Continuous-Time vs. Discrete-Time Vision-based SLAM: A Comparative Study

Giovanni Cioffi, Titus Cieslewski, and Davide Scaramuzza

***Abstract*—Robotic practitioners generally approach the vision-based SLAM problem through discrete-time formulations. This has the advantage of a consolidated theory and very good understanding of success and failure cases. However, discrete-time SLAM needs tailored algorithms and simplifying assumptions when high-rate and/or asynchronous measurements, coming from different sensors, are present in the estimation process. Conversely, continuous-time SLAM, often overlooked by practitioners, does not suffer from these limitations. Indeed, it allows integrating new sensor data asynchronously without adding a new optimization variable for each new measurement. In this way, the integration of asynchronous or continuous high-rate streams of sensor data does not require tailored and highly-engineered algorithms, enabling the fusion of multiple sensor modalities in an intuitive fashion. On the down side, continuous time introduces a prior that could worsen the trajectory estimates in some unfavorable situations. In this work, we aim at systematically comparing the advantages and limitations of the two formulations in vision-based SLAM. To do so, we perform an extensive experimental analysis, varying robot type, speed of motion, and sensor modalities. Our experimental analysis suggests that, independently of the trajectory type, continuous-time SLAM is superior to its discrete counterpart whenever the sensors are not time-synchronized. In the context of this work, we developed, and open source, a modular and efficient software architecture containing state-of-the-art algorithms to solve the SLAM problem in discrete and continuous time.**

***Index Terms*—SLAM, Mapping, Localization, Sensor Fusion.**

Supplementary Material

The code can be found here https://github.com/uzh-rpg/rpg_vision-based_slam

I. Introduction

SIMULTANEOUS localization and mapping (SLAM) [1] is the problem of building a map of the environment and concurrently estimating the state of the robot. Accurate and robust SLAM algorithms are the keys to unlock autonomous robotic navigation.

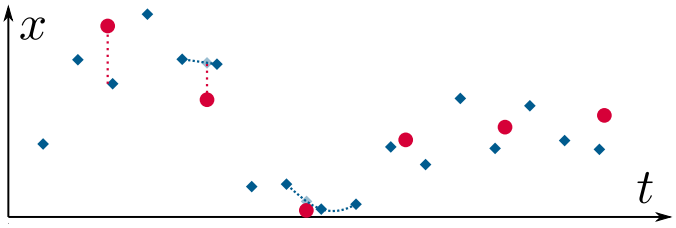
Manuscript received: September, 9, 2021; Revised December, 1, 2021; Accepted December, 27, 2021.

This paper was recommended for publication by Editor Javier Civera upon evaluation of the Associate Editor and Reviewers' comments.

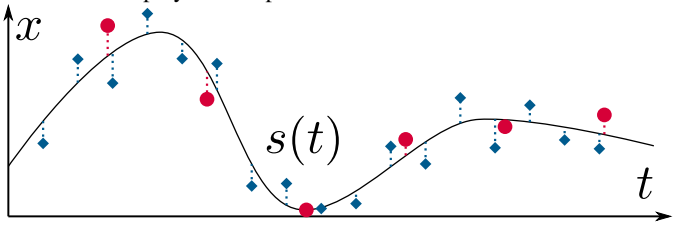
This work was supported by the National Centre of Competence in Research (NCCR) Robotics through the Swiss National Science Foundation (SNSF) and the European Union's Horizon 2020 Research and Innovation Programme under grant agreement No. 871479 (AERIAL-CORE) and the European Research Council (ERC) under grant agreement No. 864042 (AGILEFLIGHT).

The authors are with the Robotics and Perception Group, Department of Informatics, University of Zurich, and Department of Neuroinformatics, University of Zurich and ETH Zurich, Switzerland (http://rpg.ifi.uzh.ch). cioffi@ifi.uzh.ch

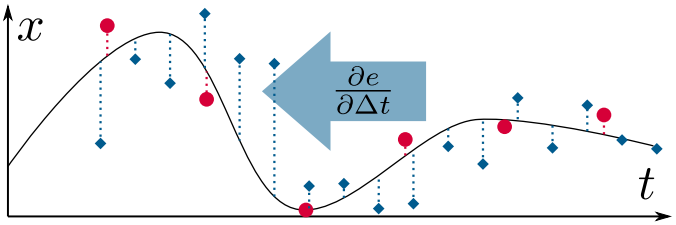
Digital Object Identifier (DOI): see top of this page.



(a) It is typical among popular SLAM methods to represent the state discretely at the measurement times of one of the sensors, e.g., the camera in vision-based SLAM. If the other sensor does not have measurements at the same times, techniques such as interpolation need to be employed to express error terms.



(b) In continuous-time SLAM, the estimated state is instead expressed using a continuous function, e.g., a spline $s(t)$. Now, for any measurement at a time t_i, a meaningful error term can be expressed by comparing the measurement to the spline sample $s(t_i)$, or any of its derivatives $s'(t_i), s''(t_i), ...$, e.g., no integration needed for IMU measurements.



(c) Furthermore, a continuous-time representation allows the simultaneous estimation of time offset or even drift between sensors. To this end, the error terms of one of the sensors is simply expressed with the spline sampled at its measurement times plus a constant time offset shared among all measurements, $s(t_i + \Delta t)$. Then, Δt can be co-optimized with the parameters of s. In this example, the blue measurements would thus be "shifted back to the left".

Fig. 1: Benefits of using a continuous-time state representation illustrated on a simple example where a variable $x(t)$ is estimated from two noisy sensors that measure x at different frequencies (red dots and blue diamonds).

Among the plethora of sensors providing relevant information for localization and mapping, cameras are a very convenient solution in virtue of their information-rich measurements, low cost, and low weight. Vision-based SLAM algorithms use cameras as the main, but not necessarily the only sensor modality. The most common vision-based SLAM formulation

is based on the discrete-time (DT) trajectory representation [2]. Namely, the poses of the camera are estimated at the time a new measurement, i.e., an image, is available. The discrete-time formulation has the benefit of a very consolidated theory. In the past years, many successful applications have been seen [1]. Its limitations are well understood and are addressed in ongoing research.

Although cameras can be used as the only source of information in SLAM systems, fusing multiple sensor modalities is beneficial for accuracy and robustness. In discrete-time SLAM, customized algorithms are necessary to include asynchronous measurements coming from different sources in the estimation process [3]. Similarly, ad-hoc solutions are needed to avoid adding a new state to the estimation problem every time a new measurement is available [4]. In this way, high-rate sensors can be incorporated, thus limiting the increase in computational complexity.

