Simultaneous Calibration, Localization, and Mapping

Rainer Kümmerle Giorgio Grisetti Wolfram Burgard

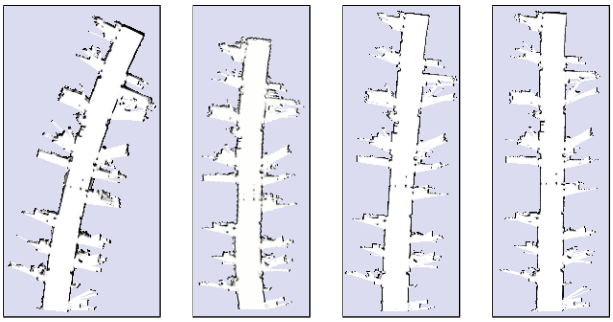
Abstract— **The calibration parameters of a mobile robot play a substantial role in navigation tasks. Often these parameters are subject to variations that depend either on environmental changes or on the wear of the devices. In this paper, we propose an approach to simultaneously estimate a map of the environment, the position of the on-board sensors of the robot, and its kinematic parameters. Our method requires no prior knowledge about the environment and relies only on a rough initial guess of the platform parameters. The proposed approach performs on-line estimation of the parameters and it is able to adapt to non-stationary changes of the configuration. We tested our approach in simulated environments and on a wide range of real world data using different types of robotic platforms.**

I. Introduction

Many approaches that address navigation tasks, including localization, path planning, motion control, and simultaneous localization and mapping (SLAM) rely on the knowledge of the specific robot parameters. These parameters typically include the position of the sensor on the platform or the parameters of the kinematic model that translates encoder ticks into a relative movement of the mobile base.

The influence of the parameters on the accuracy of state estimation processes can be substantial. For instance, an accurate calibration of the odometry can seriously improve the expected accuracy of the motion prediction by reducing the search space of the algorithms that provide the motion estimates. Figure 1 shows a motivating example. Here, we ran a scan-matching algorithm based on the odometry prediction of a robot that moves along a corridor. Since the corridor is not rich in features, the scan matcher yields solutions that are highly ambiguous along the corridor's axis. As a result, scan-matching approaches tend to make corridors "shorter". To limit this effect one can restrict the search space of the scan-matcher to a small region around the position predicted by odometry. This reduces the computational requirements but requires a highly accurate calibration of the odometry.

To obtain these parameters it is common to either rely on the specifications of the platform, to manually measure them, or to run ad-hoc calibration procedures before a mission. The latter solution is typically the most accurate, but suffers from two main drawbacks: it is not able to estimate non-stationary parameters and it needs to be repeated whenever there is a potential change in the robot configuration. Such odometry parameter changes can, for example, be the consequence of a changed load on the robot or the particular surface the robot moves upon. One solution to estimate these parameters is to treat them as hidden state variables that have to be estimated simultaneously to the map and the position of the robot. A possible method to estimate them is to use special equipment such as an external position tracking device. More promising, however are calibration procedures that only rely on the data gathered by the robot and do not require any preparation or additional information.



(a) (b) (c) (d)

Fig. 1. (a) Map obtained by the raw uncalibrated odometry of a robot with unevenly inflated tires traveling along a corridor. The result of applying a scan-matching algorithm with a large search space to account for the uncalibrated odometry leads to the shortened map shown in (b). A restriction of the search space is not able to fully correct the errors as visualized in (c). However, applying the correct calibration together with a small search space leads to an accurate estimate depicted in (d).

In this paper, we present an approach to estimate the calibration parameters of a robot equipped with a laser scanner and wheel encoders while it performs SLAM. We model the problem as a hyper-graph, where each node represents either a robot position, the laser position on the robot, or the kinematic parameters of the odometry. Our approach allows to determine these state variables on the fly (e.g., sensor positions and odometry calibration). To deal with temporal changes or more in general with interdependencies between the parameters and the other state variables we estimate the parameters on the most recent data. This approach allows a mobile robot, for example, to estimate a different set of odometry parameters for different regions of the environment and to better model the motion of the robot in these areas. Our approach might additionally be beneficial in a variety of contexts including, for instance, terrain classification. We present evaluations of our approach in simulated and a wide range of real world experiments using several robot platforms moving on different types of ground.

II. Related Work

The traditional approaches to calibrate a mobile robot and its sensors involve to accurately measure the trajectory of the robot while recording odometry and sensor measure-

This work has partly been supported by the European Commission under FP7-231888-EUROPA and FP7-248873-RADHAR.

