $\mathcal{U} = \{\mathbf{u}_{1:K}\}$ a set of *observable* random variables (ORV) representing measured control inputs, $\mathcal{L} = \{\boldsymbol{\ell}_{1:N}\}$ a set of LRV representing N landmark positions, $\mathcal{Z} = \{\mathbf{z}_{1_{1:N}:K_{1:N}}\}$ a set of ORV representing $K \times N$ landmark measurements, and $\boldsymbol{\Theta}$ an LRV representing

the calibration parameters of the robot's sensor. The goal of the calibration procedure is to compute the posterior marginal distribution of $\boldsymbol{\Theta}$ given all the measurements up to timestep K,

$$p(\boldsymbol{\Theta} \mid \mathcal{U}, \mathcal{Z}) = \int_{\mathcal{X},\mathcal{L}} p(\boldsymbol{\Theta}, \mathcal{X}, \mathcal{L} \mid \mathcal{U}, \mathcal{Z}). \tag{1}$$

Following [11], the full joint posterior on the right-hand side of (1) may further be factorized into

$$\begin{aligned} &p(\boldsymbol{\Theta}, \mathcal{X}, \mathcal{L} \mid \mathcal{U}, \mathcal{Z}) \propto \\ &p(\boldsymbol{\Theta}, \mathbf{x}_0, \mathcal{L}) \prod_{k=1}^{K} p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k) \prod_{k=1}^{K} \prod_{i=1}^{N} p(\mathbf{z}_{k_i} \mid \mathbf{x}_k, \boldsymbol{\ell}_i, \boldsymbol{\Theta}). \end{aligned} \tag{2}$$

We may then approximate (2) with a normal distribution whose mean, $\boldsymbol{\mu}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}$, and covariance, $\boldsymbol{\Sigma}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}$, have to be estimated. To this end, we first derive a Maximum a Posteriori (MAP) estimator for the mean,

$$\begin{aligned} \hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}} &= \arg\max_{\boldsymbol{\Theta},\mathcal{X},\mathcal{L}} p(\boldsymbol{\Theta}, \mathcal{X}, \mathcal{L} \mid \mathcal{U}, \mathcal{Z}) \\ &= \arg\min_{\boldsymbol{\Theta},\mathcal{X},\mathcal{L}} -\log p(\boldsymbol{\Theta}, \mathcal{X}, \mathcal{L} \mid \mathcal{U}, \mathcal{Z}). \end{aligned} \tag{3}$$

When setting the prior $p(\boldsymbol{\Theta}, \mathbf{x}_0, \mathcal{L})$ to a uniform distribution, (3) becomes a Maximum Likelihood (ML) estimator.

We shall further refine our problem by defining *motion* and *observation* models,

$$\begin{aligned} \mathbf{x}_k &= \mathbf{h}(\mathbf{x}_{k-1}, \mathbf{u}_k, \mathbf{w}_k), \\ \mathbf{z}_{k_i} &= \mathbf{g}(\mathbf{x}_k, \boldsymbol{\ell}_i, \boldsymbol{\Theta}, \mathbf{n}_k), \end{aligned} \tag{4}$$

where

$$\begin{aligned} \mathbf{w}_k &\sim \mathcal{N}(\mathbf{0}, \mathbf{W}_k), \text{ and} \\ \mathbf{n}_k &\sim \mathcal{N}(\mathbf{0}, \mathbf{N}_k) \end{aligned} \tag{5}$$

are normally distributed process and observation noise variables, with known covariance $\mathbf{W}_k$ and $\mathbf{N}_k$. Although, the functions $\mathbf{h}(\cdot)$ and $\mathbf{g}(\cdot)$ might be nonlinear in their parameters, we can approximate $p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k)$ and $p(\mathbf{z}_{k_i} \mid \mathbf{x}_k, \boldsymbol{\ell}_i, \boldsymbol{\Theta})$ as normal distributions through linearization.

B. Least Squares Solution

In case of linear motion and observation models, there exists a closed-form solution to (3) based on the *least squares* method due to the normally distributed noise variables. In the other case, one can resort to nonlinear least squares methods that iteratively solve a linearized version of the problem. In the following, we employ the *Gauss-Newton* algorithm [12] for this purpose.

From (3) and the approximation that the densities involved are normal, we can turn the MAP problem into the minimization of a sum of squared error terms. Nonlinear optimization techniques start with an initial guess, $\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}$, and refine it iteratively with $\delta\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}$ until convergence. $\delta\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}$ is chosen in such a way that it minimizes a quadratic approximation of the objective function around $\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}$. Gauss-Newton method only requires the stacked Jacobian matrix of the error terms, $\mathbf{H}$. In block matrix form, the update takes the form

$$(\mathbf{H}^T\mathbf{T}^{-1}\mathbf{H})\delta\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}} = -\mathbf{H}^T\mathbf{T}^{-1}\mathbf{e}(\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}), \tag{6}$$

where $\mathbf{T}$, the error covariance matrix, is built from diagonal blocks of $\mathbf{W}_k$ and $\mathbf{N}_k$, and $\mathbf{e}(\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}})$ is the error evaluated at the current estimate $\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}$.

At convergence of the algorithm to the estimate $\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}$, the quantity $\mathbf{H}^T\mathbf{T}^{-1}\mathbf{H}$ is the *Fisher Information Matrix* (FIM) and the inverse of the estimate covariance matrix, $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}$.

If we let $\mathbf{T}^{-1} = \mathbf{L}^T\mathbf{L}$ be the Cholesky decomposition of the error covariance matrix, (6) can be rewritten as

$$(\mathbf{L}\mathbf{H})^T(\mathbf{L}\mathbf{H})\delta\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}} = -(\mathbf{L}\mathbf{H})^T\mathbf{L}\mathbf{e}(\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}), \tag{7}$$

which we can recognize as the *normal equations* of the linear system

$$(\mathbf{L}\mathbf{H})\delta\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}} = -\mathbf{L}\mathbf{e}(\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}). \tag{8}$$

Thus, instead of solving (6), we can take advantage of matrix decompositions to solve (8) directly. Not only can this be more computationally efficient, the use of a *rank-revealing* decomposition allows the estimation of the numerical rank of the FIM, and consequently (as described below) provides information about the numerical observability of the parameters for a given batch of data.

