Unsupervised Calibration of Wheeled Mobile Platforms

Maurilio Di Cicco, Bartolomeo Della Corte, Giorgio Grisetti

***Abstract*— This paper describes an unsupervised approach to retrieve the kinematic parameters of a wheeled mobile robot. The robot chooses which action to take in order to minimize the uncertainty in the parameter estimate *and* to fully explore the parameter space.**

Our method explores the effects of a set of elementary motion on the platform to dynamically select the best action and to stop the process when the estimate can be no further improved.

We tested our approach both in simulation and with real robots. Our method is reported to obtain in shorter time parameter estimates that are statistically more accurate than the ones obtained by steering the robot on predefined patterns.

I. Introduction

Calibrating a robot is a tedious and time consuming mandatory procedure, whose outcome influences the overall system performances. A small error in calibration might lead to severe inconsistencies in tasks that rely on sensor information such as localization, mapping and navigation in general.

The task of calibrating a mobile robot has been addressed since many years. In this paper we focus on the so called kinematic calibration. In a wheeled mobile robot this consists of estimating the odometry parameters, that are required to convert wheel encoder ticks in a relative motion of the mobile base on a local plane, and of estimating the position of one or more sensors on the mobile base.

Several approaches to address this task have been proposed in the literature. All of them are regarded as passive processes, where the user moves the platform in an environment, recording the data from the encoder and from the exteroceptive sensors (like laser scanners or cameras) mounted on the robot. The calibration procedure consumes these data and seeks for the parameters that better explain the sensor perceptions, given a calibration pattern.

The quality of the calibration is greatly influenced by the trajectory taken by the robot, and for specific combination of platforms/sensors there are common motion patterns that are likely to produce good results. As an instance, calibrating a wheeled platform moving on a plane and equipped with a laser scanner can be done by moving the robot on a square or along an eight-shaped path in a known room so to explore the entire parameter space.

All authors are with Department of Computer, Control, and Management Engineering Antonio Ruberti of Sapienza University of Rome, Via Ariosto 25, I00185 Rome, Italy.

Email: {lastname}@dis.uniroma1.it.

This work has partly been supported by the European Commission under FP7-600890-ROVINA.

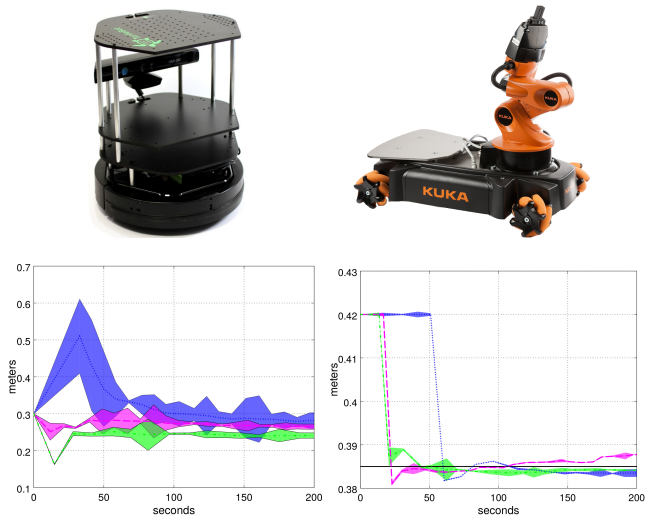


Fig. 1: On the top of the image two different kind of mobile robots are shown, a differential drive and an omnidirectional base. In the bottom the respective baseline estimates are shown during the execution of different calibration strategies along with their uncertainties. The *green* dashed line represents the evolution observed during the unsupervised procedure. The *blue* dotted line shows the evolution during the calibration following the square-path. Lastly, the *magenta* dash-dotted line shows the evolution of the observed parameter following the eight-shaped path. In the bottom right image, even the ground truth value (*black* solid line) is shown since the experiment was conducted in simulation.

Yet, this procedure needs to be typically carried on by an expert. To produce sufficient data to guarantee a reasonable calibration, the data acquisition usually lasts some minutes.

In this paper we present a fully automatic approach for calibrating the kinematic parameters of a mobile robot equipped with a sensor suitable for motion tracking, such as a depth camera or a laser scanner. The system relies only on the knowledge of the kinematic model of the robot, and chooses how to move in the environment to minimize the uncertainty of the estimate and to fully explore the parameter space.

We performed extensive tests on robots having different kinematic models both in real world scenarios and in simulation. We compared our results with the ones obtained by manually executing fixed calibration patterns. In our tests the proposed approach provided statistically more accurate parameter estimates and required less time than the manual procedures.

Figure 1 shows a motivating example of our procedure, used to calibrate two types of mobile base. The plots in the second line show the evolution of the estimate and its

uncertainty as more training data are gathered.

II. Related work

Calibrating a mobile robot has been addressed since more than two decades. In this section we provide an overview of approaches addressing this topic and we relate them with our contribution.

