Spatiotemporal Multisensor Calibration via Gaussian Processes Moving Target Tracking

Juraj Peršić, *Student Member, IEEE,* Luka Petrović, *Student Member, IEEE,* Ivan Marković, *Member, IEEE,* Ivan Petrović, *Member, IEEE,*

***Abstract*—Robust and reliable perception of autonomous systems often relies on fusion of heterogeneous sensors, which poses great challenges for multisensor calibration. We propose a method for multisensor calibration based on Gaussian processes (GPs) estimated moving target trajectories, resulting with spatiotemporal calibration. Unlike competing approaches, the proposed method is characterized by: (i) joint multisensor on-manifold spatiotemporal optimization framework, (ii) batch state estimation and interpolation using GPs, and (iii) computational efficiency with O(n) complexity. It only requires that all sensors can track the same target. The method is validated in simulation and real-world experiments on five different multisensor setups: (i) hardware triggered stereo camera, (ii) camera and motion capture system, (iii) camera and automotive radar, (iv) camera and rotating 3D lidar and (v) camera, 3D lidar and motion capture system. The method estimates time delays with the accuracy up to a fraction of the fastest sensor sampling time, outperforming a state-of-the-art ego-motion method. Furthermore, the paper is complemented by an open source toolbox implementing the calibration method available at bitbucket.org/unizg-fer-lamor/calirad.**

***Index Terms*—multisensor calibration, temporal calibration, Gaussian processes**

I. Introduction

MODERN autonomous robotic systems navigate through the environment using information gathered by various sensors. To process the gathered information, robots must rely on accurate sensor models and often fuse information from multiple sensors to improve performance. For sensor fusion, appropriate knowledge of both temporal and spatial relations between the sensors is required, which can be challenging when working with heterogeneous sensor systems, since sensors can operate based on various physical phenomena, while providing measurements asynchronously with different frame rates. The described challenges are addressed by sensor calibration, which can be divided into intrinsic, extrinsic also referred as spatial, and temporal calibration. The intrinsic calibration is related to individual sensors as it provides parameters for sensor models. The task of the extrinsic calibration is to find homogeneous transforms relating multiple sensors, while temporal calibration aims to find relation between the individual sensor clocks.

The sensor calibration approach for a particular problem depends on multiple factors, e.g., the type of involved sensors, overlapping field of view, required degree of calibration accuracy, Nevertheless, to calibrate multiple sensors extrinsically and temporally, we need to perform correspondence registration in the sensor data, which is later used to form an optimization criterion. The correspondences can originate from a designed target, yielding the target-based methods [1], [2], or from the environment itself, as in the case of the so-called targetless methods [3], [4]. For example, odometry-based methods are a special class of targetless methods suitable for online application and are based on leveraging the environment to estimate ego-motion and calibrate the multisensor system [5], [6]. The concept of sensor calibration by aligning trajectories of moving targets received most attention in the target-based calibration of depth sensors [7]–[9], and calibration of cameras, depth sensors, and lidars by exploiting human motion [10]–[13]. Specifically, to match trajectories between the sensors, the authors observe a similarity measure of the net velocity history profiles; however, in the optimization step, they rely only on the detected positions of the tracked people. In [14], authors propose to calibrate multiple 2D lidars by tracking moving targets using a pose graph, wherein rotation is decoupled from translation by using a rotation averaging approach.

Temporal calibration of a sensor system requires motion, either of the observed target [7], [15] or the system itself [16]–[22]. Furthermore, some research advocates a unified approach to spatiotemporal calibration [17], while others claim that estimating uncorrelated quantities, such as time delay and homogeneous transforms, might degrade the final result [23]. Additional challenge in temporal calibration is computational complexity; namely, at each optimization step new correspondences need to be computed due to the new time delay perturbation. Therefore, the common approach is to reduce the dimensionality of the problem and preferably remove correlation with the extrinsic calibration. In [7] authors tracked a colored sphere to perform spatiotemporal calibration of multiple Kinect v2 sensors. By performing principal component analysis on the trajectories, they obtained field of view invariant one-dimensional kernels used in temporal calibration. Even though this method is applicable to other sensors, it assumes the same frame rate of the sensors and its resolution is limited to the sampling time. In [15], temporal calibration based on target tracking is presented where the authors use linear interpolation for continuous-time representation and position norm for the dimensionality reduction. The $AX = XB$ sensor calibration problem with unknown temporal correspondences was tackled in [19]. To perform dimensionality reduction, authors used one-dimensional invariants – displacement and

Authors are with the University of Zagreb Faculty of Electrical Engineering and Computing, Laboratory for Autonomous Systems and Mobile Robotics, Croatia. {juraj.persic, luka.petrovic, ivan.markovic, ivan.petrovic}@fer.hr

angle of rotation – defined by Plücker coordinates of the screw motion. In [20], authors proposed an algorithm based on system motion by aligning curves in the 3D orientation space. The temporal calibration problem was formulated as a registration task which can be considered as a variant of the iterative closest point algorithm. Temporal camera–lidar calibration using a hardware system based on LED display and photo diode is performed in [24]. The approach in [17], [25] is similar to ours as it uses B-splines for continuous-time representation. Camera-IMU spatiotemporal calibration relies on estimator that (i) represents system's motion as a single continuous-time trajectory, (ii) incorporates raw IMU measurements and (iii) minimizes projection error of detected checkerboard corners with a camera. In addition, method from [17] can include lidar in the calibration if environment has enough planar surfaces.

In this paper, we focus on the target tracking-based spatiotemporal calibration relying on continuous-time representation using Gaussian processes (GPs). Leveraging GPs enables a theoretically grounded batch state estimation and interpolation, while it has been a well recognized tool in machine learning [26] both for regression and classification problems, and have been proposed for a variety of robotics challenges as well [27]. For example, in [28], [29] mobile robot localization was a motivation for an efficient batch state estimation using GP regression, in [30] GPs have been used for efficient motion planning, being especially valuable in high-dimensional configuration spaces, while in [31] they were used for tracking of extended objects. Common alternative to GP regression are B-splines, often used for their computational efficiency. However, recent development of the GP regression [29] enabled comparable efficiency, while GPs provide several advantages. They are configured using a standard state estimation framework, i.e. by choosing a physical motion model and tuning process and measurement noise. On the other hand, B-splines require tuning the polynomial degree and the spacing between the knots, which can be a non-trivial task [32]. Furthermore, unlike B-splines, GPs estimate trajectory covariance.

The advantages of the proposed calibration method are (i) *joint spatiotemporal calibration* based on efficient on-manifold optimization, (ii) *theoretically grounded batch state estimation and interpolation*, based on the theory of GPs, which enables both the *time delay and clock drift estimation*, (iii) graph-based extension enabling *multisensor calibration* (iv) *computational efficiency*, thanks to the exactly sparse GP priors resulting with $\mathcal{O}(N)$ complexity with respect to the number of measurements. Furthermore, the GP interpolation provides an *exact temporal registration* between the sensors which is necessary for the extrinsic calibration. We evaluate the proposed method in extensive simulation and real-world experiments with five different multisensor setups and compare the method to state-of-the-art. Note that the proposed method *requires only* that sensors can track position of the same moving target. Thus, we can use variety of different targets, while specific knowledge about the target can be used in the preprocessing step, e.g. target size for monocular camera scale recovery. Furthermore, the paper is complemented by an open-source ROS toolbox *Calirad* implementing the proposed method and a C++ library *ESGPR* implementing the GP regression.

The rest of the paper is organized as follows. Section II formulates the problem, provides theoretical insights on the used exactly sparse GP regression and elaborates the proposed multisensor spatiotemporal calibration method. Section III shows the results of the method on the simulated data where ground truth calibration is available and compares it to a state-of-the-art ego-motion based method. Experimental results with four different multisensor setups, combined with discussion on implementation details, are given in Section IV. At the end, Section V concludes the paper.

II. Proposed Calibration Method

In this section, we formulate the spatiotemporal calibration problem, present necessary theoretical insights and describe individual steps of the proposed method. The novel calibration method can be separated in two consecutive steps: (i) representing the trajectories of moving targets captured by each sensor with a separate GP and (ii) joint spatiotemporal calibration based on GP interpolation and efficient on-manifold optimization. Furthermore, method can be seamlessly extended to graph representation enabling multisensor calibration. Given that, in Sec. II-A, we first formulate our problem and then present the necessary background on GPs in Sec. II-B The following Sec. II-C describes the proposed on-manifold pairwise calibration, while Sec. II-D introduces adjustments for seamless multisensor calibration.

A. Problem formulation

The goal of our method is to enable extrinsic and temporal calibration of heterogeneous exteroceptive sensors, e.g., cameras, lidars, radars, sonars etc. The method relies on tracking the calibration target whose 3D position can be determined by all sensors. To formalize the approach, we start with defining a target reference frame $\mathcal{F}_t$, described by the target's position ${}^s\boldsymbol{p}(k)$ and orientation ${}^s\boldsymbol{R}(k)$ at discrete time instants. When target reference frames between sensors do not align (e.g. different sensor modalities measure different points on the target), target orientation from one of the sensors and known target configuration are used to express the positions in a unified target reference frame. After this step, we continue to use only target positions because some sensors cannot estimate the target orientation, e.g. the radar. In addition, it also allows us to use a linear motion model yielding faster GP regression; thus for each sensor, the GP regression takes in ${}^s\boldsymbol{p}(k)$ and outputs continuous-time target trajectories ${}^s\boldsymbol{x}(t)$. The method itself is not limited to any target design as it abstracts the sensor readings with the estimated trajectories using GPs.

One of the advantages of our method is that it does not require motion of the sensor system. By relying on target motion, we can perform highly dynamic motions and obtain informative data for precise temporal calibration regardless of the system. While hand-held device can rely on motion-based methods for temporal calibration, sensor systems such as vehicles can greatly benefit from this approach. However,

we point out that our method is not limited to static sensor systems, i.e., we can either move the sensor platform or the target itself. Lastly, to achieve accurate temporal calibration it is crucial to avoid any source of clock jitter. Clock jitter can be avoided by using local sensors' clocks even though clock drift might be present. If the clock drift is ignored, time delay becomes non-stationary and system performance degrades over time. Thus, our temporal calibration approach is extended to estimate clock drift together with the time delay. By relying solely on the sensor measurements, our method is not affected by the clock jitter.

B. GP Trajectory Representation

The proposed method is based on the GP regression approach to target trajectory estimation, leveraging the work in [27]–[29]. It enables an efficient continuous-time trajectory estimation based on discrete-time position measurements, i.e., we are able to query the state at any time of interest. Thus, continuous-time GP representation enables elegant temporal correspondence registration between asynchronous sensors with different frame rates. In this section, we give a brief overview of the GP regression necessary for our method, while we refer the reader to [27] for more details.