Let $\mathbf{L}\mathbf{H}$ be of size $m \times n$ with the following thin[3] Singular Value Decomposition (*SVD*) decomposition [13],

$$\mathbf{L}\mathbf{H} = \mathbf{U}_n\mathbf{S}_n\mathbf{V}_n^T, \tag{9}$$

where $\mathbf{U}_n$ is an $m \times n$ matrix with orthogonal columns, $\mathbf{S}_n = \text{diag}(\sigma_1, \cdots, \sigma_n)$, $\mathbf{V}_n$ is an $n \times n$ matrix with orthogonal columns, and σ_i are the singular values of $\mathbf{L}\mathbf{H}$. From (9) and using the orthogonality of $\mathbf{U}_n$ and $\mathbf{V}_n$, we can solve (6) as

$$\delta\hat{\boldsymbol{\mu}}^{(SVD)}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}} = -\mathbf{V}_n\mathbf{S}_n^{-1}\mathbf{U}_n^T\mathbf{L}\mathbf{e}(\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}). \tag{10}$$

Although useful for illustrating the concepts of our approach, the SVD decomposition can be computationally demanding for large matrices. In practice, we therefore use the thin *QR* decomposition [13] of $\mathbf{L}\mathbf{H}$,

$$\mathbf{L}\mathbf{H}\boldsymbol{\Pi} = \mathbf{Q}_n\mathbf{R}_n, \tag{11}$$

where $\boldsymbol{\Pi}$ is a permutation matrix, $\mathbf{Q}_n$ an $m \times n$ matrix with orthogonal columns, and $\mathbf{R}_n$ an $n \times n$ upper triangular matrix. In standard QR decomposition, $\boldsymbol{\Pi}$ is the identity matrix, otherwise it reflects column pivoting. From (11) and using the orthogonality of $\mathbf{Q}_n$, (6) can be expressed as

$$\mathbf{R}_n\boldsymbol{\Pi}^T\delta\hat{\boldsymbol{\mu}}^{(QR)}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}} = -\mathbf{Q}_n^T\mathbf{L}\mathbf{e}(\hat{\boldsymbol{\mu}}_{\boldsymbol{\Theta}\mathcal{X}\mathcal{L}}), \tag{12}$$

which, due the upper triangular form of $\mathbf{R}_n$, can be easily solved by *back substitution*.

[3]Only the first n out of m columns of $\mathbf{U}$ are computed in a thin SVD decomposition.

The $\mathbf{R}_n$ matrix can also be used to compute the FIM and its inverse, the estimate covariance, $\hat{\mathbf{\Sigma}}_{\mathbf{\Theta}\mathcal{X}\mathcal{L}}$. If we drop the permutation matrix for clarity, the FIM simply becomes $\mathbf{R}_n^T\mathbf{R}_n$ and there exists an efficient algorithm [14] for recovering any elements of the covariance estimate without inverting the whole FIM.

At the convergence of the Gauss-Newton optimization, we are thus left with the estimates $\hat{\boldsymbol{\mu}}_{\mathbf{\Theta}\mathcal{X}\mathcal{L}}$ and $\hat{\mathbf{\Sigma}}_{\mathbf{\Theta}\mathcal{X}\mathcal{L}}$ of the normal distribution $p(\mathbf{\Theta}, \mathcal{X}, \mathcal{L} \mid \mathcal{U}, \mathcal{Z})$. In order to solve (1), we can employ the marginalization property of normal distributions [15]. If we express the estimates in the partitioned form,

$$\hat{\boldsymbol{\mu}}_{\mathbf{\Theta}\mathcal{X}\mathcal{L}} = \begin{pmatrix} \hat{\boldsymbol{\mu}}_{\mathcal{X}} \\ \hat{\boldsymbol{\mu}}_{\mathcal{L}} \\ \hat{\boldsymbol{\mu}}_{\mathbf{\Theta}} \end{pmatrix}, \qquad \hat{\mathbf{\Sigma}}_{\mathbf{\Theta}\mathcal{X}\mathcal{L}} = \begin{pmatrix} \hat{\mathbf{\Sigma}}_{\mathcal{X}\mathcal{X}} & \hat{\mathbf{\Sigma}}_{\mathcal{X}\mathcal{L}} & \hat{\mathbf{\Sigma}}_{\mathcal{X}\mathbf{\Theta}} \\ \hat{\mathbf{\Sigma}}_{\mathcal{L}\mathcal{X}} & \hat{\mathbf{\Sigma}}_{\mathcal{L}\mathcal{L}} & \hat{\mathbf{\Sigma}}_{\mathcal{L}\mathbf{\Theta}} \\ \hat{\mathbf{\Sigma}}_{\mathbf{\Theta}\mathcal{X}} & \hat{\mathbf{\Sigma}}_{\mathbf{\Theta}\mathcal{L}} & \hat{\mathbf{\Sigma}}_{\mathbf{\Theta}\mathbf{\Theta}} \end{pmatrix}, \quad (13)$$

then $p(\mathbf{\Theta} \mid \mathcal{U}, \mathcal{Z}) \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_{\mathbf{\Theta}}, \hat{\mathbf{\Sigma}}_{\mathbf{\Theta}\mathbf{\Theta}})$. We can thus extract $\hat{\boldsymbol{\mu}}_{\mathbf{\Theta}}$ from $\hat{\boldsymbol{\mu}}_{\mathbf{\Theta}\mathcal{X}\mathcal{L}}$ very efficiently, and by placing $\mathbf{\Theta}$ to the right of the Jacobian matrix $\mathbf{H}$ and using [14], $\hat{\mathbf{\Sigma}}_{\mathbf{\Theta}\mathbf{\Theta}}$ can be computed in $O(l)$, where $l = \text{card}(\mathbf{\Theta})$.