In the past years, researchers have been investigating the use of continuous-time (CT) representations to encode the camera trajectory [5], [6], [7]. Continuous-time based probabilistic SLAM formulations have been derived similarly to the discrete-time case. In particular, applications of continuous-time trajectory representations have been used for multi-sensors calibration [5], [8], planning [9], and 3D reconstruction [10], [11]. The continuous-time formulation brings several advantages to the estimation problem. Firstly, continuous-time trajectories can be sampled at any time. This makes it easy to fuse asynchronous sensors and estimate time offsets, see Fig. 1. Such property is also beneficial for sensors with continuous stream of data, e.g., LiDARs, rolling shutter cameras, and event cameras. Secondly, the continuous-time formulation removes the need to include an optimization variable for every sensor measurement. The computational complexity of the optimization problem is kept bounded, allowing to easily include high-rate sensors, such as inertial measurement units (IMU), in the estimation process. However, the continuous-time representation introduces a prior on the smoothness of the trajectory. Modeling this prior such that it can generalize to different levels of the trajectory smoothness is not an easy task. For example, when using polynomial functions for the continuous-time representation, a not high enough degree of the polynomial function would lead to a loss of details in the trajectory estimates due to excessive smoothing.

To the best of our knowledge, there is no systematic comparison between the continuous- and discrete-time formulations for vision-based SLAM. Such systematic analysis is fundamental to guide the robotic practitioners in the design of future SLAM solutions. Therefore, we perform an extensive quantitative analysis to understand the respective advantages and limitations of the two trajectory representations. We focus on batch SLAM with visual, inertial, and global positional (i.e., Global Positioning System (GPS)) measurements and also investigate the contribution of each sensor modality. IMU and GPS provide local high-rate and global low-rate measurements, respectively, which are complementary to the camera measurements. Camera, IMU, and GPS measurements are fused together to obtain locally accurate and long-term drift-free trajectory estimates. We run experiments in both hardware-in-the-loop simulation and on real-world trajectories of flying and ground robots.

Our experiments indicate that discrete-time and continuous-time representations produce equivalent results when the sensors are time-synchronized. However, when there is an offset in the time synchronization, continuous-time is superior. The main reason behind this result is that the simplifying assumptions made for estimating the time offsets in discrete time do not always hold. These finding are valid for both aerial and ground robots.

To summarize, the main contributions of this work are:

- An in-depth study on the comparison of discrete- and continuous-time trajectory representations in vision-based SLAM.
- Extensive experimental evaluation in hardware-in-the-loop simulation and on real-world data collected from flying and ground robots.
- A modular, efficient software architecture including state-of-the-art algorithms to solve the SLAM problem in the discrete and continuous time. We release the code fully open-source.

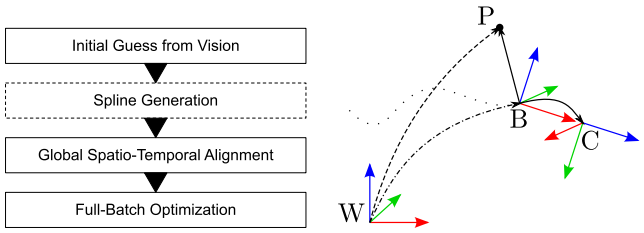
II. Related Work

A popular choice for approximating continuous functions are temporal basis functions. In the set of temporal basis functions, B-splines [5], [12] have properties that are useful for trajectory representation: *locality* and *smoothness*. B-splines define the continuous time evolution by using a set of control points. Sampling at any time only depends on a local subset of the control points (*locality*). The size of this subset corresponds to the order k of the B-spline. A B-spline of order k is a function of class C^{k-1} (*smoothness*). Other suitable choices of basis functions are wavelets [13]. Gaussian processes have also been used to represent continuous-time trajectories for batch state estimation problems [14].

Recent works have tried to reduce the computational complexity of sampling and computing derivatives from B-splines. In [12], a novel derivation for the computation of the time derivatives of cumulative B-spline is proposed. This approach reduces the computational complexity from quadratic to linear in the order of the spline. A novel efficient non-uniform split interpolation for cumulative B-splines is proposed in [15].

The most related applications of continuous-time trajectory representation to our work are multi-sensor calibration [5], [8], 3D reconstruction [6], [10], visual-inertial odometry for event cameras [7], and batch state estimation [14].

The literature on the classical discrete-time SLAM formulation is very broad. Due to space constraints, we refer the reader to the survey paper [1] for an overview on the discrete SLAM formulation. Differently from continuous time, specific algorithms are needed to estimate the time offsets between different sensors. In our discrete-time implementation, we use the method proposed in [3] to estimate the camera-IMU time offset. A similar approach to the one in [16] is used for the GPS-IMU time offset estimation. We describe these two methods in more details in Sec. III-B.



(a) Estimation flowchart (b) Reference frames

Fig. 2: (a) The steps involved in the batch trajectory estimation. Spline generation only applies in continuous-time estimation. (b) The frames and the positions involved in the estimation process. Solid black lines indicate optimizable, calibrated transformations, the dashed lines the position measurements, and the dashed-dotted lines the time-varying, estimated transformation. The dotted lines indicate the trajectory of the latter. Red, green and blue arrows indicate x-, y- and z-axes of the frames.

III. Methodology

We solve the estimation problem using a multi-step approach that involves a few initialization steps before the full-batch optimization. Fig. 2 (a) shows the flow chart of the optimization pipeline for the both continuous- and discrete-time approaches. The reference frames are depicted in Fig. 2 (b). W is the fixed world frame, whose z axis is aligned with the gravity. When available, GPS measurements are expressed in this frame. B is the moving body frame. We set it equal to the IMU frame. C is the camera frame. P is the GPS antenna position. Note that this is not a full 6-DOF frame, as no orientation is provided by the GPS measurement. We use the notation $(\cdot)^w$ to represent a quantity in the world frame W. Similar notation applies for each reference frame. The position, orientation, and velocity of B with respect to W at time t_k are written as $\mathbf{p}^w_{b_k} \in \mathbb{R}^3$, $\mathbf{R}^w_{b_k} \in \mathbb{R}^{3\times3}$ part of the 3-D rotation group $SO(3)$, and $\mathbf{v}^w_{b_k} \in \mathbb{R}^3$, respectively. We use 4×4 matrices, $\mathbf{T} \in SE(3)$ (the special Euclidean group) to express 6-DOF Euclidean transformations.