We consider systems with a continuous-time GP model prior

$$\boldsymbol{x}(t) \sim \mathcal{GP}(\check{\boldsymbol{x}}(t), \check{\boldsymbol{P}}(t,t')), \tag{1}$$

and a discrete-time, linear measurement model:

$$\boldsymbol{y}_k(t) = \boldsymbol{C}_k \boldsymbol{x}_k(t_k) + \boldsymbol{n}_k, \tag{2}$$

where $\boldsymbol{x}(t)$ is the state, $\check{\boldsymbol{x}}(t)$ is the mean function, $\check{\boldsymbol{P}}(t,t')$ is the covariance function, $\boldsymbol{y}_k$ are the measurements, $\boldsymbol{n}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_k)$ is Gaussian measurement noise, and $\boldsymbol{C}_k$ is the measurement model matrix. For now, we assume that the state is queried at the measurement times, and we will describe querying at other times in (13) and (14). Following the approach presented in [27], the Gaussian posterior evaluates to

$$p(\boldsymbol{x}|\boldsymbol{y}) = \mathcal{N}\Big(\underbrace{(\check{\boldsymbol{P}}^{-1} + \boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{C})^{-1}(\check{\boldsymbol{P}}^{-1}\check{\boldsymbol{x}} + \boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{y})}_{\hat{\boldsymbol{x}},\ \text{posterior mean}}, \underbrace{(\check{\boldsymbol{P}}^{-1} + \boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{C})^{-1}}_{\hat{\boldsymbol{P}},\ \text{posterior covariance}} \Big). \tag{3}$$

After rearranging the posterior mean expression, a linear system for stacked vector of posterior states $\hat{\boldsymbol{x}}$ is obtained

$$(\check{\boldsymbol{P}}^{-1} + \boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{C})\hat{\boldsymbol{x}} = (\check{\boldsymbol{P}}^{-1}\check{\boldsymbol{x}} + \boldsymbol{C}^T\boldsymbol{R}^{-1}\boldsymbol{y}), \tag{4}$$

where $\check{\boldsymbol{P}}$, $\boldsymbol{C}$, and $\boldsymbol{R}$ are batch matrices defined as $\check{\boldsymbol{P}} = [\check{\boldsymbol{P}}(t_i,t_j)]_{ij}$, $\boldsymbol{C} = \text{diag}(\boldsymbol{C}_0, \ldots, \boldsymbol{C}_N)$, and $\boldsymbol{R} = \text{diag}(\boldsymbol{R}_0, \ldots, \boldsymbol{R}_N)$, while $\check{\boldsymbol{x}}$ and $\boldsymbol{y}$ are stacked vectors of prior states at measurement times and actual sensor measurements, $\check{\boldsymbol{x}} = [\check{\boldsymbol{x}}_0, \ldots, \check{\boldsymbol{x}}_N]^T$ and $\boldsymbol{y} = [\boldsymbol{y}_0, \ldots, \boldsymbol{y}_N]^T$, with N being the number of measurements. In general, time complexity for solving (4), as currently presented, is $\mathcal{O}(N^3)$ [29]. To improve the computational efficiency, a special class of GP priors is introduced, whose sparsely structured matrices can be exploited.

The special class of GP priors is based on the following linear time-varying stochastic differential equation (LTV-SDE)

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{F}(t)\boldsymbol{x}(t) + \boldsymbol{v}(t) + \boldsymbol{L}(t)\boldsymbol{w}(t), \tag{5}$$

where $\boldsymbol{F}$ and $\boldsymbol{L}$ are system matrices, $\boldsymbol{v}$ is a known control input, and $\boldsymbol{w}(t)$ is generated by a white noise process. The white noise process is itself a GP with zero mean value

$$\boldsymbol{w}(t) \sim \mathcal{GP}(\boldsymbol{0}, \boldsymbol{Q}_c\delta(t-t')), \tag{6}$$

where $\boldsymbol{Q}_c$ is a power spectral density matrix.

The mean and the covariance of the GP are generated from the solution of the LTV-SDE given in (5)

$$\check{\boldsymbol{x}}(t) = \boldsymbol{\Phi}(t,t_0)\check{\boldsymbol{x}}_0 + \int_{t_0}^{t} \boldsymbol{\Phi}(t,s)\boldsymbol{v}(s)\,\mathrm{d}s, \tag{7}$$

$$\check{\boldsymbol{P}}(t,t') = \boldsymbol{\Phi}(t,t_0)\check{\boldsymbol{P}}_0\boldsymbol{\Phi}(t',t_0)^T + \int_{t_0}^{\min(t,t')} \boldsymbol{\Phi}(t,s)\boldsymbol{L}(s)\boldsymbol{Q}_c\boldsymbol{L}(s)^T\boldsymbol{\Phi}(t',s)^T\,\mathrm{d}s, \tag{8}$$

where $\check{\boldsymbol{x}}_0$ and $\check{\boldsymbol{P}}_0$ are the initial mean and covariance of the first state, and $\boldsymbol{\Phi}(t,s)$ is the state transition matrix [28].

Due to the Markov property of the LTV-SDE in (5), the inverse kernel matrix $\check{\boldsymbol{P}}^{-1}$ of the prior, which is required for solving the linear system in (4), is exactly sparse block tridiagonal [28]:

$$\check{\boldsymbol{P}}^{-1} = \boldsymbol{F}^{-T}\boldsymbol{Q}^{-1}\boldsymbol{F}^{-1}, \tag{9}$$

where

$$\boldsymbol{F}^{-1} = \begin{bmatrix} \boldsymbol{1} & 0 & \ldots & 0 & 0 \\ -\boldsymbol{\Phi}(t_1,t_0) & \boldsymbol{1} & \ldots & 0 & 0 \\ 0 & -\boldsymbol{\Phi}(t_2,t_1) & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \boldsymbol{1} & 0 \\ 0 & 0 & \ldots & -\boldsymbol{\Phi}(t_N,t_{N-1}) & \boldsymbol{1} \end{bmatrix} \tag{10}$$

and

$$\boldsymbol{Q}^{-1} = \text{diag}(\check{\boldsymbol{P}}_0^{-1}, \boldsymbol{Q}_{0,1}^{-1}, \ldots, \boldsymbol{Q}_{N-1,N}^{-1}) \tag{11}$$

with

$$\boldsymbol{Q}_{a,b} = \int_{t_a}^{t_b} \boldsymbol{\Phi}(t_b,s)\boldsymbol{L}(s)\boldsymbol{Q}_c\boldsymbol{L}(s)^T\boldsymbol{\Phi}(t_b,s)^T\,\mathrm{d}s. \tag{12}$$

This kernel allows for computationally efficient, structure-exploiting inference with $\mathcal{O}(N)$ complexity. This is the main advantage of the proposed exactly sparse GP priors based on a LTV-SDE in (5).

As we previously stated, the key benefit of using GPs for the continuous-time target trajectory estimation is the possibility to query the state $\hat{\boldsymbol{x}}(\tau)$ at any time of interest τ, and not only at measurement times. For multisensor calibration, this proves to be extremely useful, since many sensors operate at different frequencies; thus, the GP approach enables us to temporally align the measurements. If the prior proposed in (7) is used, GP interpolation can be performed efficiently due to the aforementioned Markovian property of the LTV-SDE

in (5). State $\hat{\boldsymbol{x}}(\tau)$ at $\tau \in [t_i, t_{i+1}]$ is a function of only its neighboring states [29],

$$\hat{\boldsymbol{x}}(\tau) = \check{\boldsymbol{x}}(\tau) + \boldsymbol{\Lambda}(\tau)(\hat{\boldsymbol{x}}_i - \check{\boldsymbol{x}}_i) + \boldsymbol{\Psi}(\tau)(\hat{\boldsymbol{x}}_{i+1} - \check{\boldsymbol{x}}_{i+1}), \quad (13)$$

$$\boldsymbol{\Lambda}(\tau) = \boldsymbol{\Phi}(\tau, t_i) - \boldsymbol{\Psi}(\tau)\boldsymbol{\Phi}(t_{i+1}, t_i), \quad (14)$$

$$\boldsymbol{\Psi}(\tau) = \boldsymbol{Q}_{i,\tau}\boldsymbol{\Phi}(t_{i+1}, \tau)^T \boldsymbol{Q}_{i,i+1}^{-1}, \quad (15)$$

where $\boldsymbol{Q}_{a,b}$ is given in (12). The fact that any state $\check{\boldsymbol{x}}(\tau)$ can be computed in $O(1)$ complexity can be exploited for efficient matching of trajectories of a target detected by multiple sensors.

For the calibration purposes, measurements from individual sensors are used to create separate GPs, where $s \in S$ represents a particular sensor. As we will see in the Sec. II-C, temporal calibration requires velocity estimates in the analytical Jacobians. While the simplest applicable motion model is the constant velocity (CV) model, we opt for the constant acceleration (CA) model. From our experience, CV model cannot capture the necessary maneuvering dynamics of the target and provides slightly lower precision. However, it can be applied if further decrease in computation time is needed. The model for the sensor s trajectory ${}^s\boldsymbol{x}(t) \in \mathbb{R}^{9\times 1}$ consists of position ${}^s\boldsymbol{p}(t) \in \mathbb{R}^{3\times 1}$, velocity ${}^s\boldsymbol{v}(t) \in \mathbb{R}^{3\times 1}$ and acceleration ${}^s\boldsymbol{a}(t) \in \mathbb{R}^{3\times 1}$:

$${}^s\boldsymbol{x}(t) = \begin{bmatrix} {}^s\boldsymbol{p}(t) \\ {}^s\boldsymbol{v}(t) \\ {}^s\boldsymbol{a}(t) \end{bmatrix} \sim \mathcal{GP}({}^s\check{\boldsymbol{x}}(t), {}^s\check{\boldsymbol{P}}(t, t')). \quad (16)$$

To employ the CA motion prior, the LTV-SDE matrices in (5) have the following form

$$\boldsymbol{F}(t) = \begin{bmatrix} \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}, \boldsymbol{L}(t) = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix}, \boldsymbol{C}(t) = \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}^T, \quad (17)$$

while the matrices $\boldsymbol{\Phi}(t, s)$ and $\boldsymbol{Q}_{a,b}$ are defined as

$$\boldsymbol{\Phi}(t, s) = \begin{bmatrix} \mathbf{1} & (t-s)\mathbf{1} & \frac{(t-s)^2}{2}\mathbf{1} \\ \mathbf{0} & \mathbf{1} & (t-s)\mathbf{1} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}, \quad (18)$$

$$\boldsymbol{Q}_{a,b} = \begin{bmatrix} \frac{\Delta t^5}{20}\boldsymbol{Q}_c & \frac{\Delta t^4}{8}\boldsymbol{Q}_c & \frac{\Delta t^3}{6}\boldsymbol{Q}_c \\ \frac{\Delta t^4}{8}\boldsymbol{Q}_c & \frac{\Delta t^3}{3}\boldsymbol{Q}_c & \frac{\Delta t^2}{2}\boldsymbol{Q}_c \\ \frac{\Delta t^3}{6}\boldsymbol{Q}_c & \frac{\Delta t^2}{2}\boldsymbol{Q}_c & \Delta t\boldsymbol{Q}_c \end{bmatrix}, \quad (19)$$

with $\Delta t = t_b - t_a$. We would also like to emphasize that using motion prior with proper covariances can help mitigate the effects of occasional outliers which can occur in the context of the sensor calibration.

C. Joint On-Manifold Optimization

Let's consider a sensor setup consisting of two sensors. Once a GP target trajectory for each of them is estimated, we proceed to joint spatiotemporal calibration. Our goal is to find temporal and extrinsic parameters between the sensors which best align the target trajectories in terms of their positions. This task can be treated as an iterative closest point (ICP) problem with known point correspondence, but unknown temporal correspondence. We propose an iterative least-square solver that leverages previous work on efficient on-manifold optimization for ICP presented by Grisetti et al. [33]. By relying on continuous-time trajectory estimates using the GPs, we are able to extend the solver to estimate temporal calibration between the sensors as well.

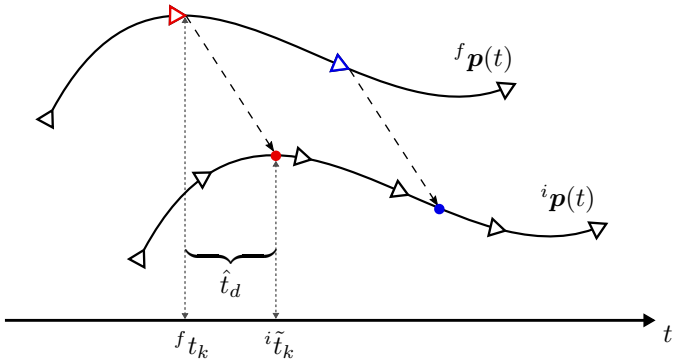


Fig. 1. Continuous-time trajectory representation using Gaussian processes provides an elegant temporal registration of asynchronous measurements. Illustration shows the time delay estimation by aligning two target trajectories, ${}^f\boldsymbol{p}(t)$ and ${}^i\boldsymbol{p}(t)$. States of the fixed sensor at measurement times (triangles) and states at interpolated times (circle) are used to generate correspondences (blue and red pairs).

We start by defining one sensor as fixed (label f) whose states are evaluated at its respective measurement time instances ${}^f t_k$, $k \in (1, N)$. The other sensor we define as the interpolated one (label i), because we interpolate its states at each optimization step using (13)-(15) at corresponding time instances based on the current temporal parameters. Figure 1 illustrates temporal correspondence registration and target position trajectories observed by a fixed and an interpolated sensor, labeled ${}^f\boldsymbol{p}(t)$ and ${}^i\boldsymbol{p}(t)$, respectively. It is worth noting that in the case of different sensor frame rates, the slower sensor should be chosen as the fixed one to reduce interpolation errors [34] and computational costs.