C. Truncated SVD and QR Solutions

There exists a solution to (6) iff the FIM is invertible, i.e., it is *full rank*[4]. Jauffret [16] clearly articulates the link between the rank of the FIM and observability of the parameters being estimated. In the context of calibration, a singular FIM corresponds to some unobservable directions in the parameter space given the current set of observations. Classical observability analysis, for example the widely used method of Hermann and Krener [17] (c.f. [18] or [19]), proves what we will call *structural observability*—that there exists some dataset for which the parameters are observable—it does not guarantee that the parameters are observable for any dataset.

In real-world scenarios, where data is contaminated by noise, the FIM might appear to be full rank although it is actually *rank-deficient*. We can illustrate this with the matrix

$$\mathbf{A} = \begin{pmatrix} 0.9999 & 1.9999 & 3.0014 \\ 4.0007 & 5.0015 & 6.0007 \\ 6.9998 & 8.0014 & 8.9988 \end{pmatrix}. \quad (14)$$

Although $\mathbf{A}$ is technically full rank, it was constructed from a structurally rank-deficient matrix (the second row is a linear combination of the first and the last row) with added Gaussian noise.

Using SVD decomposition, we can identify a numerically rank-deficient matrix by analyzing its singular values [20] and consequently the numerical observability of the system [21] [22]. The *numerical rank* r of a matrix is defined as the index of its smallest singular value σ_r larger than a user-defined tolerance ϵ, i.e.,

$$r = \arg\max_i \sigma_i \geq \epsilon. \quad (15)$$

[4]The rank of a matrix is the maximum number of linearly independent column or row vectors. A matrix with m rows and n columns has full rank when its rank is equal to $\min(m, n)$.

For example, the singular values of $\mathbf{A}$ are approximately 16.8, 1.1, and 0.0001. From (10), we can see that if some of the σ_i are close to zero, i.e. $r < n$, the update vector $\delta\hat{\boldsymbol{\mu}}^{(SVD)}_{\mathbf{\Theta}\mathcal{X}\mathcal{L}}$ will be large (scaled by $1/\sigma_i$) and sometimes lead to a divergence of the solution. In order to cope with this issue, we can approximate $\mathbf{LH}$ with a lower-rank matrix yielding the *Truncated* SVD (TSVD) [23] solution,

$$\delta\hat{\boldsymbol{\mu}}^{(TSVD)}_{\mathbf{\Theta}\mathcal{X}\mathcal{L}} = -\mathbf{V}_r\mathbf{S}_r^{-1}\mathbf{U}_r^T\mathbf{Le}(\hat{\boldsymbol{\mu}}_{\mathbf{\Theta}\mathcal{X}\mathcal{L}}), \quad (16)$$

with $\mathbf{U}_r$ an $m \times r$ matrix, $\mathbf{S}_r = \text{diag}(\sigma_1, \cdots, \sigma_r)$, $\mathbf{V}_r$ an $n \times r$ matrix.

It is worth noting that TSVD applies a *sharp* filter to the system, whereas *regularization* methods such as Tikhonov or ridge regression work as a *smooth* filter. As shown in [24], it nevertheless produces similar results at lower computational costs when the matrix has a well-determined numerical rank, i.e., a clearly defined jump in the singular values spectrum.

A similar approach can be derived using *rank-revealing* QR decomposition [25] yielding the Truncated QR (TQR) method [26]. The strategy is to apply QR factorization with column pivoting in order to reveal the rank and to apply a cutoff on the diagonal of the $\mathbf{R}$ matrix. It can further be noted that the $\mathbf{Q}$ matrix need not be explicitly constructed, but instead returned in the form of *Householder* reflections that can be applied to the right-hand side of the linear system.

According to the discussion in [13], *column scaling* may improve the conditioning of the system, i.e. the ratio between the largest and the smallest singular value, and hence the stability of the solution. Within our framework, scaling is also relevant for merging quantities with miscellaneous magnitudes and helps in setting a rank tolerance ϵ that is widely applicable. We define the scaling matrix as

$$\mathbf{G} = \text{diag}\left\{ \frac{1}{\|\mathbf{LH}(:,1)\|}, \cdots, \frac{1}{\|\mathbf{LH}(:,n)\|} \right\}, \quad (17)$$

where $\|\cdot\|$ denotes the column vector norm and $\mathbf{H}(:,i)$ the i-th column of $\mathbf{H}$, and compute a solution to the scaled system,

$$\mathbf{LHG}\delta\hat{\boldsymbol{\mu}}^{(s)}_{\mathbf{\Theta}\mathcal{X}\mathcal{L}} = -\mathbf{Le}(\hat{\boldsymbol{\mu}}_{\mathbf{\Theta}\mathcal{X}\mathcal{L}}). \quad (18)$$

The unscaled solution is finally expressed as $\mathbf{G}\delta\hat{\boldsymbol{\mu}}^{(s)}_{\mathbf{\Theta}\mathcal{X}\mathcal{L}}$.

D. Selecting Informative Measurements

Since we are mainly interested in calibrating our sensor and thus in the posterior $p(\mathbf{\Theta} \mid \mathcal{U}, \mathcal{Z})$, we do not need to consider all the measurements. Let us define a partition of the measurements $\mathcal{D}_1 = \{\mathbf{u}_{1:i}, \mathbf{z}_{1_{1:N}:i_{1:N}}\}$ and $\mathcal{D}_2 = \{\mathbf{u}_{i+1:K}, \mathbf{z}_{i+1_{1:N}:K_{1:N}}\}$ with $i < K$. Using information theory, we can quantify the information gain when estimating $\mathbf{\Theta}$ with $\mathcal{D}_1$ alone or with $\mathcal{D}_1 \cup \mathcal{D}_2$.