All authors are with the University of Freiburg. G. Grisetti is also with Sapienza, University of Rome.

ments, for example, by external cameras or lasers [1]. To calibrate the sensor position, they match the measurements against a known map to recover the trajectory of the sensor and use a least squares estimator to determine the relative transformation between the robot and its sensor. In a similar way the odometry parameters can be estimated via another independent least-squares estimator given the knowledge of the reference trajectory [2].

In the context of computer vision, the idea to calibrate the intrinsic camera parameters while performing structure from motion is commonly known [3]. This problem has a structure which is very similar to the one addressed by our method. The main difference lies in the kind of parameters that are estimated. One of the first approaches to determine the stationary parameters of a mobile robot and to determine the error of the motion was proposed by Martinelli and Siegwart [4]. The idea behind this work is to extend the state of a Kalman filter used for localization with the kinematic parameters of the odometry. Whereas this approach can operate on-line, it requires an a-priori known map. Subsequently, Jones *et al.* [5] and Kelly *et al.* [6] extended the EKF and UKF algorithm to include calibration parameters. Despite their increased complexity, in these non-linear problems smoothing approaches outperform filtering methods in terms of accuracy.

Eliazar and Parr [7] proposed to use an EM approach to learn the motion model for a mobile robot. Their method is able to accurately estimate the parameters and it does not require to know the map in advance. However, their approach requires to be run off-line due to its high computational requirements. Based on a localization algorithm Roy and Thrun [8] estimate online the systematic error in the odometry. They treat the error in translation and rotation independently. Both approaches model the calibration as a linear function of the odometry measurement, whereas our approach estimates the physical parameters of the robot.

Gao and Spletzer [9] presented an approach to determine the extrinsic calibration parameters between two laser range finders. Underwood *et al.* [10] proposed a method to determine the 3D position of a laser within the body frame of the robot. Compared to our method those approaches either rely on establishing feature correspondences between the individual observations by preparing the environment with laser-reflective tape or assume a simple and partially known geometric environment for calibrating the sensors.

Censi *et al.* [11] proposed a technique similar to our method. They construct a least squares calibration problem that estimates both the kinematic parameters and the sensor position. Their approach does not need to know the map in advance, but it is restricted to the estimation of stationary parameters. Furthermore, since it relies on scan-matching to estimate the ego-motion of the sensor, this method does not provide an accurate map in large and loopy environments.

Our work can be seen as an extension of traditional graph-based SLAM algorithms. These methods model the SLAM problem as a graph, whose nodes represent robot poses and whose edges connect two nodes if there is a measurement involving both of them. Each edge is labeled with the relative transformation between the robot poses. An edge can arise either from matching a pair of observations or from an odometry measurement between consecutive poses. A solution to the problem is a configuration of the nodes that better satisfies the measurements encoded in the constraints. One seminal work in this context is the work of Lu and Milios [12] where the relative motion between two scans is measured by scan-matching and the resulting graph is optimized by iterative linearization. In the past, several methods have been proposed to either optimize the graph on-line and in a faster way [13], [14], [15], [16] or to extract the graph from the raw measurements in more robust ways [17], [18]. All these approaches assume a known calibration of the system.

Note that when we augment the problem with the calibration variables it cannot be described anymore by a graph but instead requires a hyper-graph. This is because a measurement does not only depend on a pair of variables (the connected nodes), but rather on a triplet (the nodes and the calibration parameters). In this paper, we therefore extend the standard graph optimization framework to handle this class of problems. Our approach is able to simultaneously estimate the map of the environment and calibrate the parameters of a robot in a continuous manner. We do not require any special preparation for the environment, such as an external tracking system or landmarks to be placed in the designated area.

III. Simultaneous Calibration, Localization, and Mapping

Our system relies on the graph-based formulation of the SLAM problem to estimate the maximum-likelihood configuration. In contrast to the traditional SLAM methods we explicitly model that the measurements obtained by the robot are given in different coordinate frames. For example, the odometry of the robot is given by the velocity measurements of its wheels. Applying the forward kinematics of the platform allows to transform the velocities measured during a time interval into a relative displacement of the platform expressed in the odometry frame. Additionally, the robot is usually equipped with a sensor that is able to observe the environment, e.g., a laser range finder. This sensor is mounted on the robot and obtains measurements in its own coordinate frame. Thus, a scan-matching algorithm which aligns two range scans in a common coordinate frame has to project the computed motion through the kinematic chain of the robot to estimate the motion of the robot's base. As it is not always easy to measure the offset transformation between the base of the robot and the sensor or to determine the parameters for the forward kinematics, we suggest to integrate those into the maximum likelihood estimation process.