To derive our method, we start by defining the optimization problem as a search for extrinsic and temporal calibration parameters defined on the manifold that is a direct product of the *SE(3)* and $\mathbb{R}^2$ Lie groups, representing extrinsic and temporal calibration parameters, respectively, i.e., $\boldsymbol{X} \in$ *SE(3)* $\times\, \mathbb{R}^2 = \mathcal{M}$. Given that, we write our state $\boldsymbol{X}$ as the following composite matrix (all other elements are zero)

$$\boldsymbol{X} = \left\{ \begin{bmatrix} {}^f_i\boldsymbol{R} & {}^f_i\boldsymbol{t} \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & t_d \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & k_d \\ 0 & 1 \end{bmatrix} \right\} \in \mathbb{R}^{8\times 8}, \quad (20)$$

where ${}^f_i\boldsymbol{R}$, ${}^f_i\boldsymbol{t}$, t_d and k_d are the rotation matrix, the translation vector, the time delay and the clock drift coefficient, respectively. At each step of the iterative optimization, using the current estimate of t_d and k_d, we obtain corresponding timestamps using:

$${}^f t_k = (1 + k_d){}^i t_k + t_d. \quad (21)$$

We note that when clock drift estimation is unnecessary, e.g. sensors use a central clock, we simply drop out the terms related to k_d. To follow the standard maximum likelihood estimation (MLE) framework [33], we treat the fixed sensor

position estimates obtained by the GP as measurements corrupted by the Gaussian noise

$$\boldsymbol{z}_k = {}^f\boldsymbol{p}\left({}^f t_k\right) + \nu_k, \quad \nu_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Omega}_k^{-1}), \tag{22}$$

where $\boldsymbol{\Omega}_k \in \mathbb{R}^{3\times3}$ defines the inverse of the position covariance matrix.

On the other hand, positions of the interpolated sensor ${}^i\boldsymbol{p}(t)$ are non-stationary since they are interpolated at each iteration of the optimization process. Thus, we treat them as a part of the observation model $\boldsymbol{h}_k(\boldsymbol{X}) : \mathcal{M} \to \mathbb{R}^3$:

$$\boldsymbol{h}_k(\boldsymbol{X}) = {}^f_i\boldsymbol{R} \cdot {}^i\boldsymbol{p}\left(\frac{{}^f t_k - t_d}{1 + k_d}\right) + {}^f_i\boldsymbol{t}. \tag{23}$$

To find the optimal solution $\boldsymbol{X}^*$ given all the measurements, the MLE approach suggests to minimize the following expression:

$$\boldsymbol{X}^* = \arg\min_{\boldsymbol{X}} F(\boldsymbol{X}), \tag{24}$$

$$F(\boldsymbol{X}) = \sum_{k=1}^{N} \boldsymbol{e}_k^T(\boldsymbol{X}) \boldsymbol{\Omega}_k \boldsymbol{e}_k(\boldsymbol{X}), \tag{25}$$

$$\boldsymbol{e}_k(\boldsymbol{X}) = \boldsymbol{h}_k(\boldsymbol{X}) - \boldsymbol{z}_k. \tag{26}$$

To solve this optimization problem we follow the on-manifold Gauss-Newton (GN) optimization framework [35]. Our solver builds upon the formulation of the ICP problem [33] with additional estimation of temporal calibration parameters. Computationally the most demanding part is the state interpolation that occurs at each iteration. Therefore, our goal is to minimize the number of cost function evaluations by obtaining the parameter perturbations on the manifold and by using analytical Jacobians that will be derived in the sequel.

As previously stated, the manifold $\mathcal{M}$ on which we perform the optimization is a direct product of the *SE(3)* and $\mathbb{R}^2$ Lie groups, thus the perturbation vector $\Delta\boldsymbol{x} \in \mathbb{R}^8$ is the corresponding Lie algebra element:

$$\Delta\boldsymbol{x} = \mathrm{Log}(\boldsymbol{X}) = [\Delta\boldsymbol{r}\ \Delta\boldsymbol{t}\ \Delta t_d\ \Delta k_d]. \tag{27}$$

In order to avoid cluttering the section with additional mathematical notation, we do not introduce here explicitly Lie group operators. We would just like to point out that our perturbation vector is actually the Euclidean vector of the space isomorphic to the Lie algebra of *SE(3)* $\times\ \mathbb{R}^2$, while the corresponding matrix exponential and logarithm that map the vector space elements to the group, and vice-versa, are denoted as Exp and Log. We believe that this lack of mathematical accuracy does not impact the correctness, but brings clarity in presenting the current paper method. For more details on Lie groups, Lie algebra, and pertaining operators we refer the reader to [**?**], [36].

To perform the on-manifold GN optimization note that our perturbed observation model is as follows (we use perturbation on the left in this paper)

$$\boldsymbol{h}_k(\mathrm{Exp}(\Delta\boldsymbol{x}) \cdot \boldsymbol{X}) = \Delta\boldsymbol{R}\ {}^f_i\boldsymbol{R} \cdot {}^i\boldsymbol{p}\left(\tilde{t}_k\right) + \Delta\boldsymbol{R}\ {}^f_i\boldsymbol{t} + \Delta\boldsymbol{t}, \tag{28}$$

$$\tilde{t}_k = \frac{{}^f t_k - (t_d + \Delta t_d)}{1 + k_d + \Delta k_d}. \tag{29}$$

We start with an initial guess of the state $\boldsymbol{X}_0 \in \mathcal{M}$ and use it as the current estimate $\hat{\boldsymbol{X}} \in \mathcal{M}$ to evaluate the errors (26). Next, we find the optimal state perturbation $\Delta\boldsymbol{x}$ by linearizing the error term (26) at $\mathrm{Exp}(\Delta\boldsymbol{x}) \cdot \hat{\boldsymbol{X}}$ using the first-order Taylor approximation

$$\boldsymbol{e}_k(\mathrm{Exp}(\Delta\boldsymbol{x}) \cdot \hat{\boldsymbol{X}}) \approx \mathbf{e}_k(\hat{\boldsymbol{X}}) + \underbrace{\left.\frac{\partial \boldsymbol{e}_k(\mathrm{Exp}(\Delta\boldsymbol{x}) \cdot \hat{\boldsymbol{X}})}{\partial \Delta\boldsymbol{x}}\right|_{\Delta\boldsymbol{x}=0}}_{\boldsymbol{J_k}} \Delta\boldsymbol{x}. \tag{30}$$

After substituting the linearized error (30) into (25) to obtain a linearized criterion, we get the following quadratic form

$$\begin{aligned} F(\mathrm{Exp}(\Delta\boldsymbol{x}) \cdot \hat{\boldsymbol{X}}) \approx\ & \Delta\boldsymbol{x}^T \boldsymbol{H} \Delta\boldsymbol{x} + 2\boldsymbol{b}^T \Delta\boldsymbol{x} \\ & + \sum_{k=1}^{N} \boldsymbol{e}_k^T(\hat{\boldsymbol{X}}) \boldsymbol{\Omega}_k \boldsymbol{e}_k(\hat{\boldsymbol{X}}), \end{aligned} \tag{31}$$

where

$$\boldsymbol{H} = \sum_{k=1}^{N} \boldsymbol{J}_k^\top \boldsymbol{\Omega}_k \boldsymbol{J}_k, \quad \boldsymbol{b} = \sum_{k=1}^{N} \boldsymbol{J}_k^\top \boldsymbol{\Omega}_k \boldsymbol{e}_k. \tag{32}$$

The optimal perturbation vector at each iteration is found by equating the derivative of (31) with zero

$$\Delta\boldsymbol{x} = -\boldsymbol{H}^{-1}\boldsymbol{b}. \tag{33}$$

We then update the current state estimate using $\hat{\boldsymbol{X}} \leftarrow \mathrm{Exp}(\Delta\boldsymbol{x}) \cdot \hat{\boldsymbol{X}}$ and the process is repeated until convergence.

As the final ingredient, we derive the analytical (left) Jacobians as they are essential for reducing the computational complexity. We start by separating the complete k-th Jacobian to subparts for convenience

$$\boldsymbol{J}_k = [\boldsymbol{J}_k^{\Delta\boldsymbol{r}}\ \boldsymbol{J}_k^{\Delta\boldsymbol{t}}\ \boldsymbol{J}_k^{\Delta t_d}\ \boldsymbol{J}_k^{\Delta k_d}]. \tag{34}$$

We approximate the perturbation rotation matrix by $\Delta R = \boldsymbol{I} + [\Delta r]_\times$ [36], where the $[\cdot]_\times$ operator constructs a skew-symmetric matrix from the vector. Leveraging this approximation and neglecting the constant terms which disappear via derivation, we obtain the following Jacobians

$$\boldsymbol{J}_k^{\Delta\boldsymbol{t}} = \left.\frac{\partial(\Delta\boldsymbol{t})}{\partial(\Delta\boldsymbol{t})}\right|_{\Delta\boldsymbol{x}=0} = \boldsymbol{I}, \tag{35}$$

$$\boldsymbol{J}_k^{\Delta\boldsymbol{r}} = \left.\frac{\partial(\Delta\boldsymbol{R} \cdot {}^i\boldsymbol{p}'_k)}{\partial(\Delta\boldsymbol{r})}\right|_{\Delta\boldsymbol{x}=0} = \left[-{}^i\boldsymbol{p}'_k\right]_\times, \tag{36}$$

$${}^i\boldsymbol{p}'_k = {}^f_i\boldsymbol{R} \cdot {}^i\boldsymbol{p}\left(\tilde{t}_k\right) + {}^f_i\boldsymbol{t}. \tag{37}$$

And regarding the temporal calibration parameters, Jacobians evaluate to the following expressions

$$\boldsymbol{J}_k^{\Delta t_d} = \left.\frac{\partial(\Delta\boldsymbol{R} \cdot {}^f_i\boldsymbol{R} \cdot {}^i\boldsymbol{p}\left(\tilde{t}_k\right))}{\partial(\Delta t_d)}\right|_{\Delta\boldsymbol{x}=0} = {}^f_i\boldsymbol{R} \cdot {}^i\boldsymbol{v}\left(\tilde{t}_k\right) \cdot \frac{-1}{1 + k_d}, \tag{38}$$

$$\boldsymbol{J}_k^{\Delta k_d} = \left.\frac{\partial(\Delta\boldsymbol{R} \cdot {}^f_i\boldsymbol{R} \cdot {}^i\boldsymbol{p}\left(\tilde{t}_k\right))}{\partial(\Delta k_d)}\right|_{\Delta\boldsymbol{x}=0} = {}^f_i\boldsymbol{R} \cdot {}^i\boldsymbol{v}\left(\tilde{t}_k\right) \cdot \frac{t_d - {}^f t_k}{(1 + k_d)^2}, \tag{39}$$

where ${}^i\boldsymbol{v}\left(\tilde{t}_k\right)$ is the interpolated sensor's velocity estimate of the target at time instant $\tilde{t}_k$. As shown in the Sec. II-B, it is readily available since the used GPs provide smooth continuous-time velocity estimates.

Finally, it is crucial to keep the number of correspondences constant to ensure convergence; otherwise, the cost function loses its smoothness, because adding or removing a correspondence inevitably introduces a discontinuity and prevents convergence. This situation occurs when the method seeks correspondence between the fixed sensor and the interpolated state of the second sensor which is outside of the trajectory lifetime. Even though the GP framework allows for extrapolation into the future or the past, thus enabling the necessary correspondences, we avoid this approach as it does not convey any additional information and could possibly degrade the calibration results. Instead, we set a lower and upper bound on the time delay. Given that, we align two GP trajectories and discard fixed measurements at the beginning and the end in accordance to the bounds, thus ensuring constant number of correspondences.