The *mutual information* (MI) [27] between two random variables $\mathbf{X}$ and $\mathbf{Y}$ is defined as

$$I(\mathbf{X}; \mathbf{Y}) = \int_{\mathbf{X}} \int_{\mathbf{Y}} p(x, y) \log \frac{p(x, y)}{p(x)p(y)}, \quad (19)$$

where $p(x, y)$ is the joint density of $\mathbf{X}$ and $\mathbf{Y}$, and $p(x)$ and $p(y)$ their marginal densities. The MI measures the amount

of information $\mathbf{X}$ and $\mathbf{Y}$ share or, in other words, quantifies the reduction of uncertainty in $\mathbf{X}$ when knowing $\mathbf{Y}$.

Before deriving the MI for our purpose, it is worth recalling another property of normal distributions [15]. If we define $\mathbf{\Theta}_1 = \mathbf{\Theta} \mid \mathcal{D}_1$ and consider the joint normal distribution $p(\mathbf{\Theta}_1, \mathcal{D}_2)$ with parameters

$$\boldsymbol{\mu}_{\mathbf{\Theta}_1\mathcal{D}_2} = \begin{pmatrix} \boldsymbol{\mu}_{\mathbf{\Theta}_1} \\ \boldsymbol{\mu}_{\mathcal{D}_2} \end{pmatrix}, \; \mathbf{\Sigma}_{\mathbf{\Theta}_1\mathcal{D}_2} = \begin{pmatrix} \mathbf{\Sigma}_{\mathbf{\Theta}_1\mathbf{\Theta}_1} & \mathbf{\Sigma}_{\mathbf{\Theta}_1\mathcal{D}_2} \\ \mathbf{\Sigma}_{\mathcal{D}_2\mathbf{\Theta}_1} & \mathbf{\Sigma}_{\mathcal{D}_2\mathcal{D}_2} \end{pmatrix}, \tag{20}$$

then $p(\mathbf{\Theta}_1 \mid \mathcal{D}_2)$ is a normal distribution with parameters

$$\begin{aligned} \boldsymbol{\mu}_{\mathbf{\Theta}_1|\mathcal{D}_2} &= \boldsymbol{\mu}_{\mathbf{\Theta}_1} + \mathbf{\Sigma}_{\mathbf{\Theta}_1\mathcal{D}_2}\mathbf{\Sigma}^{-1}_{\mathcal{D}_2\mathcal{D}_2}(\mathcal{D}_2 - \boldsymbol{\mu}_{\mathcal{D}_2}) \\ \mathbf{\Sigma}_{\mathbf{\Theta}_1|\mathcal{D}_2} &= \mathbf{\Sigma}_{\mathbf{\Theta}_1\mathbf{\Theta}_1} - \mathbf{\Sigma}_{\mathbf{\Theta}_1\mathcal{D}_2}\mathbf{\Sigma}^{-1}_{\mathcal{D}_2\mathcal{D}_2}\mathbf{\Sigma}_{\mathcal{D}_2\mathbf{\Theta}_1}. \end{aligned} \tag{21}$$

Furthermore, $\mathbf{\Sigma}_{\mathbf{\Theta}_1|\mathcal{D}_2}$ can be recognized as the *Schur complement* of the joint covariance matrix $\mathbf{\Sigma}_{\mathbf{\Theta}_1\mathcal{D}_2}$.

Following an argument of [28], the MI between $\mathbf{\Theta}_1$ and $\mathcal{D}_2$ can then be computed as

$$\begin{aligned} I(\mathbf{\Theta}_1; \mathcal{D}_2) &= \frac{1}{2} \log \frac{|\mathbf{\Sigma}_{\mathbf{\Theta}_1\mathbf{\Theta}_1}|}{|\mathbf{\Sigma}_{\mathbf{\Theta}_1\mathbf{\Theta}_1} - \mathbf{\Sigma}_{\mathbf{\Theta}_1\mathcal{D}_2}\mathbf{\Sigma}^{-1}_{\mathcal{D}_2\mathcal{D}_2}\mathbf{\Sigma}_{\mathcal{D}_2\mathbf{\Theta}_1}|} \\ &= \frac{1}{2} \log \frac{|\mathbf{\Sigma}_{\mathbf{\Theta}_1\mathbf{\Theta}_1}|}{|\mathbf{\Sigma}_{\mathbf{\Theta}_1|\mathcal{D}_2}|}, \end{aligned} \tag{22}$$

where $|\cdot|$ denotes the matrix determinant and $\mathbf{\Sigma}_{\mathbf{\Theta}_1|\mathcal{D}_2}$ is the covariance estimate of $\mathbf{\Theta}$ using $\mathcal{D}_1$ and $\mathcal{D}_2$, which is computed by Gauss-Newton optimization. Thus, using (22), we can measure the amount of information $\mathcal{D}_2$ conveys to our current estimate $\mathbf{\Theta} \mid \mathcal{D}_1$.

E. Algorithm

Thus far, we have delivered a formal introduction to the probabilistic groundings of our self-supervised calibration approach. This section will be dedicated to a practical online implementation of our algorithm.

The proposed calibration method is sketched in Alg. 1. Defining $\mathcal{D}^{info}$ as the set of *informative* measurements at time t, we collect a measurement batch $\mathcal{D}^{new} = \{\mathbf{u}_{t:t+k}, \mathbf{z}_{t_{1:N}:t+k_{1:N}}\}$ during k timesteps and compute $p(\mathbf{\Theta} \mid \mathcal{D}^{info}, \mathcal{D}^{new})$ using Gauss-Newton method with TQR updates and threshold ϵ. In a second step, if $I(\mathbf{\Theta} \mid \mathcal{D}^{info}; \mathcal{D}^{new})$ is larger than a user-defined threshold λ, $\mathcal{D}^{new}$ is added to $\mathcal{D}^{info}$ and the estimate of $\mathbf{\Theta}$ is updated.

Given the structure of the model (2), the Jacobian matrix $\mathbf{H}$ will be sparsely populated. Therefore, in order to obtain a tractable and scalable method, we adopt sparse matrices algorithms and data structures. The memory and computational costs are then $O(\text{card}(\mathcal{X}) \times \text{card}(\mathcal{L}))$.