Borenstein and Feng [4] developed a method named UMBmark. It requires to drive a differential drive robot along a square path both clockwise and counterclockwise. Following this cyclic path, the robot has to come back to the starting position. The distance between the initial pose and the final pose measured by the odometry represents the error. Minimizing such error leads to the calibration of the direct kinematics of the wheeled base. This method relies on a pre-constructed trajectory and it neglets the possibility to calibrate nothing but the odometry parameters.

Subsequently Kelly [7] proposed a generic odometry calibration approach that does not require the robot to travel on a squared path. Such method practically requires to drive the robot along any trajectories for a couple of dozens of times while predefined waypoints along the paths are used as ground truth. The quality of results is highly dependant on the trajectories involved in the calibration as well as the number of ground truth points.

Chenavier and Crowley [6] worked on a Kalman filter to calibrate both the odometry parameters and the camera pose on the mobile robot. A known map of fixed objects, deployed into the environment, is exploited by the camera, which is treated as bearing sensor.

The approach of Martinelli et al. [2] simultaneusly estimates the systematic and non systematic odometry error of a mobile robot. By using this statistics they enhance the effectiveness of a gaussian filter for SLAM [8].

Censi et al. [1] proposed a max likelihood approach which does not rely on specific trajectories. The method provides a simultaneous calibration of both the intrinsics odometry parameters as well as the pose of a 2D laser scanner used egomotion estimator.

Kümmerle et al. [9] augment the state of a graph based SLAM method with unknown calibration parameters. This approach allows to simultaneously calibrate both odometry and position of sensors while the robot operates. It is particularly effective in case of non-stationary situations that might occur during a mission (e.g. changes in the kinematics due do varying load distributions).

The last two approaches are among the few that address the issue of calibrating the odometry and the sensor positions. Yet, they operate as passive processes consuming the data as they are gathered by the moving robot.

The path taken by the robot, however has a great influence on the quality of the calibration. Each particular motion along the trajectory might depend only on a subset of parameters.

III. Least Squares Calibration of a Mobile Robot

Calibrating the kinematic parameters $\mathbf{k}$ of a wheeled mobile platform means estimating the coefficients $\mathbf{k}^{\mathrm{o}}$ that allow us to compute the relative motion of the base from the ticks measured by the encoders. If the robot is equipped with one or more exteroceptive sensor, these parameters also include the relative position of the sensor $\mathbf{k}^{\mathrm{s}}$ w.r.t. the mobile base, as shown in Figure 2.

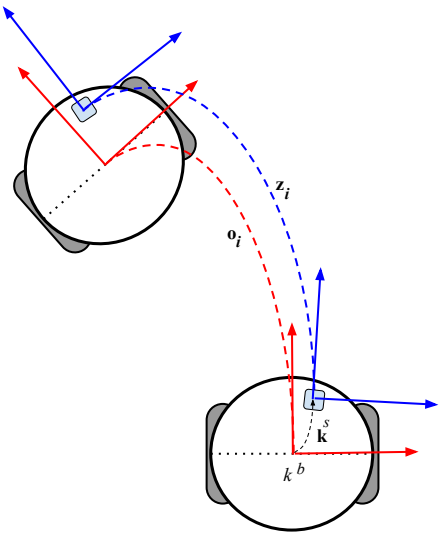


Fig. 2: This figure illustrates the relation between the mobile base and the sensor reference frame. The relative pose of the sensor, with respect to the robot frame, is $\mathbf{k}^{\mathrm{s}}$. While the mobile base moves along the trajectory $\mathbf{o}_i$, the sensor will move along the path $\mathrm{z_i}$ in its own reference frame. Lastly, k^{b} represents the robot baseline.

Knowing the mobile base parameters $\mathbf{k}^{\mathrm{o}}$ and the ticks of the wheel encoders $\mathbf{u}_i$ we can compute the motion of the mobile base during the i^{th} interval, through a direct kinematics function $\mathbf{f}(\mathbf{k}^{\mathrm{o}}, \mathbf{u}_i)$. Let $\mathbf{o}_i^T = (x_i^{\mathrm{o}}, y_i^{\mathrm{o}}, \theta_i^{\mathrm{o}})^T$ be this motion. If the robot is equipped with an exteroceptive sensor such as a laser scanner, and it operates on a plane, the sensor parameters are augmented with the sensor position $\mathbf{k}^{\mathrm{s}} = (x^{\mathrm{s}}\ y^{\mathrm{s}}\ \theta^{\mathrm{s}})^T$. Estimating the full kinematics of the robot thus means estimating the vector $\mathbf{k}^T = (\mathbf{k}^{\mathrm{o}T}\ \mathbf{k}^{\mathrm{s}T})$. If there is an algorithm that allows to estimate the ego-motion of an exteroceptive sensor uniquely from the sensor's observations, we can use the output of this algorithm as a reference for calibration. In particular, if the robot is equipped with a laser scanner such an estimate can be obtained by a scan-matcher. Also RGBD cameras can be used for this purpose, as their data can be fed to tracking algorithm such as KINFU[3] or NICP [11] that provides the trajectory of the sensor. In the remainder of this section we focus on laser scanners and 2D motion, however the results of this paper can be straightforwardly generalized to the 3D case. Let $\mathbf{z}_i = (x_i\ y_i\ \theta_i)^T$ be the estimated motion of a sensor during the i^{th} interval.