GPS measurements consist in the position of the GPS antenna in the world frame, $\mathbf{p}^w_p \in \mathbb{R}^3$. The position of the i-th 3-D visual landmark in the world frame is $\mathbf{p}^w_{l_i} \in \mathbb{R}^3$. The set $\mathcal{L}$ contains the 3-D visual landmarks. The time t^c_i is the time offset between camera and IMU such that $t_{imu} = t_{cam} + t^c_i$. Using the same convention, t^g_i is the GPS-IMU time offset.

The state vectors containing the optimization variables are ${}^c\mathcal{X}$ and ${}^d\mathcal{X}$, for the continuous- and discrete-time approaches, respectively. They are introduced in Sec. III-A and Sec. III-B. The transformation $\mathbf{T}^c_i$ is the extrinsic calibration matrix between camera and IMU. The vector $\mathbf{p}^b_p \in \mathbb{R}^3$ is the position of the GPS antenna in the body frame. We use different representations for the IMU biases in continuous time, ${}^c\mathcal{B}$, and discrete time, ${}^d\mathcal{B}$. They are introduced in Sec. III-A and Sec. III-B, respectively.

In this work, we focus on global shutter cameras, which are the common camera choice in vision-based SLAM systems. The initial camera poses and 3-D landmarks are obtained by using COLMAP [17]. COLMAP is a monocular Structure-from-Motion pipeline which is widely used in the computer vision and robotic community. The camera poses and 3-D landmarks reconstructed by COLMAP are expressed in the scaleless reference frame G, which is a fixed frame defined such that the first camera pose is equal to the identity. We use the notation $\bar{\cdot}$ to denote a sensor measurement and the notation $\hat{\cdot}$ to denote a ground-truth measurement.

A. Continuous-time representation

We use cumulative B-splines for the continuous-time trajectory representation. A uniform B-spline of order k and $N+1$ control nodes $\mathbf{x}_i$ is defined by

$$\mathbf{x}(t) = \sum_{i=0}^{N} B_{i,k}(t)\mathbf{x}_i, \tag{1}$$

where t is the time variable. The control points are equally spaced in time by an interval Δt, $t_i = t_0 + i\Delta t$, and $B_{i,k}(t)$ are given by the De Boor-Cox recurrence relations [18]. For cumulative B-splines, Eq. (1) can be rewritten as

$$\mathbf{x}(t) = \tilde{B}_{0,k}(t)\mathbf{x}_0 + \sum_{i=0}^{N} \tilde{B}_{i,k}(t)\mathbf{d}_i, \tag{2}$$

with $\tilde{B}_{i,k}(t) = \sum_{s=i}^{N} B_{s,k}(t)$, and $\mathbf{d}_i = \mathbf{x}_i - \mathbf{x}_{i-1}$. The coefficients of a uniform B-spline are constant and can be written in matrix form [19]. This matrix form can be also used for cumulative B-splines. Combining the matrix formulation [19] with the uniform representation proposed in [12], the calculations can be simplified and the equation for sampling from the B-spline at time t becomes

$$\mathbf{x}(u) = \mathbf{x}_i + \sum_{j=1}^{k-1} \lambda_j(u) \cdot \mathbf{d}^i_j. \tag{3}$$

The uniform time representation proposed in [12] defines $u(t)$, with $t \in [t_i, t_{i+1})$, as the normalized time elapsed since the start of the segment. Defined $h(t) = \frac{t-t_0}{\Delta t}$, u can be computed as $u(t) = h(t) - i$. The coefficients $\lambda_j(u) \in \mathbb{R}$ are constant and only depend on the order of the B-spline (see Eq. (18-21) in [12]). The difference vector is $\mathbf{d}^i_j = \mathbf{x}_{i+j} - \mathbf{x}_{i+j-1}$. In [20], the cumulative B-spline formulation was derived for elements of Lie groups. In the Lie group $SO(3)$, the addition corresponds to the matrix multiplication, but scaling by a scalar λ is not defined. The scaling operation requires to map an element, $\mathbf{R} \in SO(3)$, of the group to a vector space (the Lie algebra), performing the scaling operation, and then remapping to the Lie group: $\mathrm{Exp}(\lambda \cdot \mathrm{Log}(\mathbf{R}))$. Elements of the Lie algebra can be mapped to the Lie group by the exponential map, $\mathrm{Exp}(\cdot)$. The inverse operation is the logarithm map, $\mathrm{Log}(\cdot)$. The cumulative B-spline formulation for $SO(3)$ is

$$\mathbf{R}(u) = \mathbf{R}_i \cdot \prod_{j=1}^{k-1} \mathrm{Exp}(\lambda_j(u) \cdot \mathbf{d}^i_j), \tag{4}$$

where $\mathbf{d}^i_j = \mathrm{Log}(\mathbf{R}^{-1}_{i+j-1} \cdot \mathbf{R}_{i+j})$. We use two B-splines to represent the position, $\mathbf{p}(u) \in \mathbb{R}^3$, and orientation, $\mathbf{R}(u) \in SO(3)$, of the trajectory we are interested in estimating. We use the notation ${}^c\mathcal{T} = \{\mathbf{p}_i, \mathbf{R}_i\}$, with $i = 0, \cdots, N$, and $N+1$ the number of control nodes, to denote the

continuous-time representation of a 6-DOF trajectory. Using one (${}^c\mathcal{T} = \{\mathbf{T}_i\}, \mathbf{T} \in SE(3)$) or two splines to represent a 6-DOF trajectory is equivalent, but the two splines solution is computationally less expensive [12], [21], [6]. We compute the time derivatives of the B-spline as proposed in [12], which reduces the number of matrix operations from $\mathcal{O}(k^2)$ to $\mathcal{O}(k)$.

