Unified Motion-Based Calibration of Mobile Multi-Sensor Platforms with Time Delay Estimation

Bartolomeo Della Corte[1], Henrik Andreasson[2], Todor Stoyanov[2] and Giorgio Grisetti[1]

***Abstract*—The ability to maintain and continuously update geometric calibration parameters of a mobile platform is a key functionality for every robotic system. These parameters include the intrinsic kinematic parameters of the platform, the extrinsic parameters of the sensors mounted on it and their time delays. In this paper, we present a unified pipeline for motion-based calibration of mobile platforms equipped with multiple heterogeneous sensors.**

We formulate a unified optimization problem to concurrently estimate the platform kinematic parameters, the sensors extrinsic parameters and their time delays. We analyze the influence of the trajectory followed by the robot on the accuracy of the estimate. Our framework automatically selects appropriate trajectories to maximize the information gathered and to obtain a more accurate parameters estimate. In combination with that, our pipeline observes the parameters evolution in long-term operation to detect possible values change in the parameters set. The experiments conducted on real data show a smooth convergence along with the ability to detect changes in parameters value. We release an open-source version of our framework to the community.

***Index Terms*—Calibration and Identification.**

I. Introduction

CALIBRATING a robotic platform is a mandatory and a time-consuming procedure, whose outcome may considerably improve the performance of the system. The exact knowledge of the robot kinematic parameters and of the relative poses of the sensors constitutes the basis for the successful application of complex higher level techniques, such as localization and mapping or trajectory planning and tracking.

A reliable dead reckoning system, whose intrinsic accuracy is solely based on the kinematic parameters estimate, constitutes a cornerstone of modern robotic systems. The integrated odometry, usually coupled with inertial sensors, can be used as prior for visual odometry and localization systems. At the same time, it provides a prediction measure for modern trajectory tracking systems, mostly based on model predictive control algorithms. As depicted in Fig. 1, a custom low-cost platform, accurately calibrated in terms of kinematic parameters and sensors poses, is able to build a map using only its dead reckoning system. Such a map might be used as a robust prior for mapping systems.

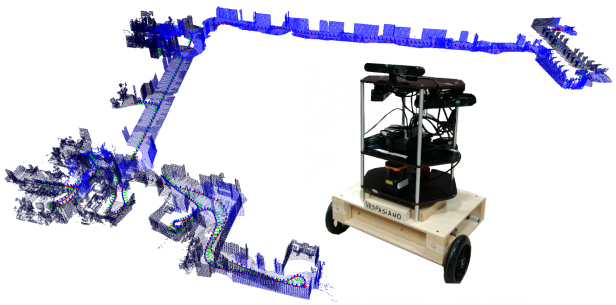


Fig. 1: Dead reckoning map built on a trajectory of approximately 200 meters, after the calibration procedure. The robot used for building the map (shown in the bottom right part of the figure) is a custom built platform, equipped with commercial RGB-D cameras. Such a locally accurate map may aid localization and mapping systems acting as a prior information.

Recently, with the massive deployment of robotic platforms in both social and industrial contexts, the robots themselves are supposed to work seamlessly for hours or days. This makes them subject to possible failures in term of parameters change. Thus, they need a robust and on-the-fly re-calibration procedure.

Two main classes of approaches for motion-based calibration have been proposed: either the platform autonomously executes a calibration procedure, or an expert user drives the platform. In the first case, the calibration outcomes will statistically be more accurate, since the robot performs an ad-hoc calibration routine, as described in our previous work [1]. On the other hand, the robot is forced to stop working, something undesirable for long-term operations. Moreover, such an approach does not allow constant monitoring of the evolution of the parameters. A failure, *e.g.* a change in a parameter value, can happen between two scheduled re-calibrations without being immediately detected. In the second case, when an expert user gathers data and runs the calibration, or in case of a long-term procedure, the user manually selects the portions of the trajectory to feed the calibration system. As a consequence, the outcome is likely to be less accurate, since the chosen trajectory highly affects the quality of the estimate for motion-based calibration problems, as extensively proven [2], [3].

We investigate the problem of calibrating a mobile robotic

Manuscript received: September, 10, 2018; Revised November, 5, 2018; Accepted December, 27, 2018.

This paper was recommended for publication by Dezhen Song upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by *Semantic Robots Research Profile*, funded by the Swedish Knowledge Foundation (KKS).

[1]Bartolomeo and Giorgio are with Department of Computer, Control, and Management Engineering "Antonio Ruberti", Sapienza University of Rome, Italy.

[2]Henrik and Todor are with the Center of Applied Autonomous Sensor Systems (AASS), Örebro University, Sweden.

Digital Object Identifier (DOI): see top of this page.

platform, equipped with several heterogeneous sensors. More concretely our goal is to provide functionality to robotic platforms able to perform both an accurate active calibration for the platform startup and to keep constantly updated the kinematic parameters in a passive way. This allows, at the same time, the detection of possible failures arising either from the platform mechanics, whose effect is reflected on its kinematic parameters, or from the sensor extrinsics.

The main contribution of this paper is the characterization of a methodology for mobile robotic platform calibration, suitable for long-term operations. We constantly keep updated the geometric parameters while looking for fault detection through the parameters evolution observation. The calibration involves both the platform kinematic parameters and the sensors extrinsic ones, embedding in a single optimization algorithm both the platform-sensor constraints and the sensor-sensor constraints. As an additional feature, the estimate includes also the sensors time delays.

An open source implementation of our framework is available online [1].

II. Related Work

The first set of calibration methods listed in the state of the art relies on particular external configurations, *i.e.*, a predefined path to be followed [4] or environment geometry to be known [5]. In the first case, the error to be minimized is formulated as the difference between starting and final positions, limiting the calibration to the platform kinematic parameters only. The latter, instead, expresses the error in terms of the difference between what the sensor mounted on a robot perceives compared with the known structure of the environment. Apart from the intrinsic limitation of the approach presented in [4], the complete knowledge of the environment structure might be challenging in long-term contexts, making the method of Underwood *et al.* [5] unsuitable.