Knowing the position of the sensor on the robot $\mathbf{k}^{\mathrm{s}}$ and the motion of the base we can estimate the motion of the sensor $\hat{\mathbf{z}}_i$ from the encoder measurements $\mathbf{u}_i$ as

$$\hat{\mathbf{z}}_i = \mathbf{h}_i(\mathbf{k}) \quad (1)$$

$$= \mathbf{o}_i \oplus \mathbf{k}^{\mathrm{s}} \quad (2)$$

$$= \mathbf{f}(\mathbf{k}^{\mathrm{o}}, \mathbf{u}_i) \oplus \mathbf{k}^{\mathrm{s}}. \quad (3)$$

Here $\oplus$ denotes the motion composition operator as described by Smith and Cheesman [12], and $\mathbf{h}(\cdot)$ is the prediction function depending on the parameters and the encoder measurements.

The goal of our least squares calibration procedure is thus to find the parameters $\mathbf{k}^*$, that minimize the difference between the predicted and measured displacements of the sensor:

$$\mathbf{k}^* = \operatorname{argmin} \sum_i \|\mathbf{h}_i(\mathbf{k}) - \mathbf{z}_i\|_{\mathbf{\Omega}_i}. \tag{4}$$

Here $\|\mathbf{v}\|$ denotes the $\mathbf{\Omega}$ norm of a vector $\mathbf{v}$, as

$$\|\mathbf{v}\|_{\mathbf{\Omega}} = \mathbf{v}^T \mathbf{\Omega} \mathbf{v}. \tag{5}$$

The matrix $\mathbf{\Omega}_i$ denotes the information matrix of the solution reported by the scan matcher and it can be obtained by using the method of Censi [5]. Considering the information matrices it allows to adjust the contribution of the different error terms based on the quality of the scan matcher solution. Since our measurement $\mathbf{z}_i \in SE2$, performing the vector difference in Eq. (4) might result in non-valid configurations due to angular wraparounds. This effect can be lessened by substituting the $-$ with a more appropriate motion decomposition, denoted by the symbol $\ominus$. Thus, our function to minimize becomes:

$$\mathbf{k}^* = \operatorname{argmin} \sum_i \| \underbrace{\mathbf{h}_i(\mathbf{k}) \ominus \mathbf{z}_i}_{\mathbf{e}_i(\mathbf{k})} \|_{\mathbf{\Omega}_i}. \tag{6}$$

Here $\mathbf{e}_i(\mathbf{k})$ is the error of the i^{th} measurement and denotes the relative transform that would bring the estimate of the sensor motion obtained from the odometry to match the motion estimated by the scan matcher.

Using least squares the problem in Eq. (6) can be solved iteratively, assuming a reasonable estimate of the nominal parameters is available. At each iteration the algorithm refines the current estimate $\mathbf{k}$ by computing a perturbation $\mathbf{\Delta}_\mathbf{k}$ that minimizes a quadratic approximation of Eq. (6) in the form:

$$\mathbf{\Delta}_\mathbf{k}^T \mathbf{H} \mathbf{\Delta}_\mathbf{k} + \mathbf{b}^T \mathbf{\Delta}_\mathbf{k} + c. \tag{7}$$

The minimum of Eq. (7) can be found by solving the following linear system:

$$\mathbf{H} \mathbf{\Delta}_\mathbf{k} = -\mathbf{b}. \tag{8}$$

The matrix $\mathbf{H}$ and the vector $\mathbf{b}$ are obtained from Eq. (6) by linearizing each error $\mathbf{e}_i(\mathbf{k})$ around the current guess as:

$$\mathbf{e}_i(\mathbf{k} + \mathbf{\Delta}_\mathbf{k}) \simeq \underbrace{\mathbf{e}_i(\mathbf{k})}_{\mathbf{e}_i} + \underbrace{\frac{\partial \mathbf{e}_i(\mathbf{k})}{\partial \mathbf{k}}}_{\mathbf{J}_i} \mathbf{\Delta}_\mathbf{k}. \tag{9}$$