A. Description of the Hyper-Graph

Whenever the robot obtains a measurement we add a node to the graph. This node represents the position of the robot at which the measurement was obtained. Let $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^\top$ be a vector of parameters, where $\mathbf{x}_i =$

$(x_i, y_i, \theta_i)^\top$ describes the position of node i. Furthermore, let $\mathbf{l}$ be the 2D pose of the sensor relative to the coordinate frame of the robot and let $\mathbf{z}_{ij}$ and $\mathbf{\Omega}^{\mathbf{z}}_{ij}$ be respectively the mean and information matrix of an observation of node j seen from node i. Finally, let $\mathbf{k}$ be the parameters of the forward kinematics function and $\mathbf{u}_i$ and $\mathbf{\Omega}^{\mathbf{u}}_i$ be respectively the motion command and the information matrix which translates the robot from node i to $i+1$.

The error function $\mathbf{e}^{\mathbf{l}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{l}, \mathbf{z}_{ij})$ measures how well the parameter blocks $\mathbf{x}_i$, $\mathbf{x}_j$, and $\mathbf{l}$ satisfy the constraint $\mathbf{z}_{ij}$. If the three parameters perfectly satisfy the error function, then its value is $\mathbf{0}$. Here, we assume that the laser is mounted without inclination which is the ideal condition. For simplicity of notation, we will encode the involved quantities in the indices of the error function:

$$\mathbf{e}^{\mathbf{l}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{l}, \mathbf{z}_{ij}) \stackrel{\text{def.}}{=} \mathbf{e}^{\mathbf{l}}(\mathbf{x}_i, \mathbf{x}_j) \stackrel{\text{def.}}{=} \mathbf{e}^{\mathbf{l}}_{ij}(\mathbf{x}). \tag{1}$$

The error function $\mathbf{e}^{\mathbf{l}}_{ij}(\mathbf{x})$ has the following form:

$$\mathbf{e}^{\mathbf{l}}_{ij}(\mathbf{x}) = ((\mathbf{x}_j \oplus \mathbf{l}) \ominus (\mathbf{x}_i \oplus \mathbf{l})) \ominus \mathbf{z}_{ij}, \tag{2}$$

where $\oplus$ is the usual motion composition operator [19] and $\ominus$ its inverse.

Additionally, the error function $\mathbf{e}^{\mathbf{u}}_i(\mathbf{x}_i, \mathbf{x}_{i+1}, \mathbf{k}, \mathbf{u}_i)$ measures how well the parameter blocks $\mathbf{x}_i$, $\mathbf{x}_j$, and $\mathbf{k}$ satisfy the constraint $\mathbf{u}_i$. Again, a value of $\mathbf{0}$ means that the constraint is perfectly satisfied by the parameters. The error function $\mathbf{e}^{\mathbf{u}}_i(\mathbf{x})$ is defined as

$$\mathbf{e}^{\mathbf{u}}_i(\mathbf{x}) = (\mathbf{x}_{i+1} \ominus \mathbf{x}_i) \ominus K(\mathbf{u}_i, \mathbf{k}), \tag{3}$$

where $K(\cdot)$ is the forward kinematics function converting from wheel velocities to a relative displacement of the vehicle. In Eq. (3) we applied the same simplifying notation as in Eq. (1).

For a robot with a differential drive, which is one of the most common types of robots, the odometry $\mathbf{u} = (v_l, v_r)^\top$ consists of the velocities of the left and the right wheel. The wheel velocities are computed by counting the encoder ticks of the motors during the time step which are multiplied by the respective radii r_l and r_r of the wheels. Furthermore, the distance b between the two wheels has to be known to compute the circular arc on which the robot moves. The relative motion during the time interval Δt is given by

$$K(\mathbf{u}, \mathbf{k}) = \begin{pmatrix} R(\Delta t\omega) & \mathbf{0} \\ \mathbf{0} & 1 \end{pmatrix} \begin{pmatrix} -ICC \\ 0 \end{pmatrix} + \begin{pmatrix} ICC \\ \Delta t\omega \end{pmatrix}, \tag{4}$$

where $R(\cdot)$ is the 2D rotation matrix of its argument, $ICC = (0, \frac{b}{2}\frac{r_l v_l + r_r v_r}{r_l v_l - r_r v_r})^\top$, and $\omega = \frac{r_l v_l - r_r v_r}{b}$. Thus, the calibration parameter $\mathbf{k} = (r_r, r_l, b)^\top$ for the odometry is a three-dimensional vector.