Algorithm 1: calibrateSensor()

Input: Initial guesses $\hat{\mathbf{\Theta}}^{(0)}$, $\hat{\mathbf{x}}_0^{(0)}$, $\hat{\mathcal{L}}^{(0)}$
Input: Motion $\mathbf{h}(\cdot)$ and observation $\mathbf{g}(\cdot)$ models
Input: Batch size k, TQR threshold ϵ, MI threshold λ
Output: $\hat{\boldsymbol{\mu}}_{\mathbf{\Theta}}$ and $\hat{\mathbf{\Sigma}}_{\mathbf{\Theta}\mathbf{\Theta}}$

```
D^info ← ∅ ;
t ← 1 ;
while calibrate do
    // Collecting measurements ;
    t^init ← t ;
    D^new ← ∅ ;
    while t < t^init + k do
        x̂_t^(0) ← h(x̂_{t−1}^(0), u_t, 0) ;
        D^new ← D^new ∪ {u_t, z_{t_{1:N}}} ;
        t ← t + 1 ;
    end
    // TQR optimization with threshold ε ;
    μ̂_{ΘXL} ← arg max_{Θ,X,L} p(Θ, X, L | D^info, D^new) ;
    // Marginalization ;
    μ̂_Θ ← μ̂_{ΘXL} ;
    Σ̂_{ΘΘ} ← Σ̂_{ΘXL} ;
    // MI decision ;
    if I(Θ | D^info; D^new) > λ then
        D^info ← D^info ∪ D^new ;
        Θ ~ N(μ̂_Θ, Σ̂_{ΘΘ}) ;
    end
end
```

IV. Experiments

In order to evaluate and validate the approach proposed in this paper, we have conducted experiments on simulated and real-world data. We shall first start with the description of the experimental conditions and then demonstrate the performance of our algorithm, along with some comparisons against existing methods.

A. Experimental Setup

Our method has been fully implemented in MATLAB and we have used the SuiteSparseQR [29] package for performing sparse matrix operations.

Although our eventual goal is full, self-supervised calibration for an autonomous vehicle system with multiple heterogeneous sensors, the experiments in this paper are based on a realistic, but somewhat simplified robotic system. This allows us to precisely control the observability properties in simulation and test the behavior of our algorithm.

Fig. 2 depicts our experimental setup. The platform is a differential drive mobile robot equipped with a range sensor delivering range and bearing measurements. The calibration parameters of the range sensor consist in the transformation of its coordinate system to the robot's coordinate system. The platform is further endowed with wheel odometers outputting translational and rotational speeds. While navigating on the plane, the robot observes a known number of landmarks through its range sensor. Both the range sensor and the odometry produce data at 10 Hz.

More formally, we adopt the following motion and observation models

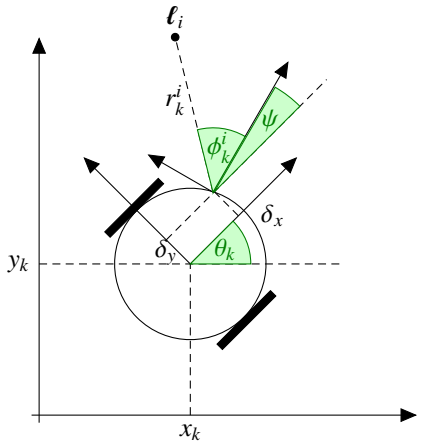

Fig. 2. Experimental setup. A 2D robot moving in a plane and observing landmarks with a range sensor providing range and bearing angle measurements. The calibration process needs to find the 2D transformation between the robot's coordinate system and the sensor's coordinate system.

$$\begin{aligned}
\underbrace{\begin{pmatrix} x_k \\ y_k \\ \theta_k \end{pmatrix}}_{\mathbf{x}_k} &= \underbrace{\begin{pmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{pmatrix} + T \begin{pmatrix} \cos\theta_{k-1} & 0 \\ \sin\theta_{k-1} & 0 \\ 0 & 1 \end{pmatrix} \left(\begin{pmatrix} v_k \\ w_k \end{pmatrix} + \mathbf{w}_k \right)}_{\mathbf{h}(\mathbf{x}_{k-1}, \mathbf{u}_k, \mathbf{w}_k)} \\
a &= x_i - x_k - \delta_x \cos\theta_k + \delta_y \sin\theta_k \\
b &= y_i - y_k - \delta_x \sin\theta_k - \delta_y \cos\theta_k \\
\underbrace{\begin{pmatrix} r_k^i \\ \phi_k^i \end{pmatrix}}_{\mathbf{z}_{k_i}} &= \underbrace{\begin{pmatrix} \sqrt{a^2 + b^2} \\ \text{atan2}(b, a) - \theta_k - \psi \end{pmatrix} + \mathbf{n}_k}_{\mathbf{g}(\mathbf{x}_k, \boldsymbol{\ell}_i, \boldsymbol{\Theta}, \mathbf{n}_k)},
\end{aligned} \tag{23}$$

where $\mathbf{x}_k = [x_k \; y_k \; \theta_k]^T$ denotes the robot pose at timestep k, T the sampling period, $\mathbf{u}_k = [v_k \; w_k]^T$ the measured translational and rotational speeds, $\mathbf{z}_{k_i} = [r_k^i \; \phi_k^i]^T$ the range and bearing observation of landmark i with pose $\boldsymbol{\ell}_i = [x_i \; y_i]^T$, $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{W}_k)$ with $\mathbf{W}_k = \text{diag}(\sigma_v^2, \sigma_w^2)$, $\mathbf{n}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{N}_k)$ with $\mathbf{N}_k = \text{diag}(\sigma_r^2, \sigma_\phi^2)$, and $\boldsymbol{\Theta} = [\delta_x \; \delta_y \; \psi]^T$ the range sensor's calibration parameters.