D. Multisensor extension

The proposed method can be easily modified and applied in multisensor scenarios (with more than two sensors) by relying on extrinsic graph-based calibration [37] and extending it to perform temporal calibration as well. For the multisensor case, in addition to previously defined fixed and interpolated sensors, we also need to declare one sensor as the global reference sensor, labeled *r*, since fixed and interpolated sensors now relate a pairwise relation within the graph. Then, we search for extrinsic and temporal parameters relating sensors $(2, \ldots, S)$ to the reference sensor. When there are only 2 sensors, the reference and fixed sensor are the same, while here we choose one fixed sensor for each edge of the graph, preferably the one with the lower frame rate. To start, we need to modify the state vector from (20) to

$$\boldsymbol{X} = \{X_1 \times X_2 \times \ldots \times X_S\} \in \mathbb{R}^{8 \cdot S \times 8 \cdot S}. \tag{40}$$

Each node in the graph represents sensor's extrinsic and temporal parameters, while edges represent correspondences between sensors. Due to multiple edges in a general graph, we need to redefine the observation model for the multisensor approach as follows

$$\boldsymbol{z}_k^{f,i} = \nu_k^{f,i}, \tag{41}$$

$$\nu_k^{f,i} = \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Omega}_{f,i,k}^{-1}), \tag{42}$$

$$\boldsymbol{h}_k^{f,i}(\boldsymbol{X}) = \boldsymbol{h}_k^{i}(\boldsymbol{X}) - \boldsymbol{h}_k^{f}(\boldsymbol{X}), \tag{43}$$

$$\boldsymbol{h}_k^{f}(\boldsymbol{X}) = {}_f^r\boldsymbol{R} \cdot {}^f\boldsymbol{p}({}^f t_k^i) + {}_f^r\boldsymbol{t}, \tag{44}$$

$$\boldsymbol{h}_k^{i}(\boldsymbol{X}) = {}_i^r\boldsymbol{R} \cdot {}^i\boldsymbol{p}({}^i\tilde{t}_k^f) + {}_i^r\boldsymbol{t}, \tag{45}$$

$${}^i\tilde{t}_k^f = \frac{(1 + k_{d,f}) \cdot {}^f t_k^i + t_{d,f} - t_{d,i}}{1 + k_{d,i}}. \tag{46}$$

In the multisensor approach, the target positions from all sensors depend on the estimated temporal parameters (except for the reference sensor); thus, target positions from both sensors within a graph edge are part of the observation model $\boldsymbol{h}_k^{f,i}(\boldsymbol{X})$. To avoid interpolation of both sensor trajectories, we have decided to keep time instances ${}^f t_k^i$ fixed, where they represent measurement times of the fixed sensor that have correspondence with the interpolated sensor. As such, $\boldsymbol{h}_k^f(\boldsymbol{X})$ depends only on the extrinsic parameters of the fixed sensor. On the other hand, we combine temporal parameters of both the fixed and the interpolated sensor into $\boldsymbol{h}_k^i(\boldsymbol{X})$ by first transforming ${}^f t_k^i$ into the reference clock and than into the interpolated sensors clock via (46).

Following these extensions, we need to modify the objective function (25) to sum over all edges defined with set $\mathcal{E}$

$$F(\boldsymbol{X}) = \sum_{(f,i) \in \mathcal{E}} \sum_{k=1}^{N_{f,i}} (\boldsymbol{e}_k^{f,i}(\boldsymbol{X}))^T \boldsymbol{\Omega}_k \boldsymbol{e}_k^{f,i}(\boldsymbol{X}), \tag{47}$$

$$\boldsymbol{e}_k^{f,i}(\boldsymbol{X}) = \boldsymbol{h}_k^{f,i}(\boldsymbol{X}) - \boldsymbol{z}_k^{f,i}. \tag{48}$$

While the remaining expressions are trivially adjusted and omitted here for brevity, we state the Jacobians with respect to the temporal parameters, since they are slightly more complex due to the novel formulation (46)

$$\boldsymbol{J}_k^{\Delta t_{d,f}} = {}_i^r\boldsymbol{R} \cdot {}^i\boldsymbol{v}\left({}^i\tilde{t}_k^f\right) \cdot \frac{1}{1 + k_{d,i}}, \tag{49}$$

$$\boldsymbol{J}_k^{\Delta t_{d,i}} = {}_i^r\boldsymbol{R} \cdot {}^i\boldsymbol{v}\left({}^i\tilde{t}_k^f\right) \cdot \frac{-1}{1 + k_{d,i}}, \tag{50}$$

$$\boldsymbol{J}_k^{\Delta k_{d,f}} = {}_i^r\boldsymbol{R} \cdot {}^i\boldsymbol{v}\left({}^i\tilde{t}_k^f\right) \cdot {}^f t_k \cdot \frac{{}^f t_k^i}{1 + k_{d,i}}, \tag{51}$$

$$\boldsymbol{J}_k^{\Delta k_{d,i}} = {}_i^r\boldsymbol{R} \cdot {}^i\boldsymbol{v}\left({}^i\tilde{t}_k^f\right) \cdot {}^f t_k \cdot \frac{(1 + k_{d,f}) \cdot {}^f t_k^i + t_{d,f} - t_{d,i}}{-(1 + k_{d,i})^2}. \tag{52}$$

III. Simulation Results

Sensor calibration is a task for which ground truth is virtually impossible to obtain in real world experiments. Given that, we use synthetic datasets with known ground truth to assess accuracy of our method and compare it to a state-of-the-art motion-based method [38].

A. Method analysis

To analyze the results of our method in a controlled environment, we simulated an experiment which would mimic a real world experiment. We simulated 1000 sinusoidal trajectories that lasted for 60 s (20 s in each direction) with an amplitude of 1 m and sine period of 4 s. All the sensors operate at 20 Hz and we add white noise with standard deviation of $\sigma_p = 0.01\,\mathrm{m}$ to position measurements.

To test the graph based multisensor calibration, we simulated a graph of four sensors with edges $\mathcal{E} = \big((1,2),(1,3),(2,3),(3,4)\big)$. The first sensor is chosen as the reference, while we set various relative transformations and delays between different sensors up to 400 ms, 40 cm and 70° in Euler angles. Table I shows mean absolute errors between the estimated parameters and the ground truth for five sensor pairs of interest. Rotational error $\Delta_i^f\boldsymbol{R}$ is defined as angle in the angle-axis representation of the rotation matrix ${}_i^f\boldsymbol{R}^T {}_i^f\boldsymbol{R}_{gt}$, while translational error $\Delta^f\boldsymbol{t}_i$ is the standard Euclidean norm of the difference ${}^f\boldsymbol{t}_i - {}^f\boldsymbol{t}_{i,gt}$. From Table I

TABLE I
MEAN ABSOLUTE ERROR OVER GRAPH

f – i	$\Delta_{\boldsymbol{i}}^{\boldsymbol{f}}\boldsymbol{R}$ [°]	$\Delta^{\boldsymbol{f}}\boldsymbol{t_i}$ [mm]	$\lvert\Delta t_{d,f,i}\rvert$ [ms]
1 – 2	0.065	1.81	0.30
1 – 3	0.066	1.76	0.30
1 – 4	0.065	1.73	0.29
2 – 3	0.066	1.75	0.30
3 – 4	0.065	1.73	0.29

we can see that error is essentially equal for all the sensor combinations, whether they share a connection or not. Furthermore, we also obtained very similar results by using a pairwise approach which, however, does not preserve the global consistency. To assess the consistency error present with the pairwise approach, we have "closed the loop" by combining the following pairwise transformations: 1–2, 2–3, 3–1. The resulting mean/maximum errors after the loop closing were $0.007°/0.02°$, 0.07 mm/0.23 mm and 0.06 ms/0.27 ms for rotation, translation and time delay, respectively. Note that for the graph based approach these errors are zero.

To gain further insights about the influence of the experimental setup and modeling, we compared CV and CA motion models for the GP, tested different dynamics of the moving target, added clock drift estimation when it did not exist, and varied the measurement noise. We noted that using the simpler CV motion model resulted with an increase in the mean absolute time delay error from 0.30 ms to 0.44 ms, showing that using a more complex CA motion model is justified. When we doubled the sine frequency, it lowered the mean absolute time delay error from 0.30 ms to 0.15 ms, indicating that the precision of our method is mostly limited by the experiment design. Furthermore, when we included the clock drift in the optimization, we noticed an increase of the mean absolute time delay error from 0.30 ms to 0.62 ms. This effect is most likely due to overfitting and suggest that clock drift should not be estimated if it does not exist, e.g., if sensors use a central clock. Finally, we examined the influence of the measurement error by simulating severe noise $\sigma_p = 0.05\,\mathrm{m}$. It increased the mean absolute errors to $\Delta_{\boldsymbol{i}}^{\boldsymbol{f}}\boldsymbol{R} = 0.37°$, $\Delta^{\boldsymbol{f}}\boldsymbol{t_i} = 10.2\,\mathrm{mm}$ and $\lvert\Delta t_{d,f,i}\rvert = 2.1\,\mathrm{ms}$.

B. Comparison with an ego-motion based method

In this section, we compared our method to a state-of-the-art ego-motion based method named *SRRG* by Della Corte et al. [38]. We generated synthetic data using a *Bernoulli-Lemniscate* 3D trajectory simulator provided with the accompanying *SRRG* toolbox. The generated trajectory resembled a figure eight and excited all rotational axes leading to full observabilty for the ego-motion based methods. We simulated 1000 1-minute long datasets with two sensors operating at $20\,Hz$ ($T = 50\,\mathrm{ms}$) and we added white noise with standard deviation of $\sigma_p = 0.01\,\mathrm{m}$ to positions of the sensors and $\sigma_\theta = 0.1°$ to each Euler angle representing the sensor orientations. Ground truth time delay was set to 0 ms, translation to ${}^1\boldsymbol{t}_2 = [0.2\ 0.2\ 0.2]^T\,\mathrm{m}$ and rotation expressed as quaternion to ${}^1_2\boldsymbol{q} = [0.85\ 0.30\ 0.30\ 0.30]$. Besides odometry constraints, the *SRRG* method allows addition of a generic ICP constraints using raw sensor data to improve results of the extrinsic calibration. We tested both approaches and refer to them as *SRRG-ODO* and *SRRG-ICP*. To enable *SRRG-ICP*, the dataset was expanded with 300 points (added white noise with standard deviation of $\sigma_p = 0.01\,\mathrm{m}$) that were observed throughout the whole trajectory. The input to our *GP* method were only positions of sensor reference frame origins. To obtain continuous-time trajectories, the *SRRG-ODO* approach uses linear interpolation for translation and spherical linear interpolation for rotation. On the other hand, there is no continuous time representation for the *SRRG-ICP* constraint, but they select two closest measurements between sensors based on current time delay estimate. Thus, the *SRRG-ICP* constraint mostly helps correct the extrinsic calibration which can be unobservable for odometry-based constraints (e.g. planar motion). We briefly note that all three methods produced unbiased estimates of extrinsic parameters. They produced mean absolute translational and rotational errors $\boldsymbol{e}_{GP} = (0.2\,\mathrm{cm},\ 0.42°)$, $\boldsymbol{e}_{SRRG-ODO} = (2.4\,\mathrm{cm},\ 0.31°)$ and $\boldsymbol{e}_{SRRG-ICP} = (0.8\,\mathrm{cm},\ 0.24°)$. Furthermore, we tested the *SRRG-ICP* method with disabled temporal calibration using the ground truth delay. It reduced the errors to $\boldsymbol{e}_{SRRG-ICP} = (0.1\,\mathrm{cm},\ 0.02°)$ showing the influence of incorrect temporal calibration described in the sequel.

In this scenario, our method produced an accurate unbiased estimate of the time delay with normal distribution $t_d = \mathcal{N}(0.0004, 0.54)$ ms. On the other hand, *SRRG-ODO* and *SRRG-ICP* provided estimates of the time delay spread across the interval $(-54, 2.7)$ ms, with two modes, one at $-T$ and one at 0 ms. Furthermore, lowering the sensor frequency to 10 Hz caused the stronger separation of the modes with most of the estimates being spread ± 8 ms around the $-T$ and 0 ms modes. After thorough testing of the *SRRG* method, we concluded that the lack of smoothness in the cost function might cause incorrect convergence. Namely, the method relies on numerical calculation of the Jacobian with respect to the time delay, where parameter ϵ_{time} is used to differentiate the cost function. While the *SRRG* toolbox suggests setting it to $\epsilon_{time} = T$, we noticed that with $\epsilon_{time} < T/2$, the method completely diverges. Thus, it is not trivial to choose a proper ϵ_{time} when sensors have significantly different frequencies as in our multisensor real-world experiment that will be presented in the sequel.