The goal of our maximum likelihood approach is to find the configuration of $[\mathbf{x}^*, \mathbf{l}^*, \mathbf{k}^*]$ which minimizes the negative log-likelihood $\mathbf{F}(\mathbf{x}, \mathbf{l}, \mathbf{k})$ given all the observations

$$\begin{aligned} &\mathbf{F}(\mathbf{x}, \mathbf{l}, \mathbf{k}) \\ &= \sum_{\langle i,j \rangle} \mathbf{e}^{\mathbf{l}}_{ij}(\mathbf{x})^\top \mathbf{\Omega}^{\mathbf{z}}_{ij} \mathbf{e}^{\mathbf{l}}_{ij}(\mathbf{x}) + \sum_i \mathbf{e}^{\mathbf{u}}_i(\mathbf{x})^\top \tilde{\mathbf{\Omega}}^{\mathbf{u}}_i \mathbf{e}^{\mathbf{u}}_i(\mathbf{x}), \end{aligned} \tag{5}$$

where $\tilde{\mathbf{\Omega}}^{\mathbf{u}}_i$ is the projection of $\mathbf{\Omega}^{\mathbf{u}}_i$ through the forward kinematics function $K(\cdot)$ via the unscented transformation [20]. Since the projection depends on the estimate of $\mathbf{k}$, we update the projection if $\mathbf{k}$ changes substantially.

Given this formulation we may easily integrate prior knowledge, for example, the manually — thus non-precisely — measured transformation of the laser. This is possible as long as the prior information can be represented by a Gaussian distribution. Furthermore, state changes observed by measurements, e.g., the robot actively rotates the laser scanner, can be incorporated.

To estimate the calibration parameters, the trajectory of the robot should introduce measurements that constrain all possible dimensions of $\mathbf{k}$ and $\mathbf{l}$. Clearly, a trajectory only consisting of straight line motions does not allow to observe the position of the laser. The same holds for a circular trajectory, since the laser could be anywhere on the circle. Both cases are pathologic and can easily be avoided by varying the wheel velocities of the robot.

B. Estimation via Least Squares on a Hyper Graph

Without loss of generality we will refer to the whole state vector $[\mathbf{x}\ \mathbf{l}\ \mathbf{k}]$ as $\mathbf{y}$, without distinguishing the parameter blocks. Additionally, we identify each hyper-edge by a unique index k instead of the pair of indices i, j and the superscript letter as in Eq. (5).

If a good initial guess $\breve{\mathbf{y}}$ of the parameters is known, a numerical solution of Eq. (5) can be obtained by using the popular Gauss-Newton or Levenberg-Marquardt (LM) algorithms [21, §15.5]. The idea is to approximate the error function by its first order Taylor expansion around the current initial guess $\breve{\mathbf{y}}$

$$\begin{aligned} \mathbf{e}_k(\breve{\mathbf{y}}_k \boxplus \mathbf{\Delta y}_k) &= \mathbf{e}_k(\breve{\mathbf{y}} \boxplus \mathbf{\Delta y}) && (6) \\ &\simeq \mathbf{e}_k + \mathbf{J}_k \mathbf{\Delta y}. && (7) \end{aligned}$$

Here, $\mathbf{J}_k$ is the Jacobian of $\mathbf{e}_k(\mathbf{y} \boxplus \mathbf{\Delta y})$ with respect to $\mathbf{\Delta y}$ computed in $\mathbf{\Delta y} = 0$ and $\mathbf{e}_k \stackrel{\text{def.}}{=} \mathbf{e}_k(\breve{\mathbf{y}})$. Furthermore, $\boxplus$ is an operator that applies the increments $\mathbf{\Delta y}$ to the current state $\breve{\mathbf{y}}$. This accounts for over-parameterized states, and can better deal with non Euclidean state spaces. Clearly, if both $\breve{\mathbf{y}}$ and $\mathbf{\Delta y}$ are Euclidean the $\boxplus$ degenerates to a regular $+$. Substituting Eq. (7) in the error terms $\mathbf{F}_k$ of Eq. (5), we obtain the following quadratic form

$$\mathbf{F}(\mathbf{y}) = \mathrm{c} + 2\mathbf{b}^T \mathbf{\Delta y} + \mathbf{\Delta y}^T \mathbf{H} \mathbf{\Delta y} \tag{8}$$

that can be minimized in $\mathbf{\Delta y}$ by solving the system

$$(\mathbf{H} + \lambda \mathbf{I})\, \mathbf{\Delta y}^* = -\mathbf{b}. \tag{9}$$

Here, $\mathbf{H}$ and $\mathbf{b}$ are respectively the approximated Hessian of the error function and the error gradient:

$$\begin{aligned} \mathbf{H} &= \sum_k \mathbf{J}_k^T \mathbf{\Omega}_k \mathbf{J}_k && (10) \\ \mathbf{b} &= \sum_k \mathbf{J}_k^T \mathbf{\Omega}_k \mathbf{e}_k. && (11) \end{aligned}$$

λ is a damping factor: the larger λ is the smaller are the $\mathbf{\Delta y}$. This is useful to control the step size in case of non-linear

surfaces. The idea behind the LM algorithm is to dynamically control the damping factor. At each iteration the error of the new configuration is monitored. If the new error is lower than the previous one, λ is decreased for the next iteration. Otherwise, the solution is reverted and λ is increased. For $\lambda = 0$ this corresponds to the Gauss-Newton method.