1) Initialization: The first step of our continuous-time trajectory estimation pipeline is to fit a B-spline to the K camera poses estimated by COLMAP: $\mathbf{p}^g_{c_k}, \mathbf{R}^g_{c_k}$. The initial values of the N control nodes, $\mathbf{p}^g_{c_i}$ and $\mathbf{R}^g_{c_i}$, are obtained by linearly interpolating the COLMAP camera poses at the times $t_i = t_0 + i\Delta t$, where t_0 is the time of the first camera pose and Δt is the inverse of the control node frequency. Then, the control nodes are optimized to minimize the cost function

$$\min_{\mathbf{p}^g_{c_i}, \mathbf{R}^g_{c_i}} \sum_{j=1}^{M} \left\| \mathbf{p}^g_c(u(t_j)) - \mathbf{p}^g_{c_j} \right\|^2 + \left\| \mathrm{Log}(\mathbf{R}^g_c(u(t_j))^t \cdot \mathbf{R}^g_{c_j}) \right\|^2, \tag{5}$$

where M is the number of errors. The measurements $\mathbf{p}^g_{c_j}$ are obtained by the linear interpolation of the camera poses $\mathbf{p}^g_{c_k}$ and $\mathbf{p}^g_{c_{k+1}}$, with $t_j \in [t_k, t_{k+1})$. We similarly use spherical linear interpolation (SLERP) for the rotations $\mathbf{R}^g_{c_j}$. We minimize the cost function till convergence, which usually happens in less than 20 iterations. The result of this step is the continuous-time trajectory ${}^c\mathcal{T}^g_c = \{\mathbf{p}^g_{c_i}, \mathbf{R}^g_{c_i}\}$ that represents the camera poses in the reference frame G. We transform the camera poses in body poses and obtain the trajectory ${}^c\mathcal{T}^g_b = \{\mathbf{p}^g_{b_i}, \mathbf{R}^g_{b_i}\}$. The initial value of $\mathbf{T}^c_i$ is obtained by using the calibration toolbox Kalibr [8].

The second step of our continuous-time trajectory estimation pipeline is to estimate the actual scale of the trajectory ${}^c\mathcal{T}^g_b$ as well as to find a transformation that aligns it to the gravity aligned frame. When GPS measurements are available, we obtain an initial estimation of the 6-DOF transformation $\mathbf{T}^w_g$ and scale s using the method proposed in [22]. This method finds the least-squares solution that minimizes, with respect to $\mathbf{T}^w_g$ and s, the differences between $\mathbf{p}^g_b(u(t_j))$, sampled from the spline, and the GPS measurements $\bar{\mathbf{p}}^w_{p_j}$. Here we assumes that the GPS antenna-IMU positional offset, $\mathbf{p}^b_p$, is null. Then, $\mathbf{T}^w_g$, s, $\mathbf{p}^b_p$, and t^g_i, are estimated by minimizing the cost

$$\min_{\mathbf{T}^w_g, s, \mathbf{p}^b_p, t^g_i} \sum_{j=1}^{D} \left\| \bar{\mathbf{p}}^w_{p_j} - \mathbf{p}^w_{p_j} \right\|^2, \tag{6}$$

where the j-th $\in [1, \cdots, D)$ GPS measurement, $\bar{\mathbf{p}}^w_{p_j}$, is available at time t_j. The predicted antenna position is sampled from the spline as: $\mathbf{p}^w_{p_j} = s\mathbf{R}^w_{b_j}\mathbf{p}^b_p + \mathbf{p}^w_{b_j}$, $\mathbf{R}^w_{b_j} = \mathbf{R}^w_g\mathbf{R}^g_b(u(t_j + t^g_i))$, and $\mathbf{p}^w_{b_j} = s\mathbf{R}^w_g\mathbf{p}^g_b(u(t_j + t^g_i)) + \mathbf{p}^w_g$.

In the case when we do not use GPS measurements, we leverage the IMU measurements to obtain an initial estimate of the scale. We integrate the IMU measurements for a short period of time, usually few seconds, to obtain a small trajectory segment. This trajectory is expressed in a gravity aligned frame, I, which is estimated from the accelerometer measurements collected when the IMU is static. Similarly as before, we use [22] to obtain the transformation $s, \mathbf{T}^i_g$. This transformation is applied to transform ${}^c\mathcal{T}^g_b$ to the frame I.

2) Full-batch optimization: We use $\mathbf{T}^w_g$ and s estimated in the initialization step to transform the trajectory ${}^c\mathcal{T}^g_b$ to the global frame W (or I in the case when GPS is not used): ${}^c\mathcal{T}^w_b = \{\mathbf{p}^w_{b_i}, \mathbf{R}^w_{b_i}\}$. Similarly, the 3-D landmarks $\mathbf{p}^g_{l_r}$ are also transformed to W: $\mathbf{p}^w_{l_r} = s\mathbf{R}^w_g\mathbf{p}^g_{l_r} + \mathbf{p}^w_g$. In the full-batch optimization, the state vector ${}^c\mathcal{X} = \{{}^c\mathcal{T}^w_b, \mathcal{L}, t^c_i, \mathbf{T}^c_i, t^{\mathrm{g}}_i, \mathbf{p}^b_p, \mathbf{g}^w, {}^c\mathcal{B}\}$, is estimated by minimizing the cost function

$$\begin{aligned}\min_{{}^c\mathcal{X}} \sum_{k=1}^{K} \sum_{r \in \mathcal{R}_k} \left\| \mathbf{e}^{\mathrm{v}}_{k,r} \right\|^2_{\mathbf{W}_{\mathrm{v}}} + \sum_{m=1}^{M} (\|\mathbf{e}^{\mathrm{a}}_m\|^2_{\mathbf{W}_{\mathrm{a}}} + \|\mathbf{e}^{\omega}_m\|^2_{\mathbf{W}_{\mathrm{w}}}) + \\ \sum_{d=1}^{D} \left\| \mathbf{e}^{\mathrm{gps}}_d \right\|^2_{\mathbf{W}_{\mathrm{gps}}} + \sum_{f=1}^{F} (\left\| \mathbf{e}^{\mathrm{b_a}}_f \right\|^2_{\mathbf{W}_{\mathrm{b_a}}} + \left\| \mathbf{e}^{\mathrm{b}_\omega}_f \right\|^2_{\mathbf{W}_{\mathrm{b}_\omega}}).\end{aligned} \tag{7}$$

Optimizing this cost function results in the maximum a posteriori (MAP) estimation of the state vector ${}^c\mathcal{X}$, which is derived by using the probabilistic continuous SLAM formulation and the Gaussian distribution for all the measurements [5]. The error $\mathbf{e}^{\mathrm{v}}_{k,r} = \bar{\mathbf{z}}^{k,r} - \pi(\mathbf{R}^c_w(u)\mathbf{p}^w_{l_r} + \mathbf{p}^c_w(u))$ is the visual residuals, which describes the re-projection error of the landmark $\mathbf{p}^w_{l_r}$. The camera pose in the world frame is obtained by sampling and inverting the position and orientation from the spline at $t_k + t^c_i$: $\mathbf{R}^w_c(u) = \mathbf{R}^w_b(u(t_k + t^c_i))\mathbf{R}^i_c$, $\mathbf{p}^w_c(u) = s\mathbf{R}^w_b(u(t_k + t^c_i))\mathbf{p}^i_c + \mathbf{p}^w_b(u(t_k + t^c_i))$. The set $\mathcal{R}_k$ contains all the landmarks that project to the frame k. The image feature measurements $\bar{\mathbf{z}}^{k,r}$ are obtained from COLMAP. The function $\pi(\cdot)$ denotes the camera projection model and t_k is the timestamp of the image.