IV. EXPERIMENTAL RESULTS

To validate the proposed calibration method, we conducted thorough real-world experiments on five different sensor setups:

- Hardware synchronized stereo camera – testing the method on a setup with accurate ground truth
- Camera and motion capture system – testing heterogeneous sensors operating at significantly different frame rates with separate clocks exhibiting substantial drift
- 3D lidar and camera – calibration accounting for the lidar's sweeping data acquisition process

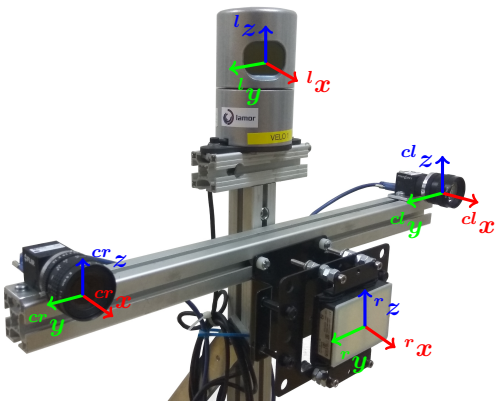


Fig. 2. Illustration of the used multisensor system with corresponding sensor coordinate systems.

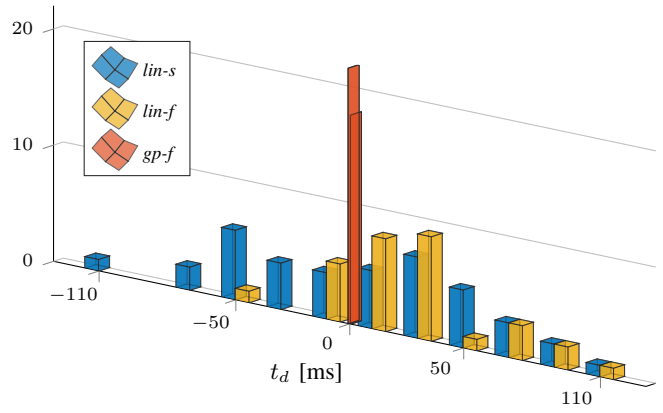


Fig. 3. Histograms compare the estimated time delays using our method (*gp-f*) applied on the whole dataset, to the linear interpolation method [15], where *lin-f* uses the whole dataset, while *lin-s* uses sequences with target moving along the optical axes, thus not introducing a bias in the estimation. Ground truth time delay was $t_d = 0\,\text{ms}$.

- Radar and camera – calibration tackling automotive radar's lack of 3D position measurement.
- Camera, 3D lidar and motion capture system – testing the multisensor graph-based calibration.

In all the experiments we used a single known target for convenience, even though the method does not rely on a special target. We covered a planar triangular cardboard with motion capture markers and an AprilTag [39], a square fiducial marker of side length $a = 16\,\text{cm}$ that removed camera's scale ambiguity, thus enabling 3D target position estimation. Additionally, to get reliable radar detections, we have adopted the target design from [40] and placed a metal corner reflector behind the cardboard. Figure 2 shows the used multisensor system that was mounted on the Husky A200 mobile robot platform. Intrinsic camera calibration was obtained using the Kalibr toolbox [41], while we used factory calibration for the remaining sensors. In the end, we analyze the proposed method's computational complexity and influence of the hyperparameters.

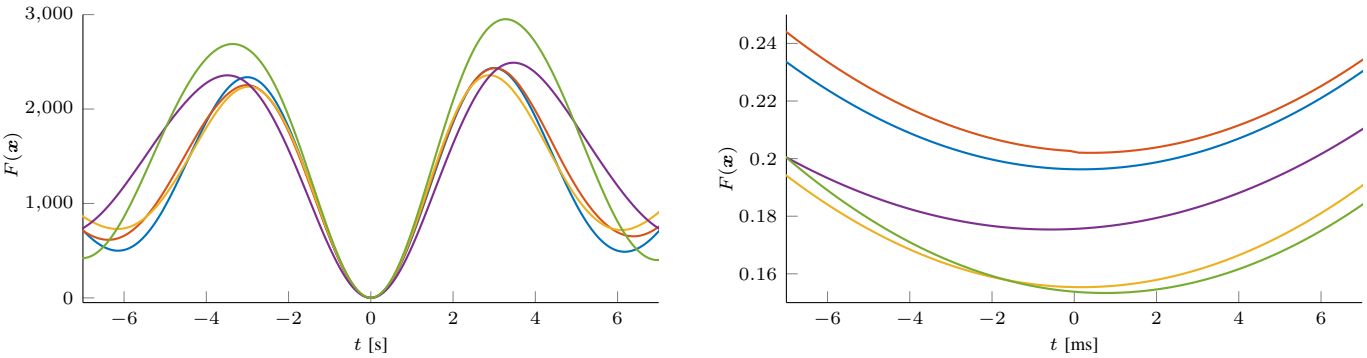
A. Hardware synchronized stereo camera

In this experiment, we used two PointGrey BFLY-U3-23S6M-C global shutter cameras with Kowa C-Mount 6 mm f/1.8-16 1" HC fixed lens with $96.8° \times 79.4°$ field of view. The cameras were synchronized by an external trigger with the sampling rate set to 0.05 s. Note that this setup does not require temporal calibration; however, we leverage this fact to have an experiment with a ground truth time delay ($t_d = 0\,\text{s}$). We have recorded 40 one minute long sequences and compared the performance of the proposed approach to two recent temporal calibration frameworks based on target tracking [7], [15]. The first method [7], correctly estimated the zero time delay for all the 40 recorded sequence, but the approach is limited to estimating the time delay as a multiple of the sampling rate, thus requiring all the sensors to operate at equal sampling rates. Given that, although accurate, due to this limitation the method is not considered further in the paper.

The second method [15], is capable of temporal calibration of asynchronous sensors in the continuous-time domain by relying on linear interpolation of the position norm thus mitigating the limitation of the previous method. In Fig. 3 we compare calibration results of our method and that of linear interpolation. Note that we applied the linear interpolation method first on the whole 40 minute long sequence (*lin-f*), and second on a subset (*lin-s*) for reasons that will be explained in the sequel. Specifically, we can see that linear interpolation exhibits a bias when applied on the full sequence, and larger variance for the subset of the sequence, in comparison to our method (*gp-f*). The explanation lies in the fact that this method relies on the position norm for the dimensionality reduction, which can cause error with the displacements of sensors. Concretely, in the first third of the dataset, the target was moved along the optical axis of both cameras, and the linear interpolation method provided estimated time delay with fitted Gaussian distribution $\hat{t}_d \sim \mathcal{N}(6.03\,, 50.99)\,\text{ms}$[1]. In the remaining parts of the dataset, motion was not aligned with the optical axis and the position norm measurements differed for the two cameras due to large enough displacement, and when applied on the whole dataset the method resulted with the following fitted Gaussian distribution $\hat{t}_d \sim \mathcal{N}(32.20\,, 33.16)\,\text{ms}$. Therefore, we believe that the position norm is not the most appropriate dimensionality reduction technique as it is not frame-invariant. As can be seen from Fig. 3 our method was able to produce an unbiased time delay estimate with the fitted Gaussian distribution $\hat{t}_d \sim \mathcal{N}(0.11,\ 0.39)\,\text{ms}$. Furthermore, all the estimates were within the range $(-0.82, 0.78)\,\text{ms}$, which corresponds to a $\pm 1.6\%$ range of the sampling interval. We can see that the proposed method supports temporal calibration of asynchronous sensors in the continuous-time domain, and that it significantly outperforms the linear interpolation method.

To further gain insight in the proposed temporal calibration method, we examined the cost function defined by (25) (which is smooth compared to linear interpolation). Figure 4a shows the value of the cost function with respect to the time delay using estimated extrinsics at the interval $t_d \in (-7, 7)\,\text{s}$, while Fig. 4b provides a closer look around the global optimum,

[1]In the paper, we represent the Gaussian distribution with the mean and the standard deviation, i.e. $\mathcal{N}(\mu, \sigma)$.



(a) Wide preview illustrating local minima and global minimum.

(b) Closer preview around the ground truth.

Fig. 4. Cost function of the proposed calibration method for 5 different stereo camera experiments. Wide preview shows that initialization within ± 3s interval is sufficient for convergence to the global minimum, while the closer preview around the ground truth time delay confirms cost function smoothness.

$t_d \in (-7, 7)$ ms. For clarity, only 5 out of 40 experiments are shown, while the remaining ones follow the same pattern. From Fig. 4a, we can see that the cost function has local minima, while the global minimum always resides near the ground truth. Since our method uses an iterative solver, proper initialization is necessary. By initializing the time delay to a starting point in the interval $(-3, 3)$ s the method would be able to converge to the global minimum for all the experiments (see Fig 5a). The local minima are tightly coupled with the executed target motion and can be further spread from the global minimum by avoiding repetitive motion or increasing its period. Figure 4b shows that our cost function is smooth with a minimum around the ground truth value, thus enabling stable and accurate results using an iterative optimization.

Furthermore, we use this experiment to evaluate the *SRRG* method on real data due to available target orientation estimates and time delay ground truth. Pose of the camera with respect to the target was used for the *SRRG–ODO* method, while the target position in the camera reference frames was added as a single point input for the *SRRG–ICP* extension. Both methods suffered the same convergence issue described in Sec. III-B with estimated time delay distributions $\hat{t}_d \sim \mathcal{N}(-14.42,\, 24.32)$ ms and $\hat{t}_d \sim \mathcal{N}(-16.20,\, 23.20)$ ms for *SRRG–ODO* and *SRRG–ICP*, respectively. Considering the extrinsic calibration, the *SRRG–ODO* method was not able to estimate translation parameters due to the lack of rotational target movement. This can be seen by comparing estimated means of the translation parameters, e.g. the ${}^1t_{2,x} = 56.9$ cm, ${}^1t_{2,x} = 53.5$ cm and ${}^1t_{2,x} = 0.9$ cm for *GP*, *SRRG-ICP* and *SRRG–ODO*, respectively. Furthermore, we noticed that our method had significantly greater extrinsic parameter repeatability, as shown with Table II. We attribute this result to wrong temporal calibration, confirming the conclusion by Zuñiga-Noël et al. [42] on the *SRRG* method.

To conclude, with this experiment we confirmed that the proposed method provides an unbiased estimate of the time delay which is precise up to a fraction of the sampling interval, and we also showed convergence to a global solution from a wide set of initial values.

TABLE II
STANDARD DEVIATION OF THE ESTIMATED EXTRINSIC CALIBRATION PARAMETERS

	GP	*SRRG–ODO*	*SRRG–ICP*
${}^1t_{2,x}$ [m]	2.79×10^{-3}	6.30×10^{-2}	2.12×10^{-2}
${}^1t_{2,y}$ [m]	4.79×10^{-3}	3.89×10^{-2}	2.22×10^{-2}
${}^1t_{2,z}$ [m]	2.11×10^{-3}	3.66×10^{-2}	5.50×10^{-3}
${}^1_2\theta_z$ [°]	1.52×10^{-1}	4.21×10^{-1}	4.23×10^{-1}
${}^1_2\theta_y$ [°]	5.79×10^{-2}	6.55×10^{-1}	5.26×10^{-1}
${}^1_2\theta_x$ [°]	3.88×10^{-2}	1.02	6.93×10^{-1}

B. Camera and Motion Capture System

In this experiment, we used a single PointGrey camera and the OptiTrack motion capture system (MOCAP). MOCAP provides 6D pose measurements at 120 Hz by processing measurements on a dedicated computer and assigns local timestamps using the computer's clock. Poses are transmitted over the wireless network to the central computer. The camera provides images at 20 Hz and has an internal clock according to which local timestamps are assigned. Images are transmitted over USB to the central computer. Given that, this setup gives us two options for handling data timestamps: (i) to use the time-of-arrival of measurements at the central computer or (ii) to use local timestamps provided by each sensor. The first approach eliminates the timestamp drift caused by separate local clocks, but suffers from the network jitter (since MOCAP data is transferred over the wireless network). The second approach is resilient to the jitter, but separate local clocks introduce a timestamp drift. We analyzed both options in a 34 minute long experiment recording a moving calibration target.