Applying the definition of Ω norm in Eq. (5), substituting Eq. (9) in Eq. (6) and grouping the terms leads to the following expression for $\mathbf{H}$ and $\mathbf{b}$:

$$\mathbf{H} = \sum_i \mathbf{J}_i^T \mathbf{\Omega}_i \mathbf{J}_i \tag{10}$$

$$\mathbf{b} = \sum_i \mathbf{J}_i^T \mathbf{\Omega}_i \mathbf{e}_i \tag{11}$$

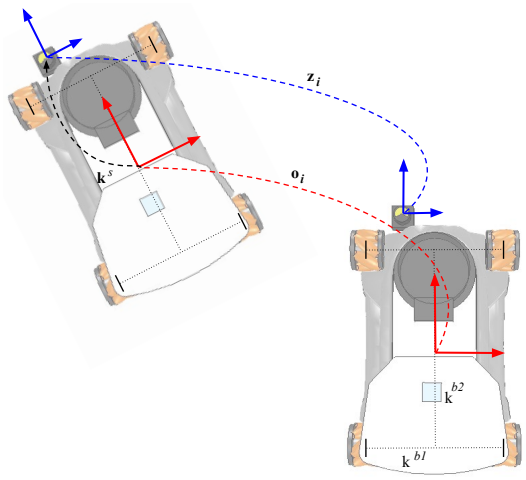


Fig. 3: This figure illustrates the relation between the youbot base and the sensor reference frame. The relative pose of the sensor, with respect to the robot frame, is $\mathbf{k}^{\text{s}}$. While the mobile base moves along the trajectory $\mathbf{o}_i$, the sensor will move along the path $\mathrm{z_i}$ in its own reference frame. As expressed by the kinematics of the robot, k^{bb} is given by the average between the two baseline, *i.e* $k^{\text{bb}} = {}^{(k^{b1}+k^{b2})}/_2$

At each iteration the current guess is refined by adding the computed increments as:

$$\mathbf{k} \leftarrow \mathbf{k} + \mathbf{\Delta}_\mathbf{k}. \tag{12}$$

Under Gaussian assumptions on the measurement noise, the $\mathbf{H}$ matrix represents the inverse covariance of the parameter estimate. Thus by analyzing $\mathbf{H}$ at the equilibrium we can evaluate the quality of the calibration.

In the remainder of this section we provide the details of the two different kinematic models we used in our experiments.

1) Differential Drive: The odometry parameters are the following

$$\mathbf{k}^{\text{o}} = \begin{pmatrix} k^{\text{l}} & k^{\text{r}} & k^{\text{b}} \end{pmatrix}^T. \tag{13}$$

Here k^{l} and k^{r} denote respectively the coefficient between the circumference of the left and right wheel and the encoder ticks per revolution of the wheels' shaft. k^{b} is the distance between the wheels (baseline). $\mathbf{u}_i = \begin{pmatrix} t_i^{\text{l}} & t_i^{\text{r}} \end{pmatrix}$ consists of the tick increments reported by the left and the right wheels, denoted respectively as t_i^{l} and t_i^{r}. The direct kinematics $\mathbf{f}(\mathbf{k}, \mathbf{u}_i)$ can be approximated as:

$$\mathbf{o}_i = \mathbf{f}(\mathbf{k}^{\text{o}}, \mathbf{u}_i) = \begin{pmatrix} \frac{(k^{\text{r}} t_i^{\text{r}} + k^{\text{l}} t_i^{\text{l}})}{2} \\ 0 \\ \frac{k^{\text{r}} t_i^{\text{r}} - k^{\text{l}} t_i^{\text{l}}}{2k^{\text{b}}} \end{pmatrix} \tag{14}$$

2) Omnidirectional Robot: The odometry parameters are

$$\mathbf{k}^{\text{o}} = \begin{pmatrix} k^{\text{fl}} & k^{\text{fr}} & k^{\text{bl}} & k^{\text{br}} & k^{\text{bb}} \end{pmatrix}^T, \tag{15}$$

where the first four parameters are the coefficients for respectively front left, front right, back left and back right wheels. k^{bb} is a coefficient defined as the average between baseline of the wheels and the distance between the front

and rear axes, as shown in Figure 3. The encoder ticks are stored in a 4 dimensional vector $\mathbf{u}_i^T = (t_i^{\mathrm{fl}}\ t_i^{\mathrm{fr}}\ t_i^{\mathrm{bl}}\ t_i^{\mathrm{br}})^T$ as

$$\mathbf{o}_i = \begin{pmatrix} -1/4 & 1/4 & -1/4 & 1/4 \\ 1/4 & 1/4 & -1/4 & -1/4 \\ 1/4k^{\mathrm{bb}} & 1/4k^{\mathrm{bb}} & 1/4k^{\mathrm{bb}} & 1/4k^{\mathrm{bb}} \end{pmatrix} \begin{pmatrix} k^{\mathrm{fl}} t_i^{\mathrm{fl}} \\ k^{\mathrm{fr}} t_i^{\mathrm{fr}} \\ k^{\mathrm{bl}} t_i^{\mathrm{bl}} \\ k^{\mathrm{br}} t_i^{\mathrm{br}} \end{pmatrix} \tag{16}$$

IV. Issues of Calibration using Least Squares

Most of the calibration procedures in the literature rely on least squares minimization. The general procedure consists in moving the robot along a predefined trajectory, while recording its encoder ticks. During the acquisition, a ground truth of the position of the mobile base or its sensors is obtained through some external observer or by algorithms that process only the exteroceptive sensor input.