The value $\mathbf{e}^{\mathrm{a}}_m = \mathbf{R}^w_b(u(t_m))^{-1}(\ddot{\mathbf{p}}^w_b(u(t_m)) + \mathbf{g}^w) - \bar{\mathbf{a}}_m + \mathbf{b}_a(u(t_m))$ is the m-th accelerometer residual with respect to the measurement $\bar{\mathbf{a}}_m$. The quantity $\ddot{\mathbf{p}}^w_b(u(t_m))$ is the second derivative of the B-spline encoding the trajectory positions. The vector $\mathbf{g}^w$ represents the gravity. We use cubic B-splines to represent accelerometer and gyroscope biases, $\mathbf{b}_a(u)$ and $\mathbf{b}_\omega(u)$ as in [8]. The m-th gyroscope residual is $\mathbf{e}^{\omega}_m = \boldsymbol{\omega}(u(t_m)) - \bar{\boldsymbol{\omega}}_m + \mathbf{b}_\omega(u(t_m))$. We refer the reader to Eq. (38) in [12] for the formula to derive $\boldsymbol{\omega}(u)$. The errors $\mathbf{e}^{\mathrm{b_a}}_f$ and $\mathbf{e}^{\mathrm{b}_\omega}_f$ are residuals on the rate of the bias changes. The GPS errors $\mathbf{e}^{\mathrm{gps}}_d$ are the same as the ones defined in Eq. (6). The matrices $\mathbf{W}$ are the weights of the residuals. Their entries are derived from the sensor noise characteristics. We assume Gaussian noise with standard deviation of 1 pixel for the 2-D image features.

B. Discrete-time representation

In the discrete-time formulation, the trajectory is represented by the body poses at the rate of the camera: ${}^d\mathcal{T}^w_b = \{\mathbf{p}^w_{b_k}, \mathbf{R}^w_{b_k}\}$. The camera-IMU time offset is estimated using the method proposed in [3], which is integrated in VINS-Mono [23], a state-of-the-art visual-inertial odometry pipeline. This method proposes to shift the 2-D image features to account for the time offset between camera and IMU measurements. It makes the assumption that the camera motion has constant velocity in a short period of time (e.g., between consecutive frames), and, based on this assumption, it calculates the velocity of the 2-D features on the image plane. This velocity is then used to shift the feature position in the small

period of time that corresponds to the camera-IMU time delay (see Eq. (4) in [3]). This allows to include the time offset in the estimation process. To define the GPS errors, the trajectory is interpolated at the time of the GPS measurements. The IMU-GPS time offset is taken into account in the interpolation factor similarly as in [16].

1) Initialization: Similarly to Sec. III-A1, we compute the body poses from the camera poses estimated by COLMAP and then transform them to the world frame W using the 6-DOF and scale transformation obtained by applying [22].

2) Full-batch optimization: Using a similar probabilistic SLAM formulation as in Sec. III-A2, we derive the cost function to minimize

$$\min_{{}^{d}\mathcal{X}} \sum_{k=1}^{K} \sum_{r \in \mathcal{R}_k} \left\| \mathbf{e}^{\mathrm{v}}_{k,r} \right\|^2_{\mathbf{W}_{\mathrm{v}}} + \sum_{k=1}^{K} \left\| \mathbf{e}^{\mathrm{i}}_{k} \right\|^2_{\mathbf{W}_{\mathrm{i}}} + \sum_{k=1}^{K} (\left\| \mathbf{e}^{\mathrm{b_a}}_{k} \right\|^2_{\mathbf{W}_{\mathrm{b_a}}} + \left\| \mathbf{e}^{\mathrm{b}_\omega}_{k} \right\|^2_{\mathbf{W}_{\mathrm{b}_\omega}}) + \sum_{d=1}^{D} \left\| \mathbf{e}^{\mathrm{gps}}_{d} \right\|^2_{\mathbf{W}_{\mathrm{gps}}}, \tag{8}$$

The state vector is ${}^{d}\mathcal{X} = \{ {}^{d}\mathcal{T}^{w}_{b}, \mathcal{V}^{w}_{b}, \mathcal{L}, t^{c}_{i}, \mathbf{T}^{c}_{i}, t^{g}_{i}, \mathbf{p}^{b}_{p}, {}^{d}\mathcal{B} \}$. The set $\mathcal{V}^{w}_{b}$ contains the velocity vectors: $\mathbf{v}^{w}_{b_k}$. The set ${}^{d}\mathcal{B}$ contains the accelerometer and gyroscope bias vectors: $\mathbf{b}_{\mathrm{a}_k}$ and $\mathbf{b}_{\omega_k}$. The initial 3-D landmarks positions in W are obtained similarly as described in III-A2. The reprojection errors $\mathbf{e}^{\mathrm{v}}_{k,r}$ and the GPS errors $\mathbf{e}^{\mathrm{gps}}_{d}$ are defined in the same way as in Sec.III-A2 with the only difference that trajectory poses are represented in discrete time. The errors $\mathbf{e}^{\mathrm{i}}_{k}$ are the inertial residuals. We compute these residuals using the IMU preintegration formulation proposed in [4]. Each inertial error $\mathbf{e}^{\mathrm{i}}_{k}$ constrains the consecutive $k-1$ and k poses and velocities according to the preintegrated IMU measurements in $[t_{k-1}, t_k]$ (see Eq. (45) in [4]). The errors $\mathbf{e}^{\mathrm{b_a}}_{k}$ and $\mathbf{e}^{\mathrm{b}_\omega}_{k}$ constrain the bias random walk (see Eq. (48) in [4]).