Different approaches try to overcome these limitations by solving the calibration problem using motion-based constraints. Censi et al. [6], [7] formulated the calibration as a maximum likelihood problem, relying on an onboard tracking system for 2D sensors. This solution has been successfully applied also with other kinds of solvers, as in [8] where the authors presented an UKF-based approach that is able to deal with multiple heterogeneous sensors, or in [9] where a Gauss-Helmert Model is used. Martinelli et al. [10] presented an Augmented Kalman Filter (AKF) approach to simultaneously estimate the robot configuration and parameters, embedding the calibration in a Kalman-based SLAM pipeline. This idea has been further investigated in [11], with a simultaneous calibration, localization, and mapping (SCLAM) approach, that incorporates the calibration parameters in a graph-based optimization problem. Despite being widely used and effective, these methods do not take care of how the trajectory followed by the robot may affect the estimate accuracy. A recent work by Huang et al. [3] highlights how the followed trajectory affects the parameter estimates in motion-based calibration problems. Kelly et al. [2] demonstrated how the chosen trajectory affects the quality of the estimate, since, by simply taking arbitrary trajectories, the quality of estimate for different parameters varies substantially.

In our previous work [1], we presented an active calibration approach based on two stages: an *exploration* phase used to discover how the robot motions affect the quality of the parameters estimate, and an *exploitation* stage, where the robot actively performs a set of motions that ensures that all parameters are observed in an equally informative manner. Despite the uniform estimate obtained with this method, its applicability to long-term operations is limited as the platform is required to stop and to actively execute the procedure.

In [12], [13] Taylor et al. proposed a full calibration pipeline for mobile platforms equipped with multiple heterogeneous sensors. Their approach performs the calibration in three stages: the estimate of individual sensor motion; the estimate of offset from motion cues; and use of appearance information to refine the estimate. Their pipeline includes also the timing offset estimate for each sensor. In contrast to our approach, Taylor et al. perform the time delay estimate as a separate step, before performing the rotational and translational estimate.

Furgale et al. [14], [15] overtook the step-by-step procedure for extrinsics and time delay sensor calibration. They presented a general solution to this problem based on maximum-likelihood estimation.

The problem of finding the time offset between sensors has been addressed also in [16], [17], with the particular focus on INS to camera time offset. Ovren et al. [18] calibrated the relative pose between a camera and a gyroscope, estimating also the time scaling and delay through non-linear least squares optimization.

Other recent works focus on using the calibration results on long-term context to perform change detection [19], [20]. The authors propose the division of the gathered data on temporal basis. In this way, they assign to each portion a score based on the covariance matrix obtained from the optimization output. The calibration is then performed with the k highest score portions. Moreover, the portions leading to rank loss are discarded. By analyzing the variation of mean and covariance of the estimate, these approaches are statistically able to detect changes in the parameters.

A. Contributions

In contrast to previous approaches, the methodology presented in this paper:

- simultaneously estimates the platform kinematic parameters, the sensor extrinsics and the sensors time delay;
- comprises the sensor to sensor constraints in the optimization procedure;
- runs in a semi-active way in long-term operation contexts, by discarding portions of a trajectory whose information does not lead to a better parameters estimate while the platform is performing seamlessly its tasks;
- is suitable to be used in a failure detection pipeline through the continuous observation of the parameters evolution.

Moreover, we released an open-source implementation to the community.

[1] http://srrg.gitlab.io/unified-calibration.html

III. MULTI-SENSOR CALIBRATION

Considering a mobile platform equipped with an arbitrary number of sensors N, our goal is to estimate, simultaneously:

- the kinematic parameters of the platform, used to integrate the dead reckoning;
- the extrinsic parameters of each sensor, *i.e.*, the relative poses of the sensors with respect to the platform base;
- the time offset of each sensor.

Thus, our parameters space $\mathbf{x}$ is composed as follows

$$\mathbf{x} = (\mathbf{x}_R^\top \quad \underbrace{\mathbf{x}_{S_1} \quad dt_1}_{Sensor_1} \quad \dots \quad \underbrace{\mathbf{x}_{S_N} \quad dt_N}_{Sensor_N})^\top. \tag{1}$$

- $\mathbf{x}_R \in \mathbb{R}^K$ stores the K kinematic intrinsic parameters of the robotic platform. As an example, for a differential drive platform we would have $\mathbf{x}_R = (k_l, k_r, b)^\top \in \mathbb{R}^3$. k_l and k_r denote respectively the ratios between the circumferences of the left and right wheels and the encoder ticks per revolution of the wheels shaft, while b is the distance between the wheels (baseline).
- $\mathbf{x}_{S_i} = [\mathbf{R}|\mathbf{t}] \in SE(3)$ expresses the extrinsic parameters of the *i-th* sensor in the platform reference frame. $\mathbf{t} \in \mathbb{R}^3$ represents the translation vector and $\mathbf{R} \in SO(3)$ is a rotation matrix.
- $dt_i \in \mathbb{R}$ denotes the time delay of the *i-th* sensor. Its value is relative to the platform timebase and it represents the temporal delay between the sensor and the platform measurements.

In the following, we report the size of the sensor variables to be $[\tilde{\mathbf{x}}_{S_i}^\top \quad dt_i]^\top \in \mathbb{R}^7$, where $\tilde{\mathbf{x}}_{S_i} = t2v(\mathbf{x}_{S_i}) \in \mathbb{R}^6$. The $t2v()$ operator extracts a 6D vector from a transformation matrix, in terms of a translation vector and the first three components of a unit quaternion.

The observations $\mathbf{z}_i^t$, namely observations gathered from the *i-th* sensor at time *t*, are defined according to the kind of constraint. More specifically, for odometry-sensor constraints they express relative motions, while for sensor-sensor constraints they represent raw data correspondences between the sensors.

Let $\mathbf{e}_i^t(\mathbf{z}_i^t, \hat{\mathbf{z}}_i^{t+dt_i}) = \mathbf{e}_i^t(\mathbf{x})$ be a function that computes the error between the actual observation $\mathbf{z}_i^t$ and the predicted observation

$$\hat{\mathbf{z}}_i^{t+dt_i} = f(\mathbf{x}_R, \mathbf{x}_{S_i}, dt_i, \mathbf{u}^t(dt_i)). \tag{2}$$

Here, $\mathbf{u}^t(dt_i)$ represents the input measurement at time t. In the following we specify the nature of the function $f(\circ)$ in Eq. (2) for each type of constraint.