By knowing the kinematic model is then possible to estimate the motion of the mobile base or its sensors from the measured encoder ticks.

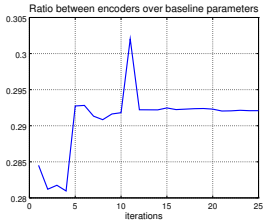
The trajectory is chosen according to some common sense rule in order to provide the least squares engine with sufficient information to produce a reasonable estimate. These trajectories are chosen to be rather long to minimize the risk of having insufficient information and thus having to repeat the task. However, the longer an experiment the higher are the chances that something goes wrong. Additionally, in case of different kinematic models the choice of a good trajectory to follow is left to the user's experience.

Often the parameters are not fully observable while following a specific motion pattern. This leads the system to converge at a value that, while minimizing the error metric in Eq. (6), produces invalid results. In such situations the calibration worsens with the length of the trajectory. This fact is shown in Figure 4, where we show the evolution of the estimate of the baseline and the encoder coefficients of a robot that is constantly rotating on the spot. Note, however that the *ratio* between baseline and ticks is approaching the nominal value.

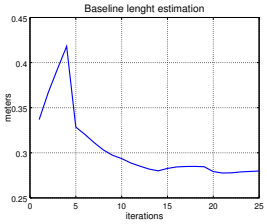
In presence of non-linearities, the success of a least squares procedure greatly depends on the initial guess. However, due to non-observabilities this initial guess might become inconsistent if the trajectory does not allow to explore sufficiently the parameter space.

A solution to this problem is to continuously estimate the quality of the calibration, and choose the motion patterns that improve the estimate. A good calibration is characterized by being

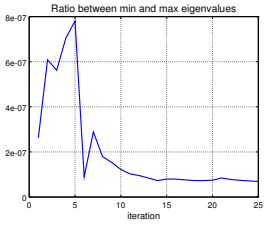
- *accurate*: it is highly consistent with respect to the training data. This can be measured by a low value of the error function in Eq. (6).
- *complete*: it explores all the parameter space, without overfitting the training data. This can be measured by analyzing the ratio between the smaller and bigger eigenvalue of the inverse covariance $\mathbf{H}$ at equilibrium. The higher this value the more "balances" is the uncertainty, thus there are no directions where the uncertainty is substantially higher than others. The eigenvectors of small eigenvalues typically denote a non-observable parameter subspace.



Even if the pure *Rotation* cannot guarantee a full observability on the whole $\mathbf{k}^{\mathrm{o}}$, the ratio between the sum of k^{l} and k^{r} over k^{b} tends to nominal value.



Pure *Rotation* motion does not led to a consistent baseline estimation without performing *Forward* or *Arc* motion at beginning.



In this figure we show the effect of loss of confidence along at least one direction. This brings to an increment of uncertainty and to a inaccurate parameter estimantion.

Fig. 4: The figures show the effects of the pure *Rotate* motion on the $\mathbf{k}^{\mathrm{o}}$ parameters.

When talking of a "balanced" covariance, the reader might notice that the units of measure strongly affect the shape of a covariance. As an example, expressing the the baseline k^{b} in cm. would result in a standard deviation of the estimate that is 100 times bigger than if the baseline is expressed in meters. This phenomena, if we need to estimate one single parameter is not relevant. However, when multiple values need to be simultaneously estimated (e.g. the baseline and the coefficients k^{l} and k^{r}), the effect becomes dominant: a standard deviation of one centimeter for the baseline might be acceptable, while for the encoder coefficients is really bad.

As a rule of thumb, one should "prescale" the units of the parameters to a factor that reflects the desired accuracies. As an example, if we want to estimate the odometry coefficients at an accuracy of $1e-5$ meters and the baseline at $1e-3$ meters of accuracy, we should express the respective units according to these scales. In this way, a standard deviation along the ticks of $1e-5$ will be mapped with an eigenvalue of 1. The same happens for a standard deviation along the baseline of $1e-3$, that results in an eigenvalue of 1 turning the uncertainty ellipse into a circle.

having two quantities expressed one incentimeters and one in meters w

V. Autonomous Calibration

In this section we sketch our method that allows a robot equipped with an exteroceptive sensor to autonomously

choose its trajectory to quickly reduce the uncertainty about its parameters. We remark that the approach is not specific to a differential drive robot and can be easily extended to other classes of wheeled robots such as Hackermann-steering, synchro-drive or omnidirectional platforms.

Our procedure works as follows: The robot is provided with a set of elementary motions $\mathrm{motion}_j \in \mathbf{m}$. Each of these motions obtained by setting a set of constant velocities to each robot wheel, measured in encoder ticks. Typically, each motion allows to observe only a subspace of the parameter set.

When started, the system computes the effect that each motion has on the observability of the parameters. This is done by executing each motion while recording the encoder ticks and the estimated motion from the scan-matcher. After one motion motion_j is executed, our procedure estimates an optimal parameter $\mathbf{k}_j$ set for that motion, together with its inverse covariance $\mathbf{H}_j$. This is done by solving the least squares problem in Eq. (6), by considering only the measurements gathered during the execution of the motion. We call this stage "exploration", as the system estimates the effect that each particular motion has to the parameters.