IV. EXPERIMENTS

We compare the continuous- and discrete-time representations in terms of accuracy of the estimated trajectory and time offsets. To this end, we use the metrics [24]:

- Positional absolute trajectory error ($\mathrm{ATE_P}$) [m] : $\sqrt{\frac{1}{N}\sum_{k=0}^{N-1} \left\| \mathbf{p}^{w}_{b_k} - \hat{\mathbf{p}}^{w}_{b_k} \right\|^2}$.
- Rotational absolute trajectory error ($\mathrm{ATE_R}$) [deg] : $\sqrt{\frac{1}{N}\sum_{k=0}^{N-1} \left\| \mathrm{Log}((\mathbf{R}^{w}_{b_k})^{t} \cdot \hat{\mathbf{R}}^{w}_{b_k}) \right\|^2}$

In the case of continuous time, the trajectory is sampled at the timestamp of the camera measurements to obtain $\mathbf{p}^{w}_{b_k}$ and $\mathbf{R}^{w}_{b_k}$. We run multiple experiments in hardware-in-the-loop simulation and on two real-world datasets. The hardware-in-the-loop simulation allows a thorough evaluation of the capability of the two trajectory representations in estimating the time offsets since ground-truth values are known. We investigate if the type of robot, flying or ground robot, has effects on the trajectory representation using the two real-world datasets. We use the Ceres Solver [25] and choose the Levenberg-Marquardt algorithm for the optimization. Derivatives are computed using the automatic differentiation method included in the solver.

TABLE I: Ablation study on the order of the B-spline. $\mathrm{ATE_P}$ is in meters, $\mathrm{ATE_R}$ is in degrees, time offset t^{c}_{i} is in milliseconds. In bold the best value for each sequence.

Order	Ev. metric	EuRoC sequence					
		V101	V102	V103	V201	V202	V203
4	$\mathrm{ATE_P}$ [m]	**0.024**	**0.014**	**0.011**	**0.011**	0.011	0.024
	$\mathrm{ATE_R}$ [deg]	**5.5**	**2.1**	**2.3**	**0.6**	0.8	1.0
	t^{c}_{i} [ms]	0.9	3.2	**-1.4**	10.8	1.0	2.2
5	$\mathrm{ATE_P}$ [m]	**0.024**	**0.014**	**0.011**	**0.011**	**0.010**	0.019
	$\mathrm{ATE_R}$ [deg]	**5.5**	2.2	**2.3**	0.9	**0.7**	0.8
	t^{c}_{i} [ms]	0.3	-5.8	2.0	-1.8	**0.0**	0.5
6	$\mathrm{ATE_P}$ [m]	**0.024**	**0.014**	**0.011**	0.012	**0.010**	**0.010**
	$\mathrm{ATE_R}$ [deg]	**5.5**	**2.1**	**2.3**	0.8	**0.7**	**0.6**
	t^{c}_{i} [ms]	**0.2**	**1.3**	**-1.4**	**1.2**	**0.0**	**0.2**

TABLE II: Ablation study on the frequency of the B-spline control nodes. $\mathrm{ATE_P}$ is in meters, $\mathrm{ATE_R}$ is in degrees, estimated time offset t^{c}_{i} is in milliseconds. In bold the best value for each sequence.

Node freq.	Ev. metric	EuRoC sequence					
		V101	V102	V103	V201	V202	V203
1 Hz	$\mathrm{ATE_P}$ [m]	0.028	0.120	0.077	0.020	0.187	0.075
	$\mathrm{ATE_R}$ [deg]	5.9	4.6	7.6	2.0	10.7	5.18
	t^{c}_{i} [ms]	6.6	19.3	1.2	0.0	-45.6	-19.0
5 Hz	$\mathrm{ATE_P}$ [m]	**0.023**	**0.014**	**0.011**	**0.010**	**0.010**	0.019
	$\mathrm{ATE_R}$ [deg]	5.6	**2.1**	2.3	0.9	0.8	0.8
	t^{c}_{i} [ms]	-1.9	-2.8	1.7	4.0	2.6	-1.5
10 Hz	$\mathrm{ATE_P}$ [m]	0.024	**0.014**	**0.011**	0.012	**0.010**	**0.010**
	$\mathrm{ATE_R}$ [deg]	**5.5**	**2.1**	2.3	**0.8**	**0.7**	**0.6**
	t^{c}_{i} [ms]	0.2	1.3	-1.4	1.2	**0.0**	**0.2**
20 Hz	$\mathrm{ATE_P}$ [m]	0.025	**0.014**	**0.011**	**0.010**	**0.010**	**0.010**
	$\mathrm{ATE_R}$ [deg]	**5.5**	**2.1**	**2.2**	1.2	**0.7**	**0.6**
	t^{c}_{i} [ms]	-2.4	**0.7**	**-0.9**	**-0.7**	-1.1	2.0
100 Hz	$\mathrm{ATE_P}$ [m]	0.024	0.226	0.117	0.060	0.168	0.136
	$\mathrm{ATE_R}$ [deg]	8.8	8.4	12.1	6.6	11.1	5.6
	t^{c}_{i} [ms]	**0.0**	-4.1	-3.0	0.0	-1.3	-0.5

We ran all the experiments on Ubuntu 18.04 workstation with an Intel Core i7 3.2GHz Processor and used 8 cores for the optimization. In all the experiments, the optimization problem is solved until convergence. This is always achieved in less than 50 iterations.

A. Hardware-in-the-Loop Simulation: EuRoC Dataset

The EuRoC dataset [26] contains sequences recorded onboard a hex-rotor flying robot equipped with a stereo camera and an IMU. This dataset is well-known for accurate ground-truth and hardware synchronizated sensors. The camera-IMU time offset can be assumed to be zero. We only use the sequences labeled with V_, which contain 6-DOF ground-truth from a motion capture system. Sequences are classified (see Table 2 in [26]) as easy, medium, or hard, reflecting the level of difficulty due to illumination conditions, scene texture and vehicle motion. Difficult sequences contain challenging illumination conditions, (e.g., motion blur), and fast motions with average linear and angular velocities up to 0.9 ms^{-1} and 0.75 rad s^{-1}, respectively. We simulate GPS measurements by downsampling and corrupting the ground-truth positions with zero-mean Gaussian noise. The rate of the simulated GPS measurements is 10 Hz and the standard deviation of the Gaussian noise is 0.1 m. The position offset $\mathbf{p}^{b}_{p}$ is equal to $\mathbf{0} \in \mathbb{R}^3$. We only use images from the right camera.

TABLE III: Comparison of continuous- and discrete-time approaches on the EuRoC dataset. ATE_P is in meters, ATE_R is in degrees, $\hat{t}_i^c$ (ground-truth) and t_i^c (estimated) time offset are in milliseconds. The best values of ATE_P and ATE_R for each sequence are in bold.