Throughout our experiments, we have used a non-informative prior $p(\boldsymbol{\Theta}, \mathbf{x}_0, \mathcal{L})$, i.e., a uniform distribution. The use of priors is still subject to controversial discussions [30] between Bayesian and non-Bayesian statisticians. Here, we argue that everything should come from the data itself and not from some subjective prior information that could bias the inference.

Our algorithm requires only 3 free parameters, namely the rank threshold ϵ, the batch size k, and the MI threshold λ. Optimally, k should be inferred from the dynamics of the system. While a large k induces storage of uninformative measurements, a small k leads to useless runs of optimization and makes it difficult to discover informative sequences of measurements. In this setup, we have used a batch size of $k = 100$ (ten seconds of data). Concerning the MI threshold, a small λ will keep most of the measurements and a large λ will ignore them all. We have set this value to $\lambda = 0.5$ [bits] in our experiments. The ϵ parameter shall be discussed below.

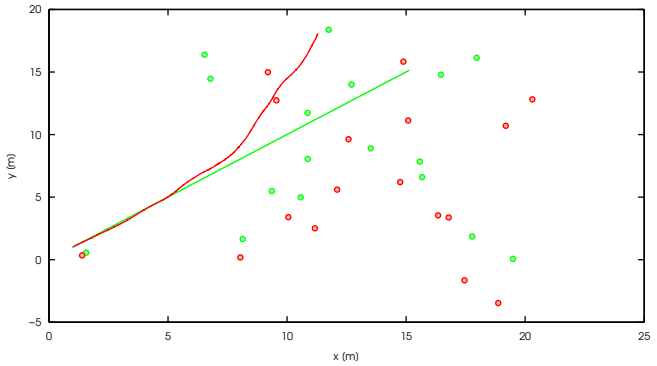

Fig. 3. Straight path example. The green line represents the ground truth path, the red line the integrated odometry path, the green circles the ground truth landmark positions, and the red circles the guessed landmark positions.

B. Simulated Data

In our simulation environment, we can generate various paths for the robot, along with corresponding sensor measurements, and thus analyze the behavior of multiple algorithms, especially in degenerate cases. We have created an environment with $N = 17$ landmarks uniformly distributed on a $20m \times 20m$ grid. We have set the noise parameters empirically to $\sigma_v^2 = 4.4 \times 10^{-3}$, $\sigma_w^2 = 8.2 \times 10^{-2}$, $\sigma_r^2 = 9.0036 \times 10^{-4}$, and $\sigma_\phi^2 = 6.7143 \times 10^{-4}$. The calibration parameters are fixed at $\delta_x = 0.219$ [m], $\delta_y = 0.1$ [m], and $\psi = \pi/4$ [rad].

In a first effort, we want to support the claims of Sec. III-C with a representative example. We simulated the robot driving along a straight path as shown in Fig. 3. Intuitively, the problem is structurally unobservable. Indeed, it has 5 unobservable parameters, 3 corresponding to the global pose of the map and trajectory (as no global measurements or prior are included) and 2 for the calibration offset variables δ_x and δ_y. In the absence of noise, 5 of the singular values of the Jacobian matrix are zero up to machine precision, i.e., a structural rank deficiency. However, when noise is added, only 3 singular values remain at 0 for the same problem. From the integrated odometry path in Fig. 3, the calibration parameters appear indeed as observable and a naive algorithm without regularization will thus wrongly optimize. Fig. 4 shows the singular values in these two cases and the related concept in the QR decomposition. With our TSVD/TQR method, we can deal with this issue by setting an adequate ϵ threshold that will recover the correct rank deficiency and therefore only optimize the observable parts. The threshold is application-specific and should be a function of the system noise. Obviously, at a certain level of noise, gaps in the singular values spectrum become indistinguishable. In our setup, we have determined the threshold empirically from the scaled system (18) and set it to $\epsilon = 0.013$.

To conclude this section on simulated data, we will analyze the performance of different algorithms, namely an Extended

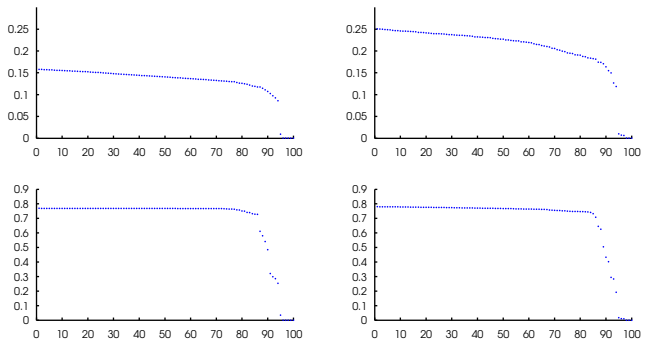

Fig. 4. Straight path example analysis. The first column shows the singular values and the diagonal elements of the **R** matrix from the QR decomposition in the noise-free case, and the second column in the noisy case. Only the 100 lower values from the scaled system (18) are plotted for visualization purpose.

Kalman Filter (EKF) similar to [6], a standard batch non-linear least squares (LS) method without regularization [7], and our TQR-MI approach, on a sine wave path with varying amplitude such that the offset parameters range from unobservable (straight path or zero amplitude) to fully observable (higher amplitude). To get significant statistical results, we have repeated the experiment 100 times with the same initial conditions ($\delta_x^{(0)} = 0.23$ [m], $\delta_y^{(0)} = 0.11$ [m], $\psi^{(0)} = 0.8$ [rad]) and amplitudes from 0 [m] to 5 [m] with steps of 0.5 [m] over 5000 timesteps (500 [s] of data). Fig. 5 displays the result of this experiment. At amplitude 0, we clearly witness that the LS and EKF estimators fail to yield reasonable offset parameters. With our TQR-MI method, these parameters remain at their initial guess until an amplitude of 1 and only the angle gets optimized. As the amplitude grows, our method becomes comparable to the LS estimator, while using fewer measurements. In these experiments, the EKF is fed by a vague initial prior and exhibits instability in case of unobservability. A more concentrated prior would improve the results at the cost of introducing potential bias.