Our goal is to find the set of parameters $\mathbf{x}^*$ that minimizes the negative log-likelihood $\mathbf{F}(\mathbf{x})$ of all the observations

$$\mathbf{F}(\mathbf{x}) = \sum_{t,i} \underbrace{\mathbf{e}_i^t(\mathbf{x})^\top \mathbf{\Omega}_i \mathbf{e}_i^t(\mathbf{x})}_{\mathbf{F}_i^t(\mathbf{x})}, \tag{3}$$

where $\mathbf{\Omega}_i$ is the information matrix of the measurement generating the error $\mathbf{e}_i^t$. Hence, we seek to solve the following equation

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}}\, \mathbf{F}(\mathbf{x}). \tag{4}$$

We solve this problem by employing an iterative least squares approach. At each iteration the algorithm computes a perturbation $\mathbf{\Delta x}$ to refine the current estimate $\mathbf{x}$. $\mathbf{\Delta x}$ minimized a quadratic approximation of Eq. (4)

$$\mathbf{\Delta x}^\top \mathbf{H} \mathbf{\Delta x} + \mathbf{b}^\top \mathbf{\Delta x} + c. \tag{5}$$

The minumum of Eq. (5) is found as $\mathbf{H}\mathbf{\Delta x} = -\mathbf{b}$, where $\mathbf{H} = \sum_i \mathbf{J}_i^\top \mathbf{\Omega}_i \mathbf{J}_i$ and $\mathbf{b} = \sum_i \mathbf{J}_i^\top \mathbf{\Omega}_i \mathbf{e}_i$ are obtained linearing the error function

$$\mathbf{e}_i^t(\mathbf{x} + \mathbf{\Delta x}) \simeq \underbrace{\mathbf{e}_i^t(\mathbf{x})}_{\mathbf{e}_i} + \underbrace{\frac{\partial \mathbf{e}_i^t(\mathbf{x})}{\partial \mathbf{x}}}_{\mathbf{J}_i} \mathbf{\Delta x}, \tag{6}$$

where $\mathbf{\Omega}_i$ denotes the information matrix of the *i-th* measure.

In particular, we simultaneously optimize two types of constraints, *i.e.*, the odometry-sensor constraints and the sensor-sensor constraints. The first kind of constraint involves the error between the trajectory computed by the dead reckoning system and the one provided by an external tracking system working on a sensor, after having expressed the odometry trajectory in sensor reference frame by the sensor extrinsics $\mathbf{x}_{S_i}$. The second kind of constraint, instead, considers the relative error between two sensors, by means of the specific error computed between their raw data.

As derived in [21], the employed method leads to build a unique linear system, whose solution serves as an update for the parameters $\mathbf{x}$. In particular, the contribution of every single constraint feeds a contribution matrix $\mathbf{H} \in \mathbb{R}^{(K+N*7)\times(K+N*7)}$ and a residual vector $\mathbf{b} \in \mathbb{R}^{(K+N*7)}$ of type

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_{[RR]} & \mathbf{H}_{[RS_1]} & \mathbf{H}_{[Rdt_1]} & \dots & \mathbf{H}_{[Rdt_N]} \\ \mathbf{H}_{[S_1R]} & \mathbf{H}_{[S_1S_1]} & \mathbf{H}_{[S_1dt_1]} & & \\ \mathbf{H}_{[dt_1R]} & \mathbf{H}_{[dt_1S_1]} & \mathbf{H}_{[dt_1dt_1]} & & \\ \vdots & & \ddots & & \vdots \\ \mathbf{H}_{[S_NR]} & & & & \mathbf{H}_{[dt_NS_N]} \\ \mathbf{H}_{[dt_NR]} & \dots & & & \mathbf{H}_{[dt_Ndt_N]} \end{pmatrix}$$

$$\mathbf{b} = \begin{pmatrix} \mathbf{b}_{[R]}^\top & \mathbf{b}_{[S_1]}^\top & \mathbf{b}_{[dt_1]}^\top & \dots & \mathbf{b}_{[S_N]}^\top & \mathbf{b}_{[dt_N]}^\top \end{pmatrix}^\top.$$

In the following we report the computation of both the odometry-sensor constraints, either with and without access to platform encoders, and the sensor-sensor constraints. For every kind of constraint we report the error function computation, the Jacobians, and their effect on the system.

A. Odometry-Sensor Constraint with Encoders Access

For an odometry-sensor constraint, every measurement is represented by a relative motion of a sensor mounted on the platform, whose outcome depends on both the robot kinematic parameters and the sensor estimated pose, *i.e.* $\mathbf{z}_i^t \in SE(3)$. Let τ be the sampling period used to get the relative measures.

Referring to Eq. (2), the relative input measurement $\mathbf{u}^t(dt)$ is obtained in terms of platform encoders counts. More specifically, we extract the relative input as follows

$$\mathbf{u}^t(dt) = \mathbf{u}_a^{t+dt+\tau} - \mathbf{u}_a^{t+dt}, \tag{7}$$

using the current estimate of the sensor delay dt. The absolute values $\mathbf{u}_a^t$ in Eq. (7) are obtained by interpolating the input data streams at the required times.

The relative encoder input measure is then used to compute the relative motion of the platform as

$$\mathbf{T}_R = g(\mathbf{u}^t(dt), \mathbf{x}_R) \in SE(3). \tag{8}$$

Here, $g()$ represents the direct kinematic function of the plaform. Thus, the predicted observation is computed as

$$\hat{\mathbf{z}}_i^t = \mathbf{x}_S^{-1}\mathbf{T}_R\mathbf{x}_S \tag{9}$$

Consequently, the error function can be computed as

$$\mathbf{e}_i^t = t2v({\mathbf{z}_i^t}^{-1}\hat{\mathbf{z}}_i^t) \in \mathbb{R}^6 \tag{10}$$

The Jacobian depends on both the kinematics parameters and the sensor ones. Hence, it has a block structure

$$\mathbf{J}_i = \left[\underbrace{\frac{\partial \mathbf{e}_i}{\partial \mathbf{x}_R}}_{\mathbf{J}_R} \quad \underbrace{\frac{\partial \mathbf{e}_i}{\partial \mathbf{x}_S}}_{\mathbf{J}_S} \quad \underbrace{\frac{\partial \mathbf{e}_i}{\partial dt}}_{\mathbf{J}_{dt}}\right] \in \mathbb{R}^{6\times(7+K)} \tag{11}$$

where the sensor Jacobian $\mathbf{J}_S$ is obtained considering the partial derivatives with respect to the vector $\mathbf{v}_S = t2v(\mathbf{x}_S)$.