For the time-of-arrival approach, the estimated time delay results, between the MOCAP and camera measurements, followed the Gaussian distribution $\hat{t}_d \sim \mathcal{N}(17.14,\, 1.59)$ ms. Note that the obtained mean value can be interpreted as the average time delay due to network jitter and we noticed that most of the deviations were in the ± 2.9 ms range. However, the time delay can differ significantly during the experiments, because of the changing intensity of the network traffic or other protocol induced stochastic effects. Notably, analysis of the MOCAP time-of-arrival jitter showed that 2.7% of the measurements fell in the range of $(8.3, 418)$ ms, indicating

that on some occasions, delay can be much greater than the MOCAP sampling time. Given that, the jitter caused by the network delay can act as a strong limiting factor for the temporal calibration accuracy.

In the local timestamps approach, we used sensor internal clocks, which eliminates the stochastic effects associated with the communication over a wireless network. However, separate local clocks introduce a drift in the time delay estimation which has to be addressed. To estimate this drift, we compared three approaches: (i) joint drift and delay estimation using the proposed GP method on the full 34 minute sequence (*gp-f*); (ii) drift estimation on the full 34 minute sequence using convex hull approach [43] (*ch-f*), (iii) drift estimation using only one minute subsets (*gp-s*/*ch-s*)[2].

In the *gp-f* approach we performed a joint drift–delay optimization using the proposed GP method on the full sequence. The estimated drift and delay were, $\hat{k}_d = 49.1\,\mu$s/s and $\hat{t}_d = 23\,$ms, respectively. In the *ch-f* approach, the authors observe the temporal evolution of the clock skew, i.e. difference between the arrival times and the local timestamps. They estimate a lower convex hull where the slope of the lower boundary represents the clock drift. With this approach we can obtain each sensor clock drift with respect to the central computer. However, since we are interested in the relative drift, as was estimated in the *gp-f* approach, we report the difference between the two line slopes. Thus the *ch-f* approach resulted with an estimated relative drift of $49.3\,\mu$s/s (at this point, unlike *gp-f*, there is no time delay estimate). To compare the accuracy of estimated drifts, we tested their impact on the time delay estimation. The estimated drifts were used to correct local timestamps, which was followed by the proposed GP time delay estimation on individual one minute intervals (30 in total). The *ch-f* approach resulted with time delay estimates in the range $(22.80, 23.56)\,$ms with estimated distribution $\hat{t}_d \sim \mathcal{N}(23.22,\ 0.17)\,$ms, while the *gp-f* approach resulted with estimates in the range $(22.79, 23.29)\,$ms with estimated distribution $\mathcal{N}(23.02,\ 0.12)\,$ms. The results depicted in the Fig. 5 show the estimated time delays throughout the whole experiment for both the *ch-f* and *gp-f* approach. We can notice that the drift estimate error by the *ch-f* approach introduced a slope of $0.17\,\mu$s/s in the time delay estimate, whereas the *gp-f* approach correctly estimated the drift and provided a consistent time delay estimate throughout the whole 34 minute experiment (the more horizontal line, the better: resulting slope of the time delay estimate was $-0.01\,\mu$s/s).

With the *gp-s* and *ch-s* approaches, the goal was to test if we can obtain accurate clock drift estimation with the proposed method by relying just on one minute long sequences (instead of 30 min long sequences). Furthermore, we also wanted to see what are the requirements on the dataset to produce a reliable drift estimate. Given that, we rearranged the whole 34 min experiment by dividing it into 30 s intervals. Afterwards, the 30 s intervals were paired so that the separation between them was $\Delta t \in (0.5, 5, 15)\,$min. We compared the proposed *GP* framework approach (labeled *gp-s*) to the convex hull approach (labeled *ch-s*). The results of the drift estimation are shown in Fig. 6, illustrating drift estimate uncertainty for different interval separations and methods. The standard deviations of the drift estimates using the *gp-s* approach were $(87.74, 2.93, 0.70)\,\mu$s/s, while *ch-s* approach yielded $(34.65, 3.61, 1.45)\,\mu$s/s, for the $\Delta t \in (0.5, 5, 15)\,$min separation, respectively. It is clear that the case $\Delta t = 0.5\,$min does not provide enough information to estimate the drift, while extending the time separation between the intervals yielded significantly better results, with *gp-s* outperforming *ch-s*.

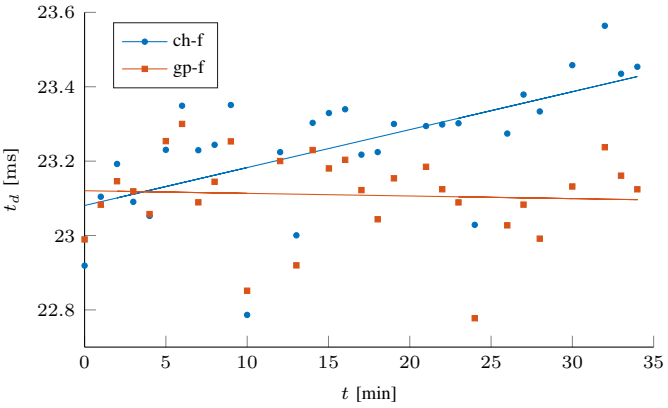


Fig. 5. Estimated time delay for each one minute interval over the whole experiment for the camera and MOCAP temporal calibration. Time delays were obtained from data with compensated drift using the *ch-f* and *gp-f* drift estimates. Steeper slope of the *ch-f* method indicates larger error in the drift estimate used for timestamp compensation.

Finally, to validate the proposed method on MOCAP and camera data sensor fusion, we conducted an experiment in which we observed the reprojection error of the target position. Namely, the target position centroid computed by MOCAP is interpolated to the closest camera frame, using temporal calibration parameters, and then projected in the image using the estimated extrinsic calibration parameters, and compared to the target image centroid. In the experiment[3], the target exhibited static and dynamic periods. From the experiment we can see that during the dynamic periods, the reprojection error rises significantly when using just the time-of-arrivals without any delay compensation, yielding an average reprojection error of $1.9\,$cm. When we compensated for the network jitter caused time delay of $17.1\,$ms, that was obtained by the GP method, the average reprojection error was reduced to $1.0\,$cm. Furthermore, by using the local timestamps approach and GP, i.e., the *gp-f* method, the average reprojection error was further reduced to $0.5\,$cm.

From all the aforementioned results, we can conclude that using time-of-arrival strongly limits the accuracy of the temporal calibration method, while it does provide an estimate of the network delay. On the other hand, the local timestamps approach provides a time delay estimate that is more accurate by an order of magnitude, but requires drift estimation.

[2]Waving a calibration target for a 34 minute stretch requires good stamina, which is why *gp-f* is not a practical approach and serves as the ground truth.

[3]The experiments are shown in the accompanying video available here: https://youtu.be/vqTR6zMIKJs

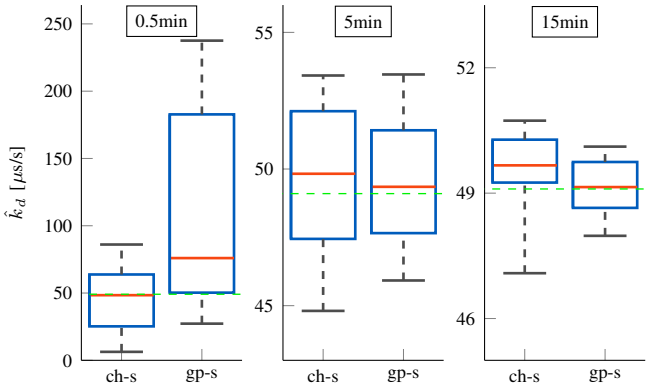


Fig. 6. Uncertainty of the drift estimates for pairs of 30 s intervals separated by 0.5, 5 and 15 minutes. Ground truth (*gp-f*) is illustrated by the horizontal green dashed line. Interval separation of 0.5 min did not produce a reliable drift estimate for both approaches. Longer separation is necessary and proposed *gp-s* outperformed the *ch-s* (notice the difference in the y axis scale).

C. Radar and camera calibration

In this experiment, we used a single PointGrey camera and a Delphi short range radar – a sensor combination commonly applied in automotive applications for tracking of moving targets, since radars are known to be robust to diverse weather conditions and offer long range with wide field of view. However, current radars have a substantial field-of-view in the elevation, but no elevation angle measurements, which makes the extrinsic calibration challenging [40]. Given that, the inability to recover a 3D position of the target violates our main assumption of the proposed calibration. In the sequel, we describe how we adapt our method to tackle this scenario. To the best of the authors knowledge, this is a first attempt of temporal calibration involving an automotive radar sensor.

To address the lack of 3D position measurements in the radar data, we propose a two-step approach. The first step, labeled *gp-3d*, neglects the 2D nature of the radar, and assigns a fictive ${}^{r}p_z = 0$ position measurement to the radar data, i.e., we assume that all the measurements have zero height, thus this step does not require extrinsic calibration a priori. Then, the second step, labeled *gp-2d*, builds upon the results of the *gp-3d* calibration by projecting the 3D camera measurements onto the 2D radar plane. The projected 2D camera measurements are then used to generate a new 2D GP from which refine the calibration. To verify the accuracy of the proposed method, we conducted experiments consisting of 30 one minute intervals with a moving calibration target. During the experiment, we moved the target in the area where camera and radar field of view overlap, while trying to avoid motions unobservable to the radar (an example is given in the accompanying video). Since quality of radar detection degrades in confined spaces, these experiments were conducted in a large open hall.

The estimated time delay using the *gp-3d* step followed the Gaussian distribution $\mathcal{N}(21.81,\, 1.23)\,\mathrm{ms}$, while the *gp-2d* refinement step produces results with the distribution $\mathcal{N}(21.89,\, 1.07)\,\mathrm{ms}$. The results show that the first step, even though neglecting the 3D nature, was able to produce a good time delay estimate without the prior knowledge of extrinsic calibration parameters, while accounting for the 2D nature of the radar produced results with slightly lower standard deviation. Additionally, we note that in these experiments we used arrival times, since using the local timestamps did not provide better results. Probable explanation is that the radar's accuracy of position measurements introduces more uncertainty than does the communication channel jitter. Thus, the estimated time delay shows that the relative latency between the sensors was approximately 22 ms.

Since radar can introduce a higher rate of outliers than other sensors analyzed in the paper, we also studied their effect on the calibration (Fig. 7). Radar outliers mostly occurred when the azimuth measurements were around 0° and are probably caused by limited radar resolution and internal data processing. Figure 7 depicts radar measurements, camera and radar GP posterior means in the y-direction after the calibration. Figure 7a shows the case with low outlier rate, where the GP posterior mean was not affected by corrupted measurements due to relying on the motion prior. On the other hand, Fig. 7b illustrates the effect of high outlier rate, where we can notice strong corruption of the radar posterior mean. Given that, during calibration we analyze the deviations of measurements from the estimated posterior and discard those above a certain threshold; thereafter, the GP regression on the radar data is recomputed.

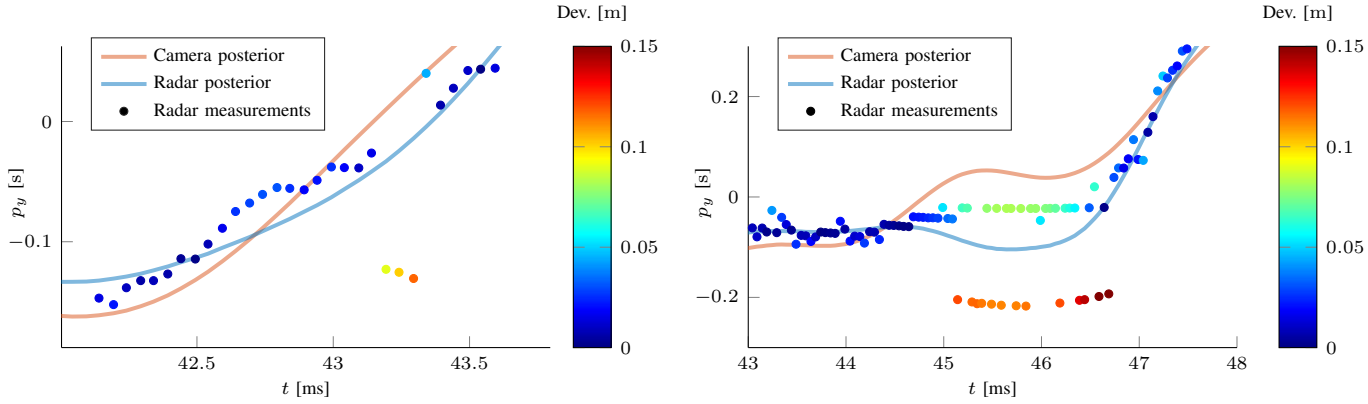
With this experiment, we showed that our method can be easily adapted to a sensor calibration scenario which violates the main assumption: availability of the target's 3D position measurements. Furthermore, we examined the influence of the outliers and showed how we can leverage the GP motion prior to mitigate their influence.