C. Real-world Data

In order to validate our method on real-world data, we have used the "Lost in the Woods Dataset" provided with the courtesy of Tim Barfoot [31]. This dataset contains approximately 20 minutes of a robot driving amongst a forest of tubes which serve as landmarks. The ground truth comes from a motion capture system that tracks robot motion and tube locations. For the calibration parameters, we have only access to $\delta_x = 0.219$ [m] that was roughly measured with a tape. We assume the others are implicitly set to 0 ($\delta_y = 0$ [m] and $\psi = 0$ [rad]). Fig. 6 displays the qualitative result of our algorithm. Using only 10% of the measurements, we could recover accurate landmark positions, robot poses, and calibration parameters. For visualization, estimated landmarks and poses have been aligned to the ground truth using the algorithm from [32].

Since the ground truth calibration parameters are inaccurate, Tab. I compares our method against a standard least squares (LS) estimator that exploits all the measurements.

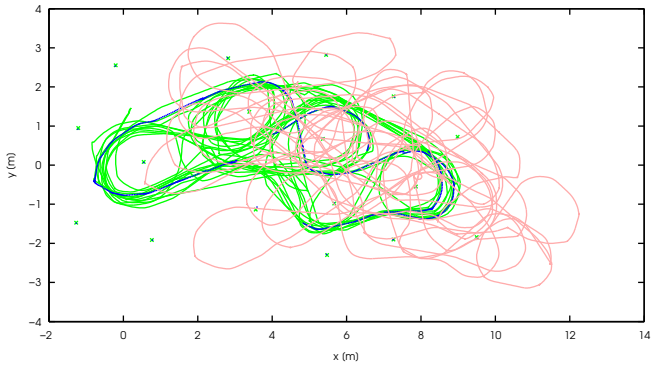

Fig. 6. Application of our algorithm on the "Lost in the Woods dataset". The green line represents the ground truth path, the red line the integrated odometry path, the green crosses the ground truth landmark positions, the blue points the estimated landmark positions, and the blue crosses the estimated robot poses. Our MI selection scheme picks only 10% of the measurements for the optimization.

TABLE I

COMPARISON OF A STANDARD LEAST SQUARES ESTIMATOR AND OUR METHOD ON THE "LOST IN THE WOODS DATASET".

	LS	**TQR-MI**
$\hat{\delta}_x$ [m]	0.2357	0.2344
$\hat{\delta}_y$ [m]	0.0031	0.0087
$\hat{\psi}$ [rad]	0.0804	0.0754
$\hat{\Sigma}_{\delta_x}$ [m^2]	0.0908×10^{-5}	0.1903×10^{-4}
$\hat{\Sigma}_{\delta_y}$ [m^2]	0.3917×10^{-5}	0.7243×10^{-4}
$\hat{\Sigma}_{\psi}$ [rad^2]	0.0286×10^{-5}	0.0582×10^{-4}
K	12609	1209

Our TQR-MI algorithm performs the optimization with $K = 1209$ out of 12609 timesteps. Although the calibration parameters are the same magnitude, the higher variances stem from the fewer number of considered measurements.

V. CONCLUSION

In this paper, we have presented a novel approach to the automatic calibration of mobile robot sensors. We have included the calibration parameters in a SLAM formulation and computed an MAP estimator using a Gauss-Newton algorithm. To cope with unobservability, we have employed truncated QR decomposition as a regularization method. For long-term and online operations, we have devised a mutual information selection scheme that solely captures informative measurements relevant to the calibration. Our algorithm has been thoroughly tested and validated through extensive simulated and real-world experiments on a 2D robot equipped with a laser range finder (LRF).

In the near future, we will consider the self calibration of multiple sensors (cameras, 3D/2D LRF, GPS/INS system, ...) mounted on an autonomous car and a C++ implementation. From a research point of view, we would like to further automate the selection of the free parameters. Lastly, we want to detect stepwise changes in the calibration parameters in order to trigger recalibration in case of mechanical shocks.

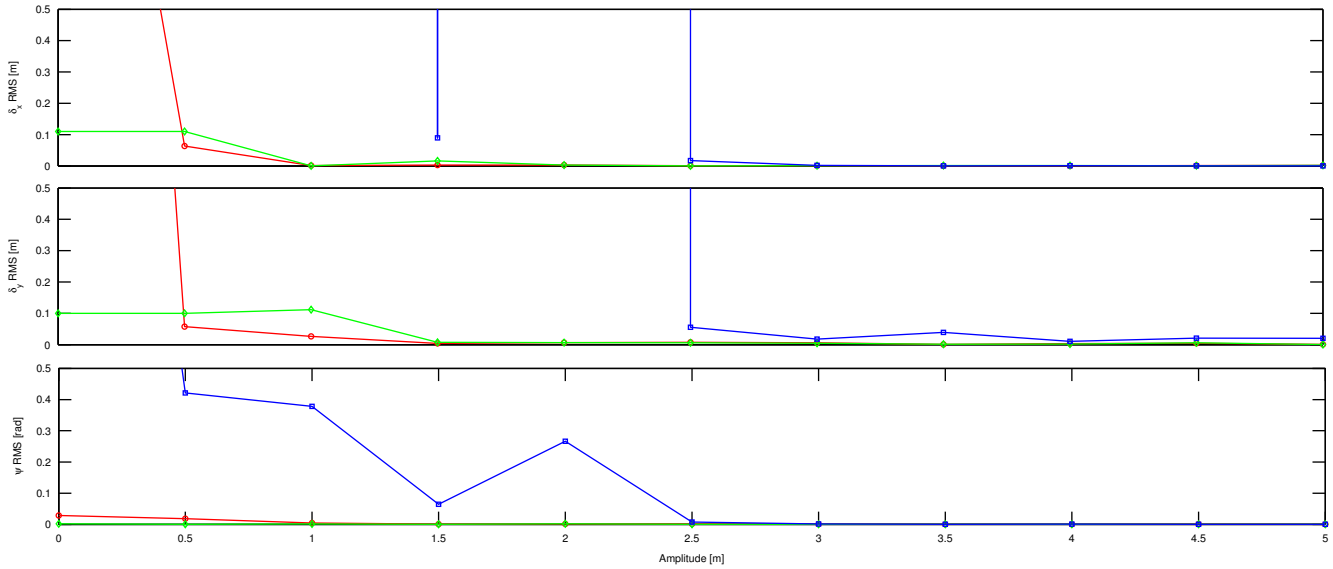

Fig. 5. Comparison of a least squares (red), EKF (blue), and our TQR-MI (green) estimator on a sine wave path of 5000 timesteps with varying amplitudes (0 to 5 [m] with steps of 0.5 [m]). For each amplitude and method, 100 simulations are performed and the RMS errors of the computed calibration parameters (δ_x, δ_y, and ψ) are reported. The plots are scaled to show the behavior of the algorithms near RMS = 0.