By using the Jacobian components, the relative blocks of $\mathbf{H}$ and $\mathbf{b}$ are updated incrementally as follows:

$$\begin{array}{ccc} \mathbf{H}_{[RR]} \mathrel{+}= \mathbf{J}_R^\top\mathbf{\Omega}_i\mathbf{J}_R & \mathbf{H}_{[RS]} \mathrel{+}= \mathbf{J}_R^\top\mathbf{\Omega}_i\mathbf{J}_S & \mathbf{H}_{[Rdt]} \mathrel{+}= \mathbf{J}_R^\top\mathbf{\Omega}_i\mathbf{J}_{dt} \\ \mathbf{H}_{[SR]} \mathrel{+}= \mathbf{J}_S^\top\mathbf{\Omega}_i\mathbf{J}_R & \mathbf{H}_{[SS]} \mathrel{+}= \mathbf{J}_S^\top\mathbf{\Omega}_i\mathbf{J}_S & \mathbf{H}_{[Sdt]} \mathrel{+}= \mathbf{J}_S^\top\mathbf{\Omega}_i\mathbf{J}_{dt} \\ \mathbf{H}_{[dtR]} \mathrel{+}= \mathbf{J}_{dt}^\top\mathbf{\Omega}_i\mathbf{J}_R & \mathbf{H}_{[dtS]} \mathrel{+}= \mathbf{J}_{dt}^\top\mathbf{\Omega}_i\mathbf{J}_S & \mathbf{H}_{[dtdt]} \mathrel{+}= \mathbf{J}_{dt}^\top\mathbf{\Omega}_i\mathbf{J}_{dt} \end{array}$$

and

$$\mathbf{b}_{[R]} \mathrel{+}= \mathbf{J}_R^\top\mathbf{\Omega}_i\mathbf{e}_i \quad \mathbf{b}_{[S]} \mathrel{+}= \mathbf{J}_S^\top\mathbf{\Omega}_i\mathbf{e}_i \quad \mathbf{b}_{[dt]} \mathrel{+}= \mathbf{J}_{dt}^\top\mathbf{\Omega}_i\mathbf{e}_i.$$

B. Odometry-Sensor Constraint without Encoders Access

When using commercial platforms, the encoder tick counts might be unaccessible, while the odometry is already provided. Once extracted the relative measure of the platform odometry there are two options, *i.e.* either we compute the inverse kinematics to get the encoder tick counts or we look for the systematic error affecting the odometry in terms of a 3×3 matrix.

In the first case, we fix a reasonable value for the kinematic parameters of the platform, and use the inverse kinematics (IK) to get the encoder tick counts we need as input. Using our method we then obtain a corrected value of the platform kinematic parameters as well as the sensor extrinsics. While the latter can be further used in higher-level tasks, the platform parameters depend on the initial guess used in the IK and should not be further used.

On the other hand, we can estimate a correction matrix instead of the platform kinematic parameters:

$$\mathbf{x}_R = (x_{11} \quad x_{12} \quad x_{13} \quad x_{21} \quad x_{22} \quad x_{23} \ldots x_{33})^\top \tag{12}$$

that we can use to correct the odometry input measurements $\mathbf{u}^t(dt)$ as

$$\mathbf{T}_R = g(\mathbf{u}^t(dt), \mathbf{x}_R) = \begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{pmatrix} \mathbf{u}^t(dt), \tag{13}$$

where $\mathbf{u}^t(dt) = [\delta_x^t(dt) \quad \delta_y^t(dt) \quad \delta_\theta^t(dt)]^\top$ represents the relative odometry. From here on, we are connected to Eq. (8).

C. Sensor-Sensor Constraint

In case of sensor to sensor constraint, two different situations arise, *i.e.*, homogeneous sensors and heterogeneous sensors. In the first case, depending on the specific sensor mode, every kind of least squares based registration algorithm can suffice, either indirect [22] or direct [23]. In the second case, instead, a data preprocessing is needed to find associations between data coming from heterogeneous sensors. The most common cases are the registration of data between cameras and Lidars [24]–[26], and between IMU and cameras [14], [16], [27].

As a general example, we report the constraint computed with a standard ICP formulation. Same holds for all the other least squares based registration algorithms reported in this section. We decided not to focus on specific registration algorithms for cross-sensor calibration, since there is already ample literature describing specific cross-sensor calibration procedures.

Given two sensors S_i and S_j, whose time delays are respectively denoted as dt_i and dt_j, we use their observed point clouds $\mathbb{P}_i^{t+dt_i}$ and $\mathbb{P}_j^{t+dt_j}$ to generate measurements. Here we use the time delay of the sensors in order to synchronize the data streams and to select corresponding point clouds.

Let $\mathbb{C} = \{p_i^c, p_j^c\}_{c=1\ldots N}$ the corresponding points of the two clouds, we compute the error for each correspondence as

$$\mathbf{e}_c = \hat{\mathbf{z}}_i(\mathbf{x}_{S_i}, p_i^c) - \hat{\mathbf{z}}_j(\mathbf{x}_{S_j}, p_j^c), \tag{14}$$

where the prediction function $\hat{\mathbf{z}}(\mathbf{x}_{S_i}, p_i^c)$ uses the current sensor extrinsics estimate $\mathbf{x}_{S_i}$ to express the point p_i^c in the platform reference frame. The error will lie in $\mathbb{R}^3$ in case of standard ICP, or in higher spaces in case of augmented measurement [28], [29].

The jacobian for each correspondence has the same structure depicted in Eq. (11), with the partial derivatives computed both with respect to the sensor extrinsics and to their time delays.

IV. Estimate Accuracy

In this section we first provide some general considerations about motion-based extrinsics calibration. These have to be considered as assumptions the system works on. Later, we present our method for measuring the accuracy of the calibration outcome. Finally, we show the direct relation between the calibration accuracy and the trajectory taken while gathering data.