Whenever the increments $\mathbf{\Delta y}$ are computed by solving Eq. (9), they can be applied to the previous solutions by means of the $\boxplus$ operator

$$\check{\mathbf{y}} \quad \leftarrow \quad \check{\mathbf{y}} \boxplus \mathbf{\Delta y}^{*}. \tag{12}$$

From Eq. (5) we notice that each of the terms in the sum depends on at most three parameter blocks. More precisely, if the constraint k arises from an odometry measurement, it will depend on the connected robot poses $\mathbf{x}_i$ and $\mathbf{x}_j$ and by the odometry parameters $\mathbf{k}$. Alternatively, if a constraint arises from a laser measurement, it will depend on the connected robot poses $\mathbf{x}_i$ and $\mathbf{x}_j$ and on the laser position $\mathbf{l}$. Accordingly, each of the Jacobians in Eq. (10) and in Eq. (11) will have only three non-zero blocks, in correspondence of the variables involved by the constraint. Thus, each of the terms in the sum of Eq. (10) will be a matrix with at most nine non-zero components. Furthermore, $\mathbf{H}$ will have a number of non-zero entries proportional to the number of constraints. This results in a sparse system that can be efficiently solved. To solve the optimization problem, we employ the g^{2}o toolkit [22] which allows us to solve one iteration of a calibration problem having 3,000 nodes in less than 0.01 s using one core of an Intel i7@2.8 GHz.

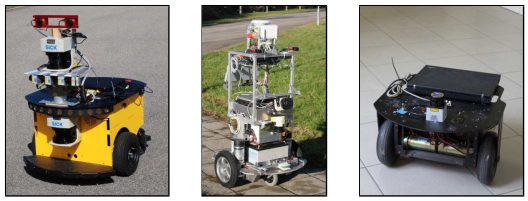
C. Monitoring the Convergence

Some calibration parameters may be constant while others change. For example, the laser position $\mathbf{l}$ is constant if the robot has no actuator to move this sensor. Therefore, it is of interest to decide whether enough data is collected to stop calibrating to reduce the computational demands. To this end, we can consider the approximated Hessian $\mathbf{H}$ and compute the marginal covariance of the calibration parameters. The marginal covariance $\mathbf{\Sigma}_{\mathbf{l}}$ of the calibration parameter is given by extracting the corresponding block of $\mathbf{H}^{-1}$. The matrix $\mathbf{H}$ is sparse, symmetric, and positive definite, thus Cholesky decomposition can be applied to factorize $\mathbf{H}$. By applying an algorithm based on dynamic programming (see [23]) we can efficiently compute the desired elements of $\mathbf{H}^{-1}$ given the Cholesky factor. As we will show in the experiments this information allows us to access the quality of the estimated parameters.

IV. Experiments

The approach described above has been implemented and evaluated on both simulated and real-world data acquired with a heterogeneous set of robots equipped with laser range finders. Figure 2 visualizes the robots we used to collect the real-world data used in this paper.

The SLAM front-end for processing the data is an own implementation of the framework described by Olson [24] which employs a correlative scan-matcher to estimate the transformation of the laser along with the 3×3 covariance matrix representing the uncertainty of the estimated transformation. The correlative scan-matcher performs an exhaustive search to determine the best fitting alignment for two laser scans within a given search radius. We add a new node to the graph whenever the robot moved 0.1 m or rotated 10° whichever occurs first.



(a) (b) (c)

Fig. 2. The robots used to acquire the real-world data sets: (a) MobileRobots PowerBot (b) a custom made platform (c) Pioneer.

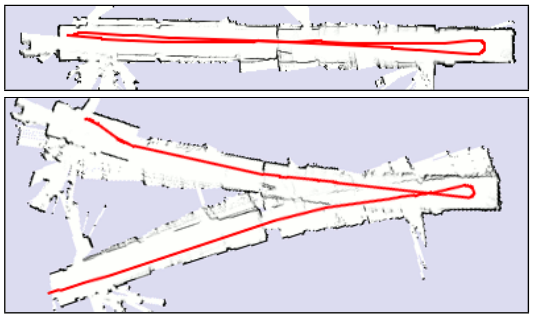


Fig. 3. Robot driving up and down a corridor. Top: Applying the calibration corresponding to the current configuration of the robot leads to a good odometry estimate. Bottom: If the robot is carrying a load, the same calibration parameters results in a severe drift in the odometry.