Acknowledgment

This work has partly been supported by the EC under FP7-269916-V-Charge.

References

[1] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*. Academic Press, 1970.

[2] P. Lébraly, E. Royer, O. Ait-Aider, and M. Dhome, "Calibration of non-overlapping cameras - application to vision-based robotics," in *Proc. 21st British Machine Vision Conf. (BMVC)*, 2010.

[3] J.-H. Kim and M. J. Chung, "Absolute motion and structure from stereo image sequences without stereo correspondence and analysis of degenerate cases," *Pattern Recognition*, vol. 39, pp. 1649–1661, 2006.

[4] P. F. Sturm and S. J. Maybank, "On plane-based camera calibration: A general algorithm, singularities, applications," in *Proc. 12th IEEE Conf. Comput. Vision and Pattern Recognition (CVPR)*, 1999.

[5] Q. Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder (improves camera calibration)," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2004.

[6] A. Martinelli, D. Scaramuzza, and R. Siegwart, "Automatic self-calibration of a vision system during robot motion," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, 2006.

[7] R. Kuemmerle, G. Grisetti, and W. Burgard, "Simultaneous calibration, localization, and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2011.

[8] J. Brookshire and S. Teller, "Extrinsic calibration from per-sensor egomotion," in *Proc. 8th Robot.: Sci. Syst. Conf. (RSS)*, 2012.

[9] J. Levinson and S. Thrun, "Unsupervised calibration for multi-beam lasers," in *Proc. 12th Int. Symp. Experimental Robotics (ISER)*, 2010.

[10] M. Sheehan, A. Harrison, and P. Newman, "Self-calibration for a 3D laser," *Int. J. Robot. Research (IJRR)*, vol. 31, pp. 675–687, 2012.

[11] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robot. Automat. Mag. (RAM)*, vol. 13, pp. 99–108, 2006.

[12] R. C. Aster, B. Borchers, and C. H. Thurber, *Parameter Estimation and Inverse Problems*. Academic Press, 2011.

[13] G. H. Golub and C. F. V. Loan, *Matrix Computations*. The Johns Hopkins University Press, 1996.

[14] M. Kaess and F. Dellaert, "Covariance recovery from a square root information matrix for data association," *Robot. Autonomous Syst. (RAS)*, vol. 57, pp. 1198–1210, 2009.

[15] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[16] C. Jauffret, "Observability and Fisher information matrix in nonlinear regression," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 43, pp. 756–759, 2007.

[17] R. Hermann and A. J. Krener, "Nonlinear controllability and observability," *IEEE Trans. Automat. Contr.*, vol. 22, pp. 728–740, 1977.

[18] J. Kelly and G. S. Sukhatme, "Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration," *Int. J. Robot. Research (IJRR)*, vol. 30, pp. 56–79, 2011.

[19] F. M. Mirzaei and S. I. Roumeliotis, "A Kalman filter-based algorithm for IMU-camera calibration: Observability analysis and performance evaluation," *IEEE Trans. Robot.*, vol. 24, pp. 1143 – 1156, 2008.

[20] P. C. Hansen, *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion*. Society for Industrial and Applied Mathematics (SIAM), 1998.

[21] B. P. Gibbs, *Advanced Kalman Filtering, Least-Squares and Modeling: A Practical Handbook*. Jon Wiley & Sons, 2011.

[22] L. K. Huat, Ed., *Industrial Robotics: Programming, Simulation and Applications*. Pro Literatur Verlag, 2006.

[23] J. M. Varah, "On the numerical solution of ill-conditioned linear systems with applications to ill-posed problems," *SIAM J. Numerical Anal.*, vol. 10, pp. 257–267, 1973.

[24] P. C. Hansen, "The truncated SVD as a method for regularization," *BIT*, vol. 27, pp. 534–553, 1987.

[25] Y. P. Hong and C.-T. Pan, "Rank-revealing QR factorizations and the singular value decomposition," *Mathematics of Computation*, vol. 58, pp. 213–232, 1992.

[26] T. Kitagawa, S. Nakata, and Y. Hosoda, "Regularization using QR factorization and the estimation of the optimal parameter," *BIT*, vol. 41, pp. 1049–1058, 2001.

[27] D. J. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2005.

[28] A. J. Davison, "Active search for real-time vision," in *Proc. 10th IEEE Int. Conf. Comput. Vision (ICCV)*, 2005.

[29] T. A. Davis, "Algorithm 915: SuiteSparseQR: Multifrontal multithreaded rank-revealing sparse QR factorization," *ACM Trans. Math. Softw.*, vol. 38, pp. 1–24, 2011.

[30] A. Gelman, "Objections to Bayesian statistics," *Bayesian Analysis*, vol. 3, pp. 445–450, 2008.

[31] C. H. Tong, P. Furgale, and T. D. Barfoot, "Gaussian process Gauss-Newton: Non-parametric state estimation," in *Proc. 9th Conf. Comput. Robot Vision (CRV)*, 2012.

[32] P. D. Fiore, "Efficient linear solution of exterior orientation," *IEEE Trans. Pattern Anal. Machine Intell. (PAMI)*, vol. 23, pp. 140–148, 2001.