Seq.	Continuous-time						Discrete-time					
	$\hat{t}_i^c$ = 0 [ms]		$\hat{t}_i^c$ = 10 [ms]		$\hat{t}_i^c$ = 20 [ms]		$\hat{t}_i^c$ = 0 [ms]		$\hat{t}_i^c$ = 10 [ms]		$\hat{t}_i^c$ = 20 [ms]	
	ATE_P / ATE_R	t_i^c	ATE_P / ATE_R	t_i^c	ATE_P / ATE_R	t_i^c	ATE_P / ATE_R	t_i^c	ATE_P / ATE_R	t_i^c	ATE_P / ATE_R	t_i^c
V101	0.024 / **5.5**	0.2	0.024 / **5.5**	11.0	0.024 / **5.5**	22.2	**0.016** / 5.6	0.3	**0.016** / 5.6	9.1	**0.016** / 5.6	18.6
V102	**0.014** / **2.1**	1.3	**0.014** / **2.1**	9.7	**0.014** / **2.1**	20.6	0.024 / 2.4	0.0	0.026 / 2.3	4.6	0.031 / 2.2	9.3
V103	**0.011** / **2.3**	-1.4	**0.011** / **2.3**	11.8	**0.011** / **2.3**	22.3	0.018 / 2.7	0.0	0.020 / 2.6	3.5	0.024 / 2.6	7.2
V201	0.012 / **0.8**	1.2	0.010 / 0.9	9.7	0.010 / 0.9	19.0	**0.009** / 1.0	0.3	0.010 / 1.0	8.1	0.012 / 1.0	16.4
V202	**0.010** / **0.7**	0.0	**0.010** / **0.7**	10.0	**0.010** / **0.7**	20.0	0.019 / 0.8	0.0	0.021 / 0.9	8.5	0.024 / 1.1	16.7
V203	**0.010** / **0.6**	0.2	**0.010** / **0.6**	10.6	**0.010** / **0.6**	21.5	0.033 / 1.1	0.0	0.036 / 1.2	4.0	0.040 / 1.3	7.5

TABLE IV: Full-batch optimization time. For the continuous-time representation, we used B-splines of order 6 and control frequency $\in [1, 2, 10, 20, 100]$ [Hz]. The time values are in minutes. The trajectory length is in meters. The lowest time per sequence is in bold.

Seq.	Length [m]	CT					DT
		1 Hz	2 Hz	10 Hz	20 Hz	100 Hz	
V101	52.6	13	15	21	36	37	**12**
V102	75.9	**19**	20	25	49	57	28
V103	79.0	24	37	22	46	46	**21**
V201	36.5	6	**5**	13	10	13	15
V202	83.2	44	54	73	44	77	**21**
V203	86.1	18	16	39	50	31	**14**

1) Ablation study on the B-spline: We conducted a study to evaluate the effects of the order and frequency of the control points of the B-spline on the trajectory and camera-IMU time offset estimates. The initial value of the time offset was set to 0 ms. The results of the ablation study on the order of the B-spline are in Table I. A B-spline of order 6, which results in a cubic polynomial encoding accelerations, is needed to correctly estimate the camera-IMU time offset. This conclusion agrees with the findings in [8]. An order higher than 6 does not have any effect on the estimation results. When the spline order is 6, the values of the time offset are close but not exactly 0 ms. Since we solve the optimization till convergence, the solution has reached a local minimum of the cost function. How far is the local minimum from the global minimum is unknown, and, in general, even the unknown global optimum of the MAP estimation can be different from the ground-truth due to modeling errors.

We set the order of the spline to 6 and performed the ablation study on the control node frequency. The results are in Table II. The values of ATE and t_i^c suggest that a frequency of 10 Hz is the optimal choice. Increasing to 20 Hz does not give any significant advantages, while making the optimization computationally more expensive. When the frequency is high, e.g., 100 Hz, the convergence to a good solution is hard to achieve. We conclude that order 6 and control nodes frequency 10/20 Hz are appropriate parameters for B-spline representing trajectories in this dataset.

2) Evaluation of the trajectory representation: In this set of experiments, we evaluated the continuous- and discrete-time trajectory representations in terms of ATE_P, ATE_R, accuracy in estimating the camera-IMU time offset, and computational cost. Following the findings of Sec. IV-A1, we used B-spline of order 6 and control node frequency of 10 Hz. To evaluate the accuracy in estimating the camera-IMU time offset, we simulated delays in the camera data stream, similarly to [3]. We ran experiments with delays of 0, 10, and 20 ms. The results of this comparison are in Table III. These results suggest that when the camera and IMU are time-synchronized both trajectory representations produce similar accuracy (see the column corresponding to $t_i^c = 0$). However, using continuous time gives a lower ATE_P in the sequences V102, V103, V202, and V203, which are medium and difficult sequences containing fast motion of the robot.

When the camera and IMU are not time-synchronized, continuous time is the best trajectory representation. This representation can properly estimate the time offset and produces ATE similar to the case of time-synchronized sensors. In particular, the discrete-time representation suffers in estimating the camera-IMU time offset in fast trajectories. This can be seen from the large difference between the estimated and the true time offset in the sequences V102, V103, V202, and V203. The reason for this result is the assumption of camera motion with constant velocity in the period of time between consecutive camera frames, which is needed to compute the velocity of the 2-D features as described in Sec. III-B. For agile motion, this assumption is no longer accurate. Table IV contains the optimization time, which is the time that the solver takes to converge.

3) Ablation study on the contribution of the different sensor modalities: In this section, we are interested in studying the contributions of each sensor modality, i.e., vision, inertia and GPS, to the accuracy of the trajectory estimation. To this end, we solved the optimization problems in Eq. (7) and Eq. (8) for the continuous- and discrete-time case respectively, with all possible combinations of at least two sensors. For the continuous-time representation, we used B-splines of order 6 with control node frequency of 10 Hz. The values of ATE_P and ATE_R for this analysis are in Table V and Table VI. The results show that vision is the most important sensor modality. For both trajectory representations, the vision-GPS configuration produces the lowest ATE_P in most of the sequences. In this case, the COLMAP reconstructed trajectory is transformed to the gravity aligned frame by using noisy simulated GPS measurements and then re-optimized together with those measurements. COLMAP is able to reconstruct an accurate trajectory on every sequence of this dataset. For this reason, including the inertial measurements in the estimation process does not bring any significant benefit.

TABLE V: Ablation study on the contribution of the different sensor modalities. The error metric is the ATE_P in meters. V: vision, I: inertia, G: simulated GPS. In bold the best value for each sequence.