D. 3D lidar and camera calibration

In this experiment, we used a single PointGrey camera and a 3D lidar Velodyne 32E. While the camera, with the global shutter imaging sensor, takes images at discrete time instances, the lidar head sweeps the environment in a continuous manner. Conventionally, despite the continuous nature of the lidar motion, a single sweep of data is most commonly packed in one point cloud, with the timestamp corresponding to the beginning or the end of the sweep. However, to obtain accurate temporal calibration with a moving target, we need to be able to interpolate timestamps between the beginning and end of the sweep. Therefore, continuous-time representations such as GP-s are necessary.

Since our method requires an exact 3D position of a target, we used an isosceles triangle (side lengths $(38, 54, 54)\,\mathrm{cm}$) which enables unambiguous target localization in a sparse point cloud [44]. Furthermore, we have developed a real-time triangle detection and tracking algorithm which is also available as part of the provided toolbox. Briefly, once the algorithm segments planes in the point cloud, it fits the lines to the edges of the planes. Intersections of the lines are then used as vertex hypotheses which are compared to the triangle model vertices. The solution is accepted if the error does not surpass a predefined threshold.

To address lidar's continuous sweep, we compensate target's timestamps by the azimuth angle of the target detection. We form the point cloud by using a fixed cut angle $\theta_{cut} = \pi$, i.e.,



(a) Low outlier rate (b) High outlier rate

Fig. 7. The effect of outliers on GP regression showing raw radar measurements, camera and radar GP mean posteriors (with applied extrinsic calibration) in y direction. Deviation between the radar posterior and measurements is color coded. Low outlier rate is virtually ignored by the posterior due to relying on the GP motion prior and as such could be used directly for temporal calibration. On the other hand, high outlier rate introduces significant discrepancy between camera and radar posterior (radar posterior is "pulled down" by the outliers), thus a measurement validation process needs to be introduced where large deviations from the mean are ignored and the mean is recomputed.

all new data is packed into a point cloud when the driver receives a new measurement at the azimuth angle θ_{cut}, at which point the latest timestamp is assigned to the point cloud. We assume a constant rotational velocity of the lidar with frequency 10 Hz and subtract the point cloud timestamps proportionally to the angular distance between the cut angle and the current target azimuth angle. Finally, we used the local timestamps, which introduced a slight drift; thus, using the *ch-f* approach as in Sec. IV-B, we estimated the relative drift of $1.33\,\mu$s/s and compensated the timestamps accordingly. The results of the time delay estimation were in the range of ± 0.85 ms around the mean value with estimated Gaussian distribution $\mathcal{N}(78.32,\, 0.42)$ ms. The results are comparable to the calibration of synchronized cameras in the Sec. IV-A, despite the challenging factors such as sensor asynchronicity, lidar's continuous sweep and twice lower sampling rate. Thus, we can assert that the method is precise up to the fraction of the fastest sensor.

To evaluate calibration qualitatively, we conducted a data fusion experiment and tested it on a validation dataset not used in calibration. We overlaid camera images with the segmented triangle points from the point clouds. To synchronize the images and the point clouds, we compensated for the estimated time delay and chose the point cloud closest in time to the current image. Finally, to align the point cloud with the image, we translate the point cloud points using the linear interpolation based on the difference vector of the two consecutive triangle positions estimated by the tracker that surround the current image. A preview of the results is shown in the Fig. 8, while the accompanying video also shows this experiment. From Fig. 8 and video we can notice excellent performance of the triangle tracker and accuracy of temporal and extrinsic calibration. Static periods corroborate the accuracy of the extrinsic calibration, as they show consistent overlap of the triangle in the image and segmented lidar points, as illustrated in Fig. 8a. Dynamic periods also exhibit proper alignment, corroborating temporal calibration accuracy, and in Figs. 8b and 8c we illustrate typical worst cases that appear during vertical and sideways motions, respectively. In addition, Fig. 8d shows that the developed tracker works properly even when the triangle is only partially visible by the 3D lidar.

This experiment showed that the proposed method is well-suited for handling sensors with continuous motion affecting data acquisition and combining them with discrete acquisition sensors, like cameras. Furthermore, the data fusion experiment showed a robust performance of the developed target tracker and further confirmed the calibration results.

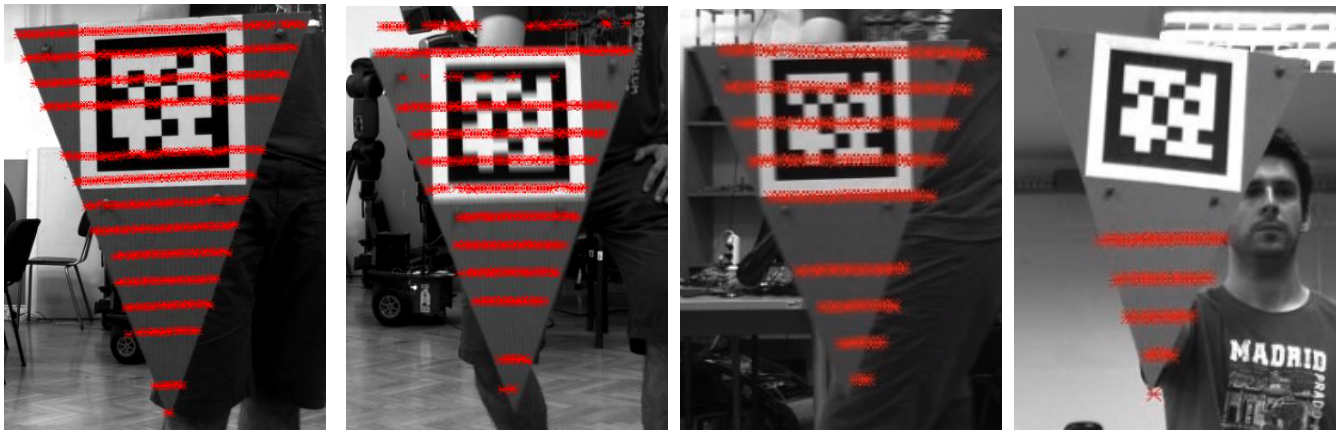
E. Extrinsic calibration

In this section, we provide extrinsic calibration results for the four previously described experiments. Table III shows estimated standard deviations of the individual extrinsic calibration parameters obtained by analyzing the results of one-minute intervals[4]. Uneven uncertainty among different sensor combinations is primarily caused by the involved sensor precision, e.g., MOCAP and camera calibration produces an order of magnitude lower uncertainty due to the high precision of the MOCAP system. Furthermore, some variations in the uncertainty of the extrinsic parameters within each sensor combination is most likely caused by an uneven excitation of different calibration directions in the datasets. Finally, extrinsic calibration of the radar and the camera resulted with a 6D transform whose translation in the z axis, and Euler angles about the y and x axis, had higher variance than the other counterparts. This effect is caused by the lack of the radar's elevation measurements and the interested reader is directed to [40] for a detailed analysis.

F. Multisensor experiment

To test the graph based approach presented in Sec. II-D, we evaluated it on the lidar-camera-MOCAP setup where all three sensors shared the same field of view, i.e., we had a

[4]Indices 1 and 2 denote first and second sensor for a specific experiment.



(a) Static period (b) Vertical motion (c) Sideways motion (d) Partially visible

Fig. 8. Images showing 3D lidar and camera sensor fusion results after proposed spatiotemporal calibration. Static case confirms the validity of extrinsic calibration results, while the dynamic cases illustrate worst case motion introduced error. Detections are present even when the triangle is partially visible.

TABLE III
STANDARD DEVIATION OF THE ESTIMATED EXTRINSIC CALIBRATION PARAMETERS

	Stereo camera	MOCAP–Camera	3D Lidar–Camera	Radar–Camera
${}^{1}t_{2,x}$ [m]	2.79×10^{-3}	4.00×10^{-4}	6.41×10^{-4}	9.26×10^{-3}
${}^{1}t_{2,y}$ [m]	4.79×10^{-3}	2.43×10^{-4}	2.06×10^{-3}	1.94×10^{-2}
${}^{1}t_{2,z}$ [m]	2.11×10^{-3}	6.24×10^{-4}	3.64×10^{-3}	5.44×10^{-2}
${}^{1}_{2}\theta_{z}$ [°]	1.52×10^{-1}	9.75×10^{-3}	6.26×10^{-2}	2.40×10^{-1}
${}^{1}_{2}\theta_{y}$ [°]	5.79×10^{-2}	3.22×10^{-2}	1.65×10^{-1}	7.88×10^{-1}
${}^{1}_{2}\theta_{x}$ [°]	3.88×10^{-2}	1.67×10^{-2}	8.74×10^{-2}	6.84×10^{-1}

fully connected three-node graph. Here we focused on the time delay estimation and thus we preprocessed the timestamps to remove the drift using the *gp-f* approach, while we analyzed results using 30 1-minute intervals. We did not notice any significant differences between the pairwise and graph-based results in terms of delay precision. Namely, standard deviations of time delay estimates for the joint/pairwise sensor combinations were: lidar–camera 0.41/0.42 ms; lidar–MOCAP 0.37/0.38 ms; and camera–MOCAP 0.12/0.12 ms. However, we did notice a significant impact on the consistency of the solution when using the pairwise approach, which is inherently solved using the graph-based approach. Therefore, we used the same consistency test as in Sec. III-A by *closing the loop* in the graph. The experiment yielded: (i) average rotational error of $0.03°$ with the maximum of $0.45°$; (ii) average translational error of 0.5 mm with the maximum of 5.9 mm and (iii) average absolute time delay error of 0.1 ms with the maximum of 1.2 ms. Note again, that for the graph based approach these errors are zero.

G. Implementation details

Computation performance of the proposed method was tested on 40 datasets from the Sec. IV-A problem of the stereo pair spatiotemporal calibration. The calibration starts with two separate GP regressions for each sensor that are completely decoupled and performed in separate threads. On average, one minute intervals consisted of 1138 measurements requiring $t_{GP} = 49$ ms for a complete GP regression. After the GP regression, we performed the GN optimization to obtain extrinsic and temporal calibration parameters. For the stereo pair problem, which had hardware ensured zero time delay, when the optimization was initialized at $t_d = 0.5$ s, ${}^{f}_{i}\boldsymbol{R} = \mathbb{I}^{3\times3}$ and ${}^{f}_{i}\boldsymbol{t} = [0\ 0\ 0]^T$ m, it took around 6 iterations to converge, which translated to the average optimization time of $t_{opt} = 41$ ms. Finally, the total time required for the delay estimation was on average $t_{total} = t_{GP} + t_{opt} = 90$ ms.[5] In general, we handle missing measurements and varying sample times; however, under the assumption of constant sample rates and absence of missing measurements, further improvements on GP regression performance are possible through offline construction of the required batch matrices. It is also important to point out that the algorithm time complexity is $\mathcal{O}(n)$, which makes the method well scalable, especially for sensors with high frame rates or longer experiments.

Considering the effect of the process noise $\boldsymbol{Q}_c$ on the performance of the method, we found that it is fairly resilient. In scenarios with higher outlier rate (e.g. radar experiment), an optimal $\boldsymbol{Q}_c$ can be found which mitigates the influence of the outliers. However, in scenarios with low or zero outlier rate (e.g. simulations or the camera and MOCAP experiment), choosing any reasonable $\boldsymbol{Q}_c$ that does not suppress the measurements in favor of the motion model leads to the same results.