A. Online odometry calibration

In real world scenarios the odometry is affected by different factors. For example, if the robot is carrying a load, the additional weight compresses inflated tires and results in reduced wheel radii. To this end, we used the PowerBot platform (see Figure 2a) which has a maximum payload of 100 kg to carry a load of approximately 40 kg. The wheels of the PowerBot are inflated tires whose radii are affected by both the air-pressure of the tires and the total weight of the platform. The load in this set of experiments was intentionally placed on the left hand side of the robot. In a first experiment we recorded datasets in which the robot was either carrying the load or it was operating in its normal configuration. We used one data set for estimating the parameters and a different one for evaluating the odometry calibration parameters. Our approach estimated wheel radii of $r_r = 0.1251$ m, $r_l = 0.1226$ m for the normal configuration of the robot and $r_r = 0.1231$ m, $r_l = 0.1223$ m while carrying the load. The difference seems to be small, however it has a substantial effect. Figure 3 shows the outcome of applying the estimate of the normal configuration to the robot carrying the load. Applying the wrong calibration parameter has a crucial effect on the trajectory as it is estimated by the odometry. Since the weight of the load is mutable and can be placed in an arbitrary position on the robot, the best

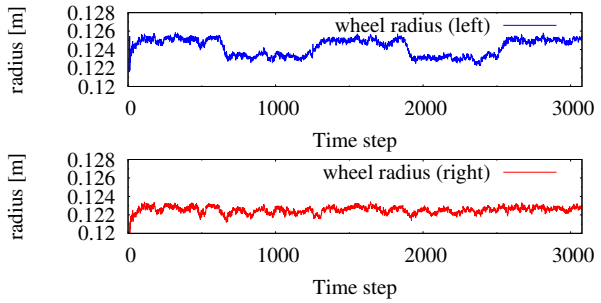


Fig. 4. Results of the online estimation of the wheel radii. The robot had to carry a load twice which was placed on the left hand side of the platform leading to a compression of the left wheel.

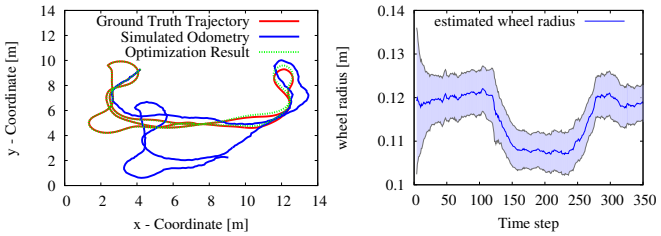


Fig. 5. Left: The simulated robot trajectory. Right: Estimating the wheel radii online based on the most recent observations. Here, the robot was carrying a load during the time interval $[120, 240]$ which leads to compressed wheels having a smaller radii.

performance can be obtained by calibrating the odometry parameters while the robot is operating.

By considering the 50 most recent measurements within a sliding window around the current node we are able to estimate the wheel radii online also when they are subject to change due to external factors. For older odometry measurements outside the sliding window we change the error function $\mathbf{e}_i^{\mathbf{u}}(\mathbf{x})$ to employ a fixed value for $\mathbf{k}$, i.e., we only estimate the physical parameters on the recent data and use the previously estimated parameters to model the odometry error term for older measurements. Figure 4 visualizes the estimated wheel radii during an experiment in which the robot had to carry a load placed on the left hand side of the platform. The robot was carrying the load during the intervals $[600, 1250]$ and $[1865, 2530]$. Using our approach we are able to correctly estimate the wheel radii independent of the load carried by the robot along with the maximum likelihood map of the environment.

As it is hard to obtain ground truth data for real-world data-sets, we simulated a robot traveling on the trajectory depicted in the left part of Figure 5. The simulator allows us to directly judge the quality of the calibration results. The odometry measurement and the range measurements obtained by the robot are perturbed by Gaussian noise. Within a simulation experiment we modeled a robot carrying a weight which we simulated having the effect of a reduction of the wheel radius from 0.12 m to 0.11 m. The robot carries the load during the time interval $[120, 240]$. Figure 5 depicts the results of the online calibration based on the most recent measurements. As we can see, the estimate is able to represent the compressed wheels and corresponds well to the ground truth given by the simulator. Furthermore, the trajectory as it is estimated by our approach also matches well to the ground truth as shown in Figure 5.