After the system is done with exploration, the real training begins. The idea is to initialize the calibration procedure from scratch, and compose a trajectory by selecting at each time the motion that improves the estimate of the parameters based on an approximation of the two criteria described above.

The tuple $\langle \mathbf{z}_{1:t}, \mathbf{\Omega}_{1:t}, \mathbf{u}_{1:t} \rangle$ constitutes our motion history up to the current time t, and at the beginning is empty. Here $\mathbf{z}_{1:t}$ and $\mathbf{\Omega}_{1:t}$ are respectively the relative motion observed by the exteroceptive sensor and its inverse covariance, while $\mathbf{u}_{1:t}$ are the encoder differences. Let $\mathbf{k}_t$ and $\mathbf{H}_t$ be the parameter estimate obtained by solving Eq. (6) with the measurements up to time t.

Assuming the current parameter estimate $\mathbf{k}_t$ to be reasonably close to the optimum, and considering the cumulative nature of the $\mathbf{H}$ matrix highlighted by Eq. (10), than selecting the motion motion_j would result in a new $\hat{\mathbf{H}}_t^j$ obtained as:

$$\hat{\mathbf{H}}_t^j = \mathbf{H}_t + \mathbf{H}_j. \tag{17}$$

This means that we can estimate the effect on our calibration of augmenting the current history of measurements with the ones obtained by motion motion_j. The determinant of the predicted information matrix denotes the cumulative information in the estimate. The higher $\det(\mathbf{H}_t^j)$, the more convenient will be executing the motion motion_j.

Thus we select the motion motion_l such that

$$l = \underset{j}{\operatorname{argmax}} \det(\mathbf{H}_t + \mathbf{H}_j) \tag{18}$$

and we perform this motion with the robot by adding to the history of measurements the ones gathered during the current execution of the movement.

This procedure is repeated until the system has reached convergence and the parameter space is sufficiently explored. The first condition is checked by monitoring the evolution of the normalized χ^2 error of the estimate, measured as the value of Eq. (6) analyzed at the current estimate. The second condition is evaluated by monitoring the ratio between the smaller and the larger eigenvalue of the current $\mathbf{H}$.

Fig. 5: This image shows the environment, and its map, where the mobile base executed the unsupervised calibration procedure. The corridor environment has not been augmented with any kind of additional structure

More formally, our approach stops when

$$\frac{1}{t}\sum_{i=1}^{t} \|\mathbf{h}_i(\mathbf{k}_t) \ominus \mathbf{z}_i\|_{\mathbf{\Omega}_i} - \|\mathbf{h}_i(\mathbf{k}_{t-1}) \ominus \mathbf{z}_i\|_{\mathbf{\Omega}_i} < \epsilon \tag{19}$$

and

$$\frac{\lambda_t^{\min}}{\lambda_t^{\max}} > \gamma. \tag{20}$$

In practice at each step our system decides which motion to execute in order to improve its knowledge based on the past experience and on the predicted behavior of the information gathered during the execution of a movement. It stops when it has sufficient knowledge about the parameter space, both in term of completeness and consistency. In our experiments we set the following values for the termination criterion: $\epsilon = 1e-9$ and $\gamma = 1e-2$.

VI. Experiments

We compared our approach with manual calibration methods, both in real world and simulated experiments using two types of mobile bases having different kinematics. We conducted real world experiments using the Kobuki Turtlebot differential drive robot in the department corridors (Figure 5), while we performed simulated experiments on V-REP [10] where we used an ActivMedia Pioneer 2 DX (differential drive) and a KUKA YouBot (omnidirectional). All platforms were equipped with a Hokuyo-URG scanner.

We performed two types of manual calibration for each platform, by executing paths that depend on the kinematics of the robot. Subsequently, we processed the data acquired

along the path by running incrementally the least squares estimation described in Section III. To monitor the evolution of the estimate as more data are gathered, we computed a new solution each time a certain number of new training data become available.

We then executed our automatic calibration for each platform. We selected the basic motions based on the type of kinematics, and let the system learn the effects of each motion. Subsequently, we let the system run and automatically choose the sequence of motions to execute, while recording the training data. After a new motion is executed, our approach provides a new estimate that can be compared with the one achieved by manual calibration after incorporating the same number of samples.

In the remainder of this section we describe the manual calibration paths taken for the different platforms and the basic motions used to start our automatic calibration.

3) Differential Drive: The calibration paths taken by the manual procedure for the differential drive robots are:

- The standard "square" path, where the robot moves along a square having a side of 1 meter, and at the end of each corner rotates of $90°$. After a full square has been completed, the robot continues by reverting the path.
- The standard "8" path, where the robot moves along an 8 shape covering an area of approximately 4 by 2 meters.