V. CONCLUSION

In this paper we have proposed a spatiotemporal multisensor calibration method based on Gaussian processes moving target tracking. The proposed method relies on the target positions in joint spatiotemporal calibration, while it can also estimate clock drift and the time delay. Method efficiency is achieved by relying on exactly sparse GP regression for target trajectory representation and on-manifold optimization framework. Furthermore, the method is applicable to any multisensor setup with arbitrary number of sensors, as long as sensors can estimate the 3D position of a moving target.

We have validated the proposed calibration method in extensive simulation and real-world experiments on four multisensor setups. The first setup consisted of two externally

[5]Machine used for testing had i7-6700HQ CPU at 2.6 GHz × 8 and 16 GB of 2133 MHz DDR4 RAM

triggered cameras, demonstrating the validity of our method on vision sensors with a readily available ground truth. The second setup consisted of a single camera and a motion capture system, demonstrating the proposed method on a heterogeneous sensor setup with significant difference in frame rates and communication over a wireless network. The third setup analyzed a common automotive heterogeneous sensor fusion setup of a single camera and radar – a challenging calibration setup due to radar's lack of elevation measurement. The fourth setup incorporated a rotating 3D lidar with a single camera, demonstrating the validity of the method on the fusion of a continuous sweeping sensor and a discrete-time acquisition sensor. Where applicable, we compared the proposed method to the state-of-the-art approaches and the results showed that the proposed method outperformed other approaches and that it reliably estimated the time delay up to a fraction of the sampling rate of the faster sensor. In the end, we discussed the computational complexity of the proposed method and the influence of hyperparameters, mainly the process noise used in the Gaussian process regression.

The subject of future research and the potential of the proposed method is to serve as the base for online calibration of autonomous vehicle or robot heterogeneous sensors by tracking multiple moving targets in the environment – an information that is potentially already available in most autonomous systems navigating in dynamic environments.

ACKNOWLEDGMENT

This work has been supported by the European Regional Development Fund under the grants KK.01.2.1.01.0022 (SafeTRAM) and KK.01.1.1.01.0009 (DATACROSS).

REFERENCES

[1] A. Richardson, J. Strom, and E. Olson, "AprilCal: Assisted and repeatable camera calibration," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1814–1821, 2013.

[2] J. K. Huang, M. Ghaffari, R. Hartley, L. Gan, R. M. Eustice, and J. W. Grizzle, "LiDARTag: A real-time fiducial tag using point clouds," *arXiv preprint*, 2020.

[3] T. Scott, A. A. Morye, P. Pinies, L. M. Paz, I. Posner, and P. Newman, "Choosing a time and place for calibration of lidar-camera systems," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4349–4356, 2016.

[4] J. Levinson and S. Thrun, "Automatic Online Calibration of Cameras and Lasers," in *Robotics: Science and Systems (RSS)*, 2013.

[5] J. Maye, H. Sommer, G. Agamennoni, R. Siegwart, and P. Furgale, "Online self-calibration for robotic systems," *The International Journal of Robotics Research*, vol. 35, no. 4, pp. 357–380, 2015.

[6] N. Keivan and G. Sibley, "Online SLAM with any-time self-calibration and automatic change detection," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 5775–5782.

[7] A. Fornaser, P. Tomasin, M. De Cecco, M. Tavernini, and M. Zanetti, "Automatic graph based spatiotemporal extrinsic calibration of multiple Kinect V2 ToF cameras," *Robotics and Autonomous Systems*, vol. 98, pp. 105–125, 2017.

[8] F. Faion, M. Baum, A. Zea, and U. D. Hanebeck, "Depth sensor calibration by tracking an extended object," in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, 2015, pp. 19–24.

[9] P. C. Su, J. Shen, W. Xu, S. C. S. Cheung, and Y. Luo, "A fast and robust extrinsic calibration for RGB-D camera networks," *Sensors (Switzerland)*, vol. 18, no. 1, pp. 1–23, 2018.

[10] F. Lv, T. Zhao, and R. Nevatia, "Camera calibration from video of a walking human," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1513–1518, 2006.

[11] D. F. Glas, T. Miyashita, H. Ishiguro, and N. Hagita, "Automatic position calibration and sensor displacement detection for networks of laser range finders for human tracking," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010, pp. 2938–2945.

[12] D. F. Glas, D. Brscic, T. Miyashita, and N. Hagita, "SNAPCAT-3D: Calibrating networks of 3D range sensors for pedestrian tracking," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 712–719.

[13] Z. Tang, Y. S. Lin, K. H. Lee, J. N. Hwang, J. H. Chuang, and Z. Fang, "Camera self-calibration from tracking of moving persons," in *International Conference on Pattern Recognition (ICPR)*, 2017, pp. 265–270.

[14] J. Quenzel, N. Papenberg, and S. Behnke, "Robust extrinsic calibration of multiple stationary laser range finders," in *IEEE International Conference on Automation Science and Engineering (CASE)*, 2016, pp. 1332–1339.

[15] M. Huber, M. Schlegel, and G. Klinker, "Application of Time-Delay Estimation to Mixed Reality Multisensor Tracking," *Journal of Virtual Reality and Broadcasting*, vol. 11, no. 3, 2014.

[16] J. Kelly and G. S. Sukhatme, "A general framework for temporal calibration of multiple proprioceptive and exteroceptive sensors," *Springer Tracts in Advanced Robotics*, vol. 79, pp. 195–209, 2014.

[17] J. Rehder, R. Siegwart, and P. Furgale, "A General Approach to Spatiotemporal Calibration in Multisensor Systems," *IEEE Transactions on Robotics*, vol. 32, no. 2, pp. 383–398, 2016.

[18] M. Li and A. I. Mourikis, "Online temporal calibration for camera-IMU systems: Theory and algorithms," *International Journal of Robotics Research*, vol. 33, no. 7, pp. 947–964, 2014.

[19] M. K. Ackerman, A. Cheng, B. Shiffman, E. Boctor, and G. Chirikjian, "Sensor calibration with unknown correspondence: Solving AX=XB using Euclidean-group invariants," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 1308–1313.

[20] 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, 2014.

[21] Z. Taylor and J. Nieto, "Motion-Based Calibration of Multimodal Sensor Extrinsics and Timing Offset Estimation," *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1215–1229, 2016.

[22] T. Qin and S. Shen, "Temporal Calibration for Monocular Visual-Inertial Systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 3662–3669.

[23] E. Mair, M. Fleps, M. Suppa, and D. Burschka, "Spatio-temporal initialization for IMU to camera registration," in *International Conference on Robotics and Biomimetics (ROBIO)*, 2011, pp. 557–564.

[24] H. Sommer, R. Khanna, I. Gilitschenski, Z. Taylor, R. Siegwart, and J. Nieto, "A low-cost system for high-rate, high-accuracy temporal calibration for LIDARs and cameras," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 2219–2226.

[25] C. Sommer, V. Usenko, D. Schubert, N. Demmel, and D. Cremers, "Efficient Derivative Computation for Cumulative B-Splines on Lie Groups," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[26] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. MIT Press, 2006.

[27] T. D. Barfoot, *State Estimation for Robotics*, 1st ed. Cambridge University Press, 2017.

[28] T. Barfoot, C. H. Tong, and S. Sarkka, "Batch Continuous-Time Trajectory Estimation as Exactly Sparse Gaussian Process Regression," in *Robotics: Science and Systems*, 2014.

[29] S. Anderson, T. D. Barfoot, C. H. Tong, and S. Särkkä, "Batch nonlinear continuous-time trajectory estimation as exactly sparse Gaussian process regression," *Autonomous Robots*, vol. 39, no. 3, pp. 221–238, 2015. [Online]. Available: https://link.springer.com/article/10.1007/s10514-015-9455-y

[30] M. Mukadam, X. Yan, and B. Boots, "Gaussian Process Motion planning," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9–15, 2016.

[31] N. Wahlström and E. Özkan, "Extended Target Tracking Using Gaussian Processes," *IEEE Transactions on Signal Processing*, vol. 63, no. 63, pp. 4165–4178, 2015.

[32] L. Oth, P. Furgale, L. Kneip, and R. Siegwart, "Rolling shutter camera calibration," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1360–1367, 2013.

[33] G. Grisetti, T. Guadagnino, I. Aloise, M. Colosi, B. Della Corte, and D. Schlegel, "Least Squares Optimization: from Theory to Practice," *arXiv preprint*, 2020.

[34] M. Huber, M. Schlegel, and G. Klinker, "Temporal calibration in multisensor tracking setups," in *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2009, pp. 195–196.
[35] C. Herzberg, "A Framework for Sparse, Non-Linear Least Squares Problems on Manifolds," *Framework*, p. 50, 2008.
[36] J. Sola, J. Deray, and D. Atchuthan, "A micro Lie theory for state estimation in robotics," *arXiv preprint*, 2020.
[37] R. Wagner, O. Birbach, and U. Frese, "Rapid development of manifold-based graph optimization systems for multi-sensor calibration and SLAM," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3305–3312, 2011.
[38] B. D. Corte, H. Andreasson, T. Stoyanov, and G. Grisetti, "Unified Motion-Based Calibration of Mobile Multi-Sensor Platforms with Time Delay Estimation," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 902–909, 2019.
[39] J. Wang and E. Olson, "AprilTag 2: Efficient and robust fiducial detection," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 4193–4198.
[40] J. Peršić, I. Marković, and I. Petrović, "Extrinsic 6DoF calibration of a radar–LiDAR–camera system enhanced by radar cross section estimates evaluation," *Robotics and Autonomous Systems*, vol. 114, 2019.
[41] J. Maye, P. Furgale, and R. Siegwart, "Self-supervised calibration for robotic systems," in *IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 473–480.
[42] D. Zuñiga-Noël, J. R. Ruiz-Sarmiento, R. Gomez-Ojeda, and J. Gonzalez-Jimenez, "Automatic Multi-Sensor Extrinsic Calibration For Mobile Robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2862–2869, 2019.
[43] L. Zhang, Z. Liu, and C. H. Xia, "Clock synchronization algorithms for network measurements," *Proceedings - IEEE INFOCOM*, vol. 1, no. c, pp. 160–169, 2002.
[44] S. Debattisti, L. Mazzei, and M. Panciroli, "Automated extrinsic laser and camera inter-calibration using triangular targets," in *IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 696–701.

Juraj Peršić received his B.Sc. degree in 2014. and M.Sc. degree in 2016., both in electrical engineering from Faculty of Electrical Engineering and Computing (FER), University of Zagreb, Croatia. He has been employed as a researcher on the SafeTRAM project since September 2016 at FER. At the same time he became a Ph.D. student at FER under mentorship of prof. dr. sc. Ivan Petrović. His main areas of interest is mobile robotics with focus on sensor calibration and fusion.

Luka Petrović received his B.Sc. and M.Sc. Degrees in electrical engineering from the University of Zagreb, Faculty of Electrical Engineering and Computing in 2015 and 2017, respectively. During his graduate studies, he was awarded with the Rector's Award (2016) for a practical application in the field of robotics and the Bronze Plaque "Josip Lončar" faculty award (2017) for outstanding academic achievement. His main research interests are in the areas of autonomous systems and robotics with focus on high-dimensional motion planning.

Ivan Marković received the M.Sc. and Ph.D. degrees in electrical engineering from the University of Zagreb, Croatia, in 2008 and 2014, respectively. He is an Associate Professor with University of Zagreb Faculty of Electrical Engineering and Computing, Croatia. He was a visiting researcher at INRIA RennesBretagne Atlantique, Rennes, France under the supervision of Prof. François Chaumette. In 2018 he received the Croatian Academy of Engineering Young Scientist Award "Vera Johanides". He also a member of the IFAC Technical Committee on Robotics and serves as the vice-president of the Croatian IEEE RAS chapter. His research interests include estimation theory with applications to autonomous mobile robotic systems.

Ivan Petrović is a professor and the head of the Laboratory for Autonomous Systems and Mobile Robotics at the Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia. His research interests include advanced control and estimation techniques and their application in autonomous systems and robotics. He published more than 60 journal and 200 conference papers. Among others, he is a full member of the Croatian Academy of Engineering, and the Chair of the IFAC Technical Committee on Robotics.