A. General Considerations

To measure the accuracy of the estimate in motion-based extrinsics calibration, some considerations have to be taken into account. For *odometry-sensor* constraints, the quality of the pose tracking system acting on a sensor affects the accuracy of the calibration outcome. In fact, the observation $\mathbf{z}_i^t$ acts as reference value to the prediction of Eq. (2). Thus, the calibration outcome can be as accurate as is possible from

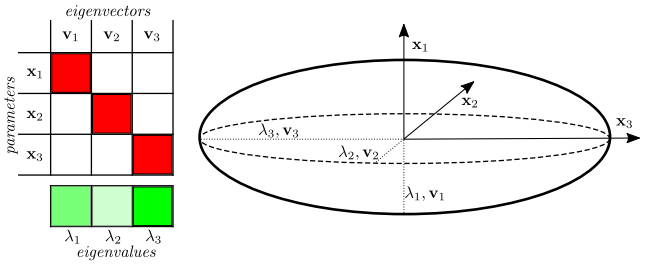
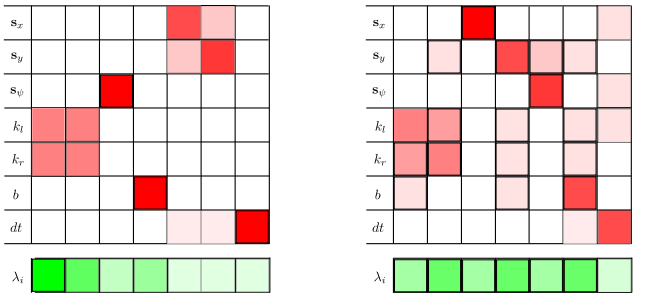


Fig. 2: Right-side of the figure depicts $\mathbf{H}$ matrix as an ellipsoid spanning over the parameters space $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$, highlighting its eigenvectors with the respective eigenvalues. This kind of visualization rapidly becomes impractical when increasing the parameters space dimension, *i.e.* dealing with hyperellipsoids. To this extent, we visualize the amount and the distribution of the information in a grid fashion (left-side). Each eigenvector is represented by a column of the grid, whose cell values (the darker the color, the higher the value) indicate the pointed directions in parameters space. In this particular example, the highest eigenvalue λ_3 is the one associated with eigenvector $\mathbf{v}_3$, that is pointing exclusively in the direction of parameter $\mathbf{x}_3$. This indicates a high confidence in the estimate of this parameter.



(a) Straight Motion (b) Rotation Motion

Fig. 3: Observation of the uncertainty of the calibration process while using exclusive motions, *i.e.* straight motions and pure rotation motions, with a differential drive platform equipped with a 2d sensor. In the figures, the darker the color, the higher the value, *e.g.* in Fig. 3a the highest eigenvalue is the one related the eigenvector pointing mostly in the wheel radii directions. This expresses a high confidence in the estimate of these parameters.

the pose tracking systems accuracies. Intuitively, in case of offline batch calibration the use of optimized trajectories from full SLAM pipelines mitigates this effect. In case of online calibration, though, the presence of sudden changes of pose due to online trajectory optimization, as an effect of local bundle adjustment or loop closures, may introduce high levels of noise. The reader can image the case of one of these conditions caught while sampling a relative datum $\mathbf{z}_i^t$, whose smoothness in terms of trajectory is lost. In these cases, a visual odometry tracking system has to be preferred.

Furthermore, specific adaptations have to be done in case of sensor clock drift, rolling shutter camera or lidar motion distorsion. Where possible, Eq. (2) should model them. Finally, as a rule of thumb, the individual sensors intrinsics have to be calibrated before running the extrinsics calibration.

B. Measuring the Accuracy

The optimization procedure described in Sec. III provides a perturbation vector $\mathbf{\Delta x}$ used to update the current estimate of the parameters. The accuracy of the output vector itself is directly correlated to the sequence, type and quality of the measurements with which the optimization has been supplied.

By analyzing the $\mathbf{H}$ matrix obtained while using a portion of trajectory it is possible to evaluate how informative the portion is and which dimensions of the estimate have been observed more. In other words, $\mathbf{H}$ stores the information about the estimate. To remove the effect of different units, the $\mathbf{H}$ matrix has to be normalized [19]. Being by construction a positive definite matrix, its eigenvalues are real and positive, and by analyzing them we can provide a measure of the *uniformity* of the estimate. As illustrated in Fig. 2, the eigenvectors of the hyperellipsoid spanned by $\mathbf{H}$, weighted by the respective eigenvalues, provide an overview of the estimate accuracy.

In particular we express uniformity in terms of the ratio between the lowest λ_{min} and the highest λ_{max} eigenvalues of the information matrix. More formally, let $\eta(\mathbf{H}) \in \mathbb{R}$ be this ratio

$$\eta(\mathbf{H}) = \frac{\lambda_{min}}{\lambda_{max}}. \tag{15}$$

The ideal condition is the one with $\eta(\mathbf{H})$ equal to 1. Such a result can be obtained if all the information directions, represented by its eigenvectors, are uniformly explored. A low value of $\eta(\mathbf{H})$ expresses a high difference in the uncertainty of the parameters estimate.

Nevertheless, expressing the goal of a robust calibration as maximizing the outcome of Eq. (15) may lead the estimate to be performed with an insufficient amount of information (expressed by the volume of the hyperellipsoid spanned by $\mathbf{H}$). Intuitively, a low magnitude information matrix can easily obtain a high value for $\eta(\mathbf{H})$, while being at the same time insufficient for a good estimate. To avoid such a case, we add the additional goal of increasing the magnitude of the information matrix $\mathbf{H}$, expressed by its determinant. Let $\phi(\mathbf{H}) \in \mathbb{R}$ define the product between the information magnitude and the uniformity value, normalized by the number of samples N

$$\phi(\mathbf{H}) = \frac{\det(\mathbf{H})\eta(\mathbf{H})}{N}. \tag{16}$$

Thus, we define the calibration objective as maximizing the value of Eq. (16).

When performing a background calibration for long-term operations, we get the matrix $\mathbf{H}_i$ for every portion of trajectory. In some cases, outlined in Sec. IV-C, the matrix $\mathbf{H}_i$ may be rank deficient, depending on the trajectory taken while gathering the data. Our policy is to discard those portions whose $\phi(\mathbf{H}_i)$ is below a certain threshold. In this way we update the parameters by using only highly informative and uniform portions.

C. Calibration Trajectory

Depending on the kinematics of the platform to calibrate, the uncertainties observed during operations may substantially vary depending on the kind of motions executed.