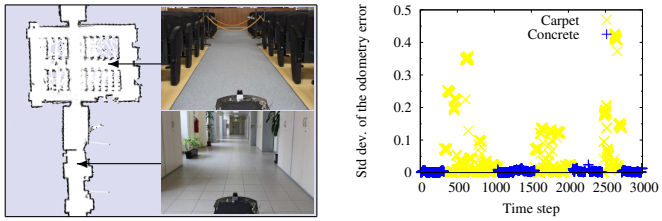


Fig. 6. Left: In indoor environments a robot may encounter different floor types. Right: The standard deviation of the error of the odometry edges for the sliding window at each time step.

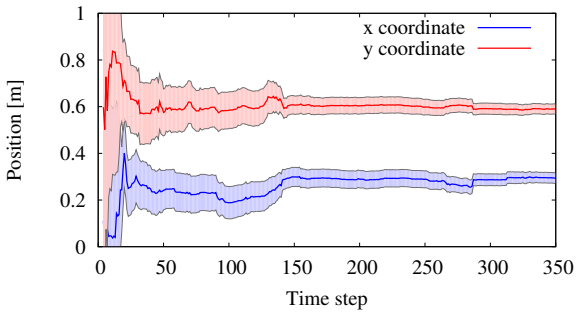


Fig. 7. The evolution of the x and y coordinate of the laser transformation as it is estimated by our approach. The true value of the x and y coordinate is 0.3 m and 0.6 m respectively.

B. Influence of the ground surface

Within real world indoor environments a robot may encounter different floor types, e.g., tiling, PVC flooring, wooden floor, or different types of carpets. To test the influence of the floor type, we recorded data sets in which the robot drives on a soft carpet and on concrete tiling floor, see left image in Figure 6. In this experiment we estimated the odometry parameters online. On both floors the estimated wheel radii were the same. However, by analyzing the standard deviation (see right part of Figure 6) in the error of the odometry edges $\mathbf{e}_i^{\mathbf{u}}$ for the sliding window around the current node, we observe a higher noise in the odometry due to slippage on the carpet. This information can be stored in the map so that the robot can use it to adjust the motion model noise in a localization task.

C. Simulation Experiments

In a first experiment we simulated the laser having a relative transformation of $(0.3, 0.6, 30°)^\top$ with respect to the odometry frame of the robot. Here, we optimized after inserting every node and monitored the evolution of the laser transformation as it is estimated by our approach at each time step. Figure 7 visualizes how the estimate for the x and y coordinate of the relative laser transformation along with their estimated uncertainty evolves. As we can see the estimate converges quickly to the correct transformation. By monitoring the marginal covariance of the estimated laser transformation we are able to judge the quality of the estimate.

Furthermore, we estimated the odometry parameters of the simulated robot whose left wheel has a radius of $r_l = 0.12\,\text{m}$ whereas the right wheel has a radius of $r_r = 0.125\,\text{m}$. The distance between the wheels is $b = 0.6\,\text{m}$. The output of

TABLE I

THE PARAMETERS OF THE ROBOTS USED FOR OUR EXPERIMENTS.

	PowerBot	Custom	Pioneer
wheel radius [m]	0.125	0.16	0.065
wheel distance [m]	0.56	0.7	0.35
ticks per revolution	22835	20000	1970
laser offset [m, m, °]	(0.22, 0, 0)	(0.3, 0, 0)	(0.1, 0, 0)
laser scanner model	Sick LMS291	Sick LMS151	Hokuyo URG

TABLE II

CALIBRATION RESULTS FOR DIFFERENT ROBOT DATA SETS.

	laser offset (m, m, °)	wheel radii (m, m)	distance m
PowerBot - 1	(0.2258, 0.0026, 0.099)	(0.1263, 0.1275)	0.5825
PowerBot - 2	(0.2231, -0.0031, 0.077)	(0.1243, 0.1248)	0.6091
Custom - 1	(0.3067, -0.0051, -0.357)	(0.1603, 0.1605)	0.6969
Custom - 2	(0.3023, -0.0087, -0.013)	(0.1584, 0.1575)	0.7109
Pioneer - 1	(0.1045, 0.009, -0.178)	(0.0656, 0.065)	0.3519
Pioneer - 2	(0.1066, -0.0031, -0.28)	(0.0658, 0.0655)	0.3461

the calibration is $\hat{r}_l = 0.1207\,\text{m}$, $\hat{r}_r = 0.1264\,\text{m}$, and $\hat{b} = 0.607\,\text{m}$.

We carried out further simulation experiments in which we randomly sampled the transformation of the laser with respect to the odometry frame and also modified the true wheel radii and the distance between the wheels. The calibration parameters as they are estimated by our approach did in all cases correspond well to the true values and the error was in the same range like in the particular example reported above.