The basic motions taken by the differential drive robot are the following:

- *Forward*: obtained by setting the speed of right and left wheel to the same value.
- *Rotate*: obtained by setting the speed of right and left wheel to opposite values.
- *Arc*: obtained by setting the speed of right and left wheel to different values.

Figures 6 and 7 show the results of these experiments for the real world experiments with the turtlebot and for the simulated experiment with the Pioneer 2. In the simulated experiments we had access to the ground truth, thus we perform a direct estimation of the accuracy, whereas in the real world experiments we could only evaluate indirectly the estimate by plotting the reconstructed motion of the sensor under different calibration estimates, as shown in Figure 9

In all cases our approach provided more consistent estimates, and required a lower number of samples to produce usable values.

4) Omnidirectional Robot: The calibration paths taken by the manual procedure for the omnidirectional platform are:

- Double "square" path, where the robot moves along a square having a side of 1 meter. The first time the robot traverses the square, it does not rotate at the corners, while the second time it performs a $90°$ rotation as if it was a differential drive, and the procedure restarts.
- Double "8" shaped path, where the robot moves along an 8 shape covering an area of approximately 4 by 2 meters. The first revolution is done without rotation, while the second revolution is done as if the platform was a differential drive.

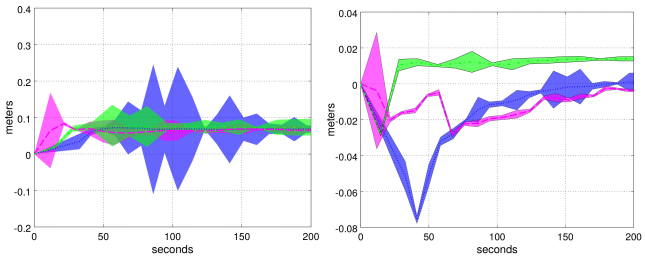


Fig. 6: This image shows the evolution of the parameters estimate for the real differential drive platform (*Turtlebot*). On the left side of the image the evolution of x^{s} is shown, while on the right side the same evolution for y^{s} is shown. In both cases, the uncertainties in the parameter estimate are highlighted by coloured strips. The *green* dashed line represents the evolution observed during the unsupervised procedure. The *blue* dotted line shows the evolution during the calibration following the square-path. Lastly, the *magenta* dash-dotted line shows the evolution of the observed parameter following the eight-shaped path.

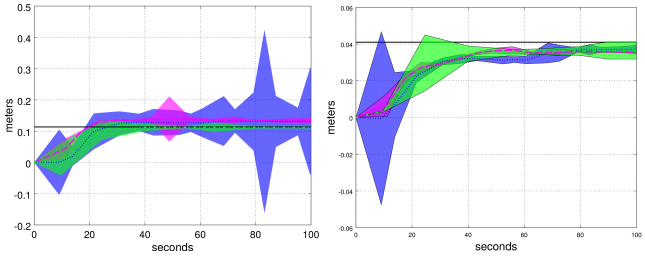


Fig. 7: This image shows the evolution of the parameters estimate for the differential drive platform simulated in V-REP (*Pioneer 2*). On the left side of the image the evolution of x^{s} is shown, while on the right side the same evolution for y^{s} is shown. In both cases, the uncertainties in the parameter estimate are highlighted by coloured strips. The *black* solid line represents the ground truth value (available in simulation) for the observed parameter. The *green* dashed line represents the evolution observed during the unsupervised procedure. The *blue* dotted line shows the evolution during the calibration following the square-path. Lastly, the *magenta* dash-dotted line shows the evolution of the observed parameter following the eight-shaped path.

The basic motions taken by the differential drive robot are the following:

- *Forward*: obtained by setting $t^{\mathrm{fl}} = t^{\mathrm{bl}} = -t^{\mathrm{fr}} = -t^{\mathrm{br}}$ to the same value.
- *Sideward*: obtained by setting $t^{\mathrm{fl}} = t^{\mathrm{fr}} = -t^{\mathrm{bl}} = -t^{\mathrm{br}}$ to the same value.
- *Rotate*: obtained by setting $t^{\mathrm{fl}} = t^{\mathrm{fr}} = t^{\mathrm{bl}} = t^{\mathrm{br}}$ to the same value.
- *Arc*: obtained by setting $t^{\mathrm{fl}} = -t^{\mathrm{br}}$ and $-t^{\mathrm{fr}} = t^{\mathrm{bl}}$ to different values.

Figure 8 shows the results of this simulated experiment. We remark that performing manual calibration for the youbot required repeated trials due to the lack of standard calibration strategies and the increased dimension of the parameter space. The same did not hold for the automatic calibration that quickly converged towards consistent estimates.