As an example, Fig. 3 reports the uncertainty observed while calibrating a differential drive equipped with a 2d LiDAR

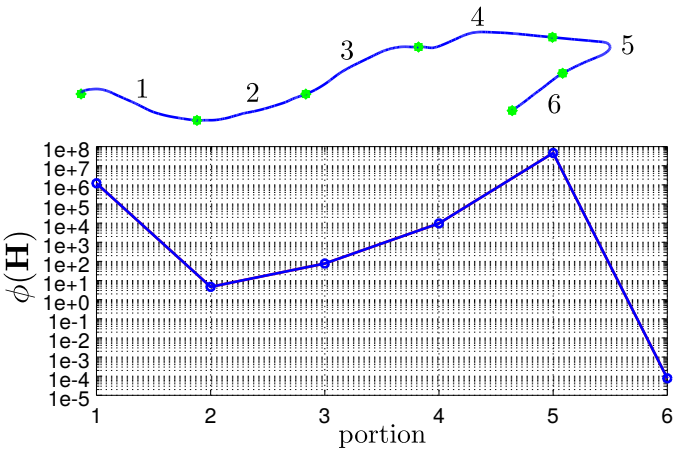


Fig. 4: Variation of the value of $\phi(\mathbf{H})$ on KITTI sequence 10 by selecting portions of trajectory of 200 samples, resulting in portions of approximately 20 seconds each.

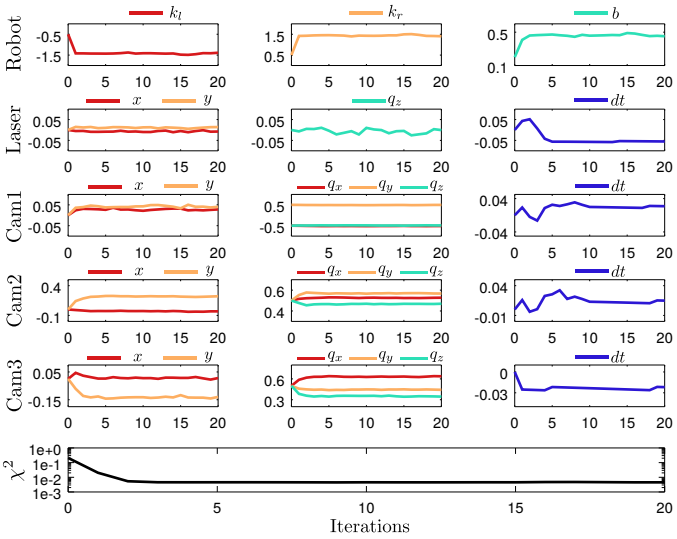


Fig. 5: Dynamics of the parameters estimate when simultaneously calibrating the kinematics parameters of a differential drive (k_l, k_r, b), the extrinsics of several sensors and their time delays, whose values are relative to the platform timebase. Last row depicts the evolution of the quadratic error. All the plots report the evolution for 20 iterations of the least squares solver (best viewed on screen).

using exclusive motions (*e.g.*, straight, rotate). Intuitively, using only straight paths in the calibration process leads to high uncertainty on the sensors positions and on the platform baseline, as shown in Fig. 3a. Conversely, as reported in Fig. 3b, using only pure rotation motions increases the uncertainty on the sensors orientations and on the wheel radii of the platform. Arc turns are usually preferred, since they lead to a relatively uniform exploration of the parameter space. The eight-shaped path is usually recommended in case of manual calibration with differential drive platforms. Another argument to use varied platform motions is that exclusively straight trajectories do not allow a proper observation of the sensor delay parameter.

Fig. 4 shows an example of the impact of trajectory taken on the calibration process. In particular we calibrated the extrinsics of the sensors mounted on a car in sequence 10 of the KITTI dataset [30]. We used ProSLAM [31] and IMLS-SLAM[2] [32] as tracking systems respectively for the stereo camera and velodyne, calibrating the extrinsics with respect to the ground truth trajectory provided by the IMU/GPS system.

We split the dataset sequence in six portions consisting of 200 samples each (approximately 20 seconds each), analyzing the output Hessian matrix. By observing the value of $\phi(\mathbf{H})$ it is easily detectable how the most informative portions are the ones consisting of multiple turns $\{1, 4, 5\}$, while the remaining provide high uncertainty in the parameters estimate.

V. Change Detection

When performing time demanding tasks, resulting in what are called long-term operations, parameter change detection represents a fundamental feature for a calibration system. If a geometric parameter changes its value without being detected, all the higher level components, *e.g.* the units of a SLAM pipeline, may fail.

A change in the parameters usually happens in two ways: either gradually by wear of mechanical components; or suddenly due to a rapid change in sensors poses. In both cases, depending on the specific task the system is carrying on, it is desirable either to re-calibrate and continue, or to stop the current task and physically correct the change.

The detection of such changes is based on the constant observation of the uncertainty evolution on the parameters estimate, as described in *Keivan et al.* [19]. We generalize the approach of *Keivan et al.* by including the kinematic parameters of the platform and by discriminating the trajectory portion in a systematic way (Sec. IV-C). In particular, given a calibrated platform, whose parameters mean μ_0 and covariance Σ_0 (obtained as the inverse of the relative $\mathbf{H}$ matrix) have been estimated before starting the task, we sample portions of the trajectory the platform is following on a temporal basis. For each portion we evaluate its normalized amount of information as in Eq. (16). We discard the portion in case the computed value is lower than a specified threshold. Otherwise, we compute the mean μ and covariance Σ from the portion. As in [19], [20], we solve a multivariate Behrens-Fisher problem [33] to define the similarity of the two distributions, computing an approximated F-distribution. We computed the p-value from this distribution to define a similarity value.

If two consecutive portions present a low value of similarity, compared with the current estimate of the parameters, we start a re-calibration due to the detected change. More specifically, we discriminate the behaviour of our system depending on the outcome of p-value:

- if p-value is close to 1, the two distributions are considered similar and the current portion of data is used to further refine the parameters estimate;
- otherwise, if p-value is less than a predefined threshold, we trigger a re-calibration process since a change has been detected.

[2]We are grateful to Jean-Emmanuel Deschaud for providing us the output trajectories of IMLS-SLAM on the KITTI dataset.

VI. Performance Evaluation

The reported experiments have been designed to prove the effectiveness of the presented techniques for calibration and change detection.

In our first experiment we used a custom differential drive platform equipped with a 2D Hokuyo-URG scanner and three RGB-D cameras. The custom platform provides encoder readings at ~ 50Hz, while the laser scanner and the cameras provide raw data respectively at ~ 40Hz and ~ 30Hz. The intrinsics of the cameras have been estimated with a standard camera resectioning tool. Furthermore, we corrected the sensor distorsion as described in [34].