D. Real-World Experiments

To evaluate our approach on real-world data we processed data of a heterogeneous set of robots depicted in Figure 2. Table I summarizes the parameters of the platforms. To collect the data, we steered each robot twice through the environment. The front-end again processed the data to estimate the motion of the laser for each time step. In case the robot re-visits an already known region, the loop closure constraints are added to the graph. Note that the estimation of the calibration parameters does not require to detect loop closures. However, such a constraint allows to reduce the residual error in the trajectory as it is estimated by our approach. Table II summarizes the calibration results. As we can see, the result for the laser transformation are within a few millimeters of the manually measured position. The same holds for the radii of the wheels and their distance to each other.

V. CONCLUSIONS

In this paper, we presented an approach to estimate the calibration parameters while performing SLAM. Our approach extends the graph-based formulation of the SLAM problem to handle the calibration parameters. The overall approach is accurate and designed for online operation, which allows us to handle changes in the parameters, for example, induced by placing a load on the robot or by wear of the robot. Furthermore, compared to ad-hoc calibration methods our approach solely relies on the on-board sensors of the robot and does not require external information.

Additionally, our approach has the potential to provide useful information about the ground surface which affects the uncertainty of the odometry measurements. This information may in the future be exploited for terrain classification and might also be considered by localization algorithms.

REFERENCES

[1] S. Ceriani *et al.*, "RAWSEEDS ground truth collection systems for indoor self-localization and mapping," *Journal of Autonomous Robots*, vol. 27, no. 4, 2009.

[2] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*. Springer, 2008.

[3] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, 2004.

[4] A. Martinelli and R. Siegwart, "Estimating the odometry error of a mobile robot during navigation," in *Proc. of the European Conference on Mobile Robots (ECMR)*, 2003.

[5] E. Jones, A. Vedaldi, and S. Soatto, "Inertial structure from motion with autocalibration," in *Proceedings of the Internetional Conference on Computer Vision - Workshop on Dynamical Vision*, 2007.

[6] J. Kelly and G. S. Sukhatme, "Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration," *Int. Journal of Robotics Research*, vol. 30, no. 1, pp. 56–79, 2011.

[7] A. I. Eliazar and R. Parr, "Learning probabilistic motion models for mobile robots," in *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2004.

[8] N. Roy and S. Thrun, "Online self-calibration for mobile robots," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 1999.

[9] C. Gao and J. R. Spletzer, "On-line calibration of multiple lidars on a mobile vehicle platform," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2010.

[10] J. Underwood, A. Hill, and S. Scheding, "Calibration of range sensor pose on mobile platforms," in *Proc. of the Int. Conf. on Intelligent Robots and Systems (IROS)*, 2007.

[11] A. Censi, L. Marchionni, and G. Oriolo, "Simultaneous maximum-likelihood calibration of robot and sensor parameters," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2008.

[12] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *Autonomous Robots*, vol. 4, no. 4, pp. 333–349, 1997.

[13] U. Frese, P. Larsson, and T. Duckett, "A multilevel relaxation algorithm for simultaneous localisation and mapping," *IEEE Transactions on Robotics*, vol. 21, no. 2, pp. 1–12, 2005.

[14] E. Olson, J. Leonard, and S. Teller, "Fast iterative optimization of pose graphs with poor initial estimates," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2006, pp. 2262–2269.

[15] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2011.

[16] G. Grisetti, C. Stachniss, and W. Burgard, "Non-linear constraint network optimization for efficient map learning," *IEEE Transactions on Intelligent Transportation Systems*, 2009.

[17] K. Konolige, "A gradient method for realtime robot control," in *Proc. of the Int. Conf. on Intelligent Robots and Systems (IROS)*, 2000.

[18] E. Olson, M. Walter, J. Leonard, and S. Teller, "Single cluster graph partitioning for robotics applications," in *Proceedings of Robotics Science and Systems*, 2005, pp. 265–272.

[19] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial realtionships in robotics," in *Autonomous Robot Vehicles*, I. Cox and G. Wilfong, Eds. Springer Verlag, 1990, pp. 167–193.

[20] S. Julier, "The scaled unscented transformation," in *Proc. of the IEEE Amer. Control Conf*, 2002.

[21] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes, 2nd Edition*. Cambridge Univ. Press, 1992.

[22] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g^{2}o: A general framework for graph optimization," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2011.

[23] M. Kaess and F. Dellaert, "Covariance recovery from a square root information matrix for data association," *Journal of Robotics and Autonomous Systems, RAS*, vol. 57, pp. 1198–1210, Dec 2009.

[24] E. Olson, "Robust and efficient robotic mapping," Ph.D. dissertation, MIT, Cambridge, MA, USA, June 2008.