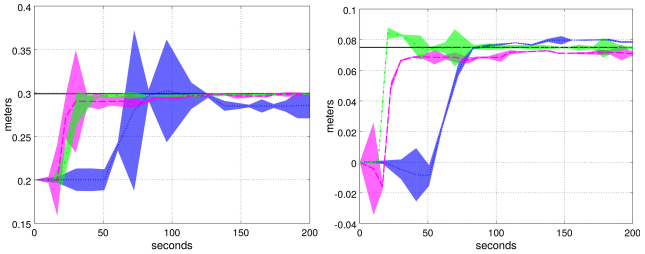


Fig. 8: This image shows the evolution of the parameters estimate for the omnidirectional platform simulated in V-REP (*Kuka Youbot*). On the left side of the image the evolution of x^{s} is shown, while on the right side the same evolution for y^{s} is shown. In both cases, the uncertainties in the parameter estimate are highlighted by coloured strips. The *black* solid line represents the ground truth value (available in simulation) for the observed parameter. The *green* dashed line represents the evolution observed during the unsupervised procedure. The *blue* dotted line shows the evolution during the calibration following the square-path. Lastly, the *magenta* dash-dotted line shows the evolution of the observed parameter following the eight-shaped path.

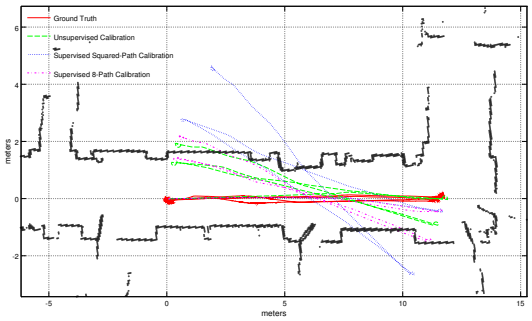


Fig. 9: In this figure we show the odometry estimation of the mobile base. The *red* solid path is the ground truth. The *green* dashed path is the odometry computed with the parameters estimated by the unsupervised calibration. The *blue* dotted path is the odometry computed using the parameter estimate obtained with following a squared path, while the *magenta* dash-dotted path is the odometry computed using the parameters estimated following an 8-shaped path.

A. Behavior of Unsupervised Calibration on a Differential Drive Robot

In this section we shortly report some recurrent behavior we observed during the calibration of the differential drive robot, to better understand the behavior of the approach. Whereas we did not hardcode any heuristic for selecting the motions, we observed that typically the system follows a specific pattern, especially at the beginning. Usually the first choice is to execute the *Forward* motion, since the $\mathbf{H}_{\mathrm{forward}}$ matrix has a higher determinant compared to the others. This results in quickly reducing the uncertainty of the encoder ratios k^l, k^r and of the laser heading θ^s.

After a pair of iterations, the system selects the *Rotate* motion, that allows to estimate the baseline and the $x-y$ displacement of the laser, as highlighed by the eigenvalue decomposition of $\mathbf{H}_{\mathrm{rotate}}$.

Subsequently we were unable to identify a specific pattern in the selection of the motion. We conjuncture it depends highly on the laser position. If the laser is very close to the center of the platform, the $x-y$ coordinates of the laser position are poorly observable through a pure rotation, and the system tends to select the *Arc* motions. Conversely if the laser is located far from the center it tends to select the *Rotate* motion.

VII. Conclusions

In this paper we propose an automatic calibration procedure to determine the kinematic parameters of a mobile robot and its sensors. Our approach dynamically explores the space of possible actions, to determine the effect that each of them has on the parameter estimate. After this exploration, our system chooses which action to take in order to improve the current estimate. To this extent it aims at observing the full parameter space and at reducing the error of the estimate until no further improvements are possible. Our method compares favourably with procedures that require the human assistance both in terms of accuracy and efficiency and dynamically adapts to the robot configuration.

References

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

[2] A. Tapus A. Martinelli, N. Tomatis and R. Siegwart. Simultaneous localization and odometry calibration for mobile robot. In *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 2, pages 1499–1504 vol.2, 2003.

[3] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, , P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *Proc. of the Int. Symposium on Mixed and Augmented Reality (ISMAR)*, 2011.

[4] J. Borenstein, B. Everett, and L. Feng. *Navigating Mobile Robots: Systems and Techniques*. A. K. Peters, Ltd., Wellesley, MA, 1996.

[5] A. Censi. An accurate closed-form estimate of ICP's covariance. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3167–3172, Rome, Italy, April 2007.

[6] F. Chenavier and J. L. Crowley. Position estimation for a mobile robot using vision and odometry. In *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, pages 2588–2593 vol.3, 1992.

[7] A. Kelly. Fast and easy systematic and stochastic odometry calibration. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 4, pages 3188–3194 vol.4, 2004.

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

[9] W. Burgard R. Kuemmerle, G. Grisetti. Simultaneous calibration, localization, and mapping. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, San Francisco, CA, USA, September 2011.

[10] E. Rohmer, S.P.N. Singh, and M. Freese. V-rep: A versatile and scalable robot simulation framework. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1321–1326, Nov 2013.

[11] J. Serafin and G. Grisetti. Nicp: Dense normal based point cloud registration. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 742–749, Sept 2015.

[12] C. R. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainly. *Int. J. Rob. Res.*, 5(4):56–68, December 1986.