We calibrated the kinematic parameters of the platform, and the extrinsics of the four sensors mounted on it, as described in Sec. III. To validate the convergence of the presented methodology, we gathered data following an eight-shaped path several times, to run a one shot optimization. We used MPR [23] to get the RGB-D camera position tracking and a scan matching system[3] for the laser position tracking. Fig. 5 depicts the results of the calibration process. As initial guess we set all the sensors positions to the origin and their time delays to zero. The camera orientations where roughly initialized as ($q_x = -0.5$, $q_y = 0.5$, $q_z = -0.5$) for the camera with the *y-axis* pointing down and ($q_x = 0.5$, $q_y = 0.5$, $q_z = 0.5$) for the two cameras with *y-axis* pointing up. We initialized the kinematic parameters of the differential drive as ($k_l = -0.5$, $k_r = 0.5$, $b = 0.2$). As shown in the last row of Fig. 5, the dynamic of the quadratic error presents a smooth convergence, even with a rough parameters initialization.

To prove the effectiveness of the calibration on the kinematic parameters of the platform, we run again the calibration, this time using the active routine described in [1], adapted with the accuracy measure described in Sec. IV-C to score the portions of trajectory. Once calibrated, we ran the platform for a path of approximatively 200 meters, reconstructing its odometry by integrating the encoder measurements using the calibrated parameters. As term of comparison we used the trajectory generated by a state of the art 2d mapper [35] acting on the laser scanner, comparing it with the dead reckoning system of the calibrated platform. Fig. 6 depicts the two trajectories. The dead reckoning trajectory is obtained integrating the encoder counts, using the estimated kinematic parameters, and transforming the obtained trajectory in the laser reference frame using the estimated extrinsics. In Tab. I we report the relative pose error between the two trajectories, evaluating the error per meter. As expected, a robust calibration procedure leads to a reasonably accurate dead reckoning system, that results to be very helpful as prior for more complex mapping and localization systems. To further stress this concept, we report in Fig. 1 the reconstructed 3d map using the estimated kinematic parameters of the platform and extrinsics of the front RGB-D camera.

We performed a change detection experiment on the KITTI dataset [30], using sequence 00. We choose this sequence for the diversity of the trajectories taken by the platform. The car in the dataset is equipped with a stereo rig, obtained with two

[3]https://gitlab.com/srrg-software/srrg_scan_matcher_ros

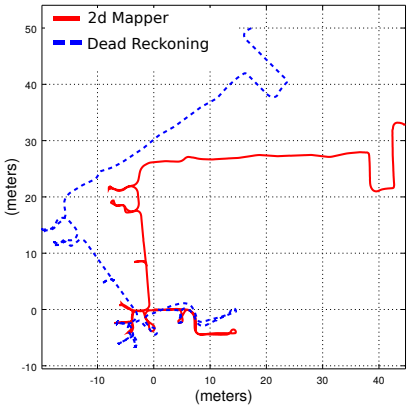


Fig. 6: Direct comparison of the output trajectory obtained with a 2d laser-based mapper [35] and the dead reckoning trajectory of a calibrated custom platform.

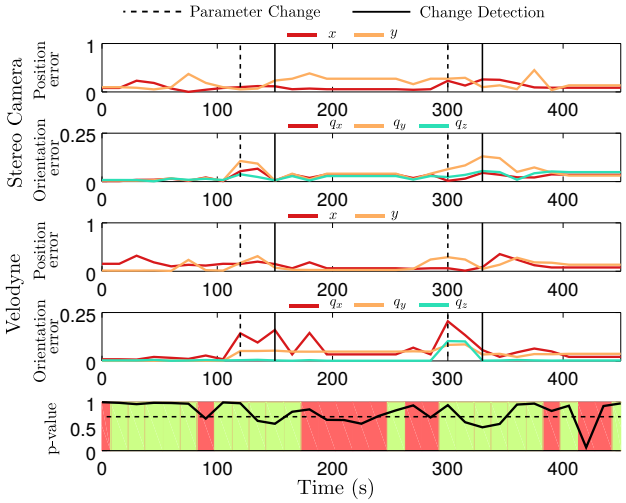


Fig. 7: Change detection experiment on KITTI sequence 00. First two rows report the error in the evolution of the parameters estimate for the stereo camera, while third and fourth rows depict the error in the evolution of the Velodyne parameters. Last row shows the evolution of the p-value either in high informative portions (highlighted in green) and in low informative ones (in red), coherently with the computed value of $\phi(\mathbf{H})$ (best viewed on screen).

Point Grey Flea 2, a Velodyne HDL-64 laser scanner and a Inertial Navigation System (INS) to provide the ground truth. In our experiment we used the INS trajectory as platform odometry and calibrate the sensors' extrinsics with respect to it. All the data are provided at 10Hz, without timestamps. Hence, we locked the time delays of the sensors to zero. We used ProSLAM [31] and IMLS-SLAM [32] as position tracking systems respectively on the stereo camera and on the laser scanner. They are reported to provide impressive accuracy on KITTI dataset: IMLS-SLAM has an average translation error of 0.61%, while ProSLAM has an average translation error of 1.37%. We computed initial mean μ_0 and covariance Σ_0 by running the calibration on the other sequences of the dataset. We simulated two parameters changes at time $t_1 = 120s$ and $t_2 = 300s$, shifting the computed trajectories of the following amounts $dt_1 : (dx = 0, dy = 0.08, dz = 0, dq_x = 0.15, dq_y = -0.05, dq_z = 0)$ and $dt_2 : (dx = 0, dy = -0.05, dz =$

	Translation	Rotation
RMSE	0.0328 m	0.619 deg
std. dev.	0.0121 m	0.421 deg

TABLE I: Relative Pose Error per meter

$0, dq_x = 0.05, dq_y = -0.05, dq_z = -0.1$). We set the change detection threshold for the p-value at 0.7. Fig. 7 depicts the results of this experiment in terms of estimate error. As shown, the two changes are detected respectively at $t = 150s$ and $t = 330s$. In the last row of Fig. 7, we reported the evolution of p-value. We reported in red the portions of the dataset where $\phi(\mathbf{H}_i)$ was below the minimum information required ($1e3$). As can be observed at time $t = 230s$ and $t = 430$s, some false positives are avoided by the analysis of the trajectory taken.

VII. Conclusion

In this paper, we presented a unified methodology for calibrating a mobile platform equipped with several heterogeneous sensors. Our method embeds in a single optimization problem multiple constraints. It simultaneously estimates the platform kinematic parameters, and the sensors extrinsic ones. The latter include the sensors time delay, relative to the platform timebase. Moreover, we highlighted the importance of the trajectory taken for motion-based calibration, providing a quality measure of the trajectory itself. We used this measure to discriminate portions of trajectory, so to obtain a highly accurate parameters estimate by using the more informative ones. In addition, we show how the presented methodology is suitable for change detection pipelines, to observe the parameters value variation. We implemented and evaluated our approach on real data, releasing an open-source version to the community.

References

[1] M. Di Cicco, B. Della Corte, and G. Grisetti, "Unsupervised calibration of wheeled mobile platforms," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, May 2016, pp. 4328–4334.

[2] A. Kelly, "Fast and easy systematic and stochastic odometry calibration," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, vol. 4, Sept 2004, pp. 3188–3194 vol.4.

[3] K. Huang and C. Stachniss, "On geometric models and their accuracy for extrinsic sensor calibration," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018. [Online]. Available: http://www.ipb.uni-bonn.de/pdfs/huang2018icra.pdf

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

[5] J. Underwood, A. Hill, and S. Scheding, "Calibration of range sensor pose on mobile platforms," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Oct 2007, pp. 3866–3871.

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

[7] A. Censi, A. Franchi, L. Marchionni, and G. Oriolo, "Simultaneous calibration of odometry and sensor parameters for mobile robots," *IEEE Trans. on Robotics (TRO)*, vol. 29, no. 2, pp. 475–492, April 2013.

[8] S. Schneider, T. Luettel, and H. J. Wuensche, "Odometry-based online extrinsic sensor calibration," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Nov 2013, pp. 1287–1292.

[9] K. Huang and C. Stachniss, "Extrinsic multi-sensor calibration for mobile robots using the gauss-helmert model," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Sept 2017.

[10] A. Martinelli, N. Tomatis, A. Tapus, and R. Siegwart, "Simultaneous localization and odometry calibration for mobile robot," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, vol. 2, Oct 2003, pp. 1499–1504 vol.2.

[11] R. Kümmerle, G. Grisetti, and W. Burgard, "Simultaneous calibration, localization, and mapping," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Sept 2011, pp. 3716–3721.

[12] Z. Taylor and J. Nieto, "Motion-based calibration of multimodal sensor arrays," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, May 2015, pp. 4843–4850.

[13] ——, "Motion-based calibration of multimodal sensor extrinsics and timing offset estimation," *IEEE Trans. on Robotics (TRO)*, vol. 32, no. 5, pp. 1215–1229, Oct 2016.

[14] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Nov 2013.

[15] J. Rehder, R. Siegwart, and P. Furgale, "A general approach to spatiotemporal calibration in multisensor systems," *IEEE Trans. on Robotics (TRO)*, vol. 32, no. 2, pp. 383–398, April 2016.

[16] M. Fleps, E. Mair, O. Ruepp, M. Suppa, and D. Burschka, "Optimization based imu camera calibration," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Sept 2011.

[17] J. Kelly, N. Roy, and G. S. Sukhatme, "Determining the time delay between inertial and visual sensor measurements," *IEEE Transactions on Robotics*, vol. 30, no. 6, pp. 1514–1523, Dec 2014.

[18] H. Ovren and P. Forssen, "Gyroscope-based video stabilisation with auto-calibration," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, May 2015, pp. 2090–2097.

[19] N. Keivan and G. Sibley, "Online slam with any-time self-calibration and automatic change detection," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, May 2015, pp. 5775–5782.

[20] F. Nobre, C. R. Heckman, and G. T. Sibley, "Multi-sensor slam with online self-calibration and change detection," in *Proc. of the Intl. Sym. on Experimental Robotics (ISER)*, 2016, pp. 764–774.

[21] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based SLAM," *IEEE Trans. on Intelligent Transportation Systems Magazine*, vol. 2, pp. 31–43, 2010.

[22] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp." in *Proc. of Robotics: Science and Systems (RSS)*, vol. 2, no. 4, 2009.

[23] B. Della Corte, I. Bogoslavskyi, C. Stachniss, and G. Grisetti, "A General Framework for Flexible Multi-Cue Photometric Point Cloud Registration," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[24] A. Napier, P. Corke, and P. Newman, "Cross-calibration of push-broom 2d lidars and cameras in natural scenes," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, May 2013.

[25] R. Wolcott and R. Eustice, "Visual localization within lidar maps for automated urban driving," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Sept 2014, pp. 176–183.

[26] T. Caselitz, B. Steder, M. Ruhnke, and W. Burgard, "Monocular Camera Localization in 3D LiDAR Maps," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2016.

[27] C. Le Gentil, T. Vidal-Calleja, and S. Huang, "3d lidar-imu calibration based on upsampled preintegrated measurements for motion distortion correction," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, May 2018, pp. 2149–2155.

[28] J. Serafin and G. Grisetti, "Using extended measurements and scene merging for efficient and robust point cloud registration," *Journal on Robotics and Autonomous Systems (RAS)*, vol. 92, no. C, 2017. [Online]. Available: https://doi.org/10.1016/j.robot.2017.03.008

[29] S. Jia, M. Ding, G. Zhang, and X. Li, "Improved normal iterative closest point algorithm with multi-information," in *Proc. of the IEEE Intl. Conf. on Information and Automation (ICIA)*, 2016, pp. 876–881.

[30] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[31] D. Schlegel, M. Colosi, and G. Grisetti, "ProSLAM: Graph SLAM from a Programmer's Perspective," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[32] J. Deschaud, "IMLS-SLAM: scan-to-model matching based on 3D data," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[33] J. Park and B. Sinha, "Some aspects of multivariate behrens-fisher problem," *Calcutta Statistical Association Bulletin*, 2009.

[34] M. Di Cicco, L. Iocchi, and G. Grisetti, "Non-parametric calibration for depth sensors," *Journal on Robotics and Autonomous Systems (RAS)*, vol. 74, pp. 309–317, 2015.

[35] M. Lázaro, R. Capobianco, and G. Grisetti, "Efficient Long-term Mapping in Dynamic Environments," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2018.