Sensors	Traj. repr.	EuRoC Sequence					
		V101	V102	V103	V201	V202	V203
V+I+G	CT	0.024	0.014	**0.011**	0.012	0.010	**0.010**
	DT	0.016	0.024	0.018	**0.009**	0.018	0.033
V+G	CT	0.011	**0.013**	0.012	**0.009**	**0.008**	0.012
	DT	**0.010**	0.025	0.024	0.010	0.012	0.029
I+G	CT	0.062	0.102	0.117	0.112	0.164	0.363
	DT	0.139	0.137	0.138	0.138	0.138	0.139
V+I	CT	0.039	0.022	0.014	0.065	0.030	0.015
	DT	0.081	0.106	0.030	0.064	0.022	0.031

TABLE VI: Ablation study on the contribution of the different sensor modalities. The error metric is ATE_R in degrees. V: vision, I: inertia, G: simulated GPS. In bold the best value for each sequence.

Sensors	Traj. repr.	EuRoC Sequence					
		V101	V102	V103	V201	V202	V203
V+I+G	CT	5.5	2.1	2.3	**0.8**	**0.7**	**0.6**
	DT	5.6	2.4	2.7	1.0	0.8	1.1
V+G	CT	5.5	2.3	2.5	1.1	0.8	1.2
	DT	5.6	2.3	2.7	1.1	0.8	0.9
I+G	CT	10.5	6.3	7.6	11.0	11.0	9.4
	DT	12.3	7.7	8.8	11.8	11.8	10.6
V+I	CT	**5.4**	**2.0**	**2.2**	0.9	0.8	**0.6**
	DT	5.5	2.2	2.8	1.0	0.8	1.2

B. Actual GPS with an outdoor flying robot

This dataset, courtesy of [27], contains outdoor flights of a flying robot equipped with a time-synchronized VI sensor (stereo camera and IMU), and a GPS receiver. GPS measurements are available at 5 Hz. We only use images from the left camera. The ground-truth position is provided by a Leica total station. Fig. 3 shows the first trajectory of the dataset. The ATE_P and the time offset estimates are in Table VII. For the continuous-time case, we used B-splines of order 6 and control node frequency of 10 Hz. As suggested by the results in Sec. IV-A, these values are suited for encoding flying robot trajectories. These results confirm the findings of Sec. IV-A: when the sensor are time-synchronized the two trajectory representations produce similar results, as shown by the similar values of ATE_P and camera-IMU time offset.

C. Outdoor Trajectory: Ground robot

This experiment contains an evaluation of a trajectory recorded on-board a ground robot. The robot is equipped with a time-synchronized VI sensor (monocular camera and IMU) and a GPS antenna [1]. The GPS measurements are available at 5 Hz. The 3-D position ground-truth is provided by a RTK-GPS system. Fig. 4 shows the traveled trajectory of the robot. To study the influence of the B-spline order and control point frequency, we ran experiments with orders: 5, 6, and 7, and control node frequency: 10, 20, 100 Hz. Both continuous-time and discrete-time representations produce similar ATE_P as reported in Table VIII. The estimated time offsets are similar for all the configurations listed in Table VIII. In

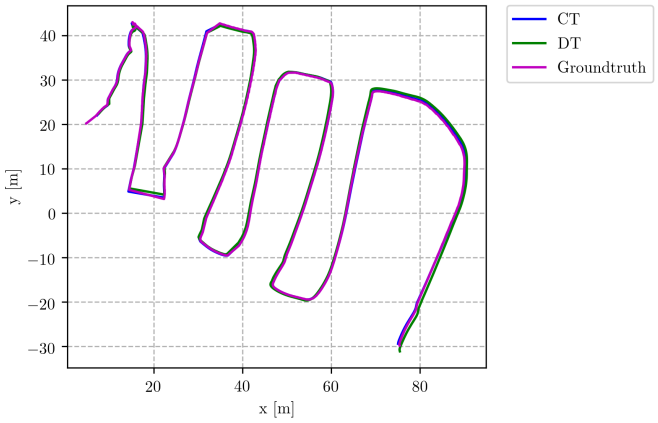


Fig. 3: XY-view of the trajectory flown in Sequence 1 of the outdoor flying robot experiments.

TABLE VII: Comparison of continuous- and discrete-time approaches on the outdoor flying robot dataset. ATE_P is in meters and t_i^c is in milliseconds. The best values of ATE_P for each sequence are in bold.

Err. metric	Traj. repr.	Seq. 1	Seq. 2
ATE_P [m]	**CT**	**0.39**	**0.50**
	DT	0.60	0.86
t_i^c [ms]	**CT**	0.4	0.6
	DT	0.4	0.4
t_i^g [ms]	**CT**	-87.0	-118.0
	DT	-81.0	-119.0

the continuous-time case, with B-spline of order 6 and control node frequency of 10 Hz, t_i^c, and t_i^g are -1.5 ms, and -26.0 ms, respectively. For the discrete-time case, they are -0.8 ms, and -36.3 ms. These results show that the findings of the experiments with a flying robot also apply to the case of a ground robot.

V. CONCLUSIONS

The objective of this work is to compare continuous vs. discrete vision-based SLAM formulations to guide practitioners in the development of SLAM algorithms. We performed ablation and comparative studies in a hardware-in-the-loop simulation with full knowledge of the ground-truth. The ablation studies on the order and frequency of the control nodes of the B-splines suggest that it is necessary to use B-splines of order 6 and control node frequency of at least 10 Hz to accurately estimate the camera-IMU time offset. The comparative studies aimed at comparing continuous- and discrete-time trajectory representations with different levels of camera-IMU time delay. We find that when the camera and IMU are time-synchronized the two representations produce similar results. When a delay is present between the two measurement streams, the continuous-time representation is able to recover an accurate estimate of the time offset and consequently, produces lower ATE. In contrast, the discrete-time formulation fails in estimating the time offset, particularly when the robot moves fast, which consequently leads to high

[1] https://www.fixposition.com/

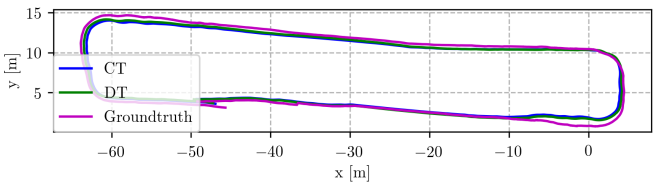


Fig. 4: XY-view of the trajectory traveled by the ground robot.

TABLE VIII: Comparison of continuous- and discrete-time approaches on the outdoor ground robot dataset. The error metric is the ATE_P in meters. The best value is in bold.

	Freq. [Hz]	Order		
		5	6	7
CT	10	0.93	0.92	0.93
	20	1.01	0.95	0.99
	100	0.80	0.89	**0.78**
DT	0.87			

values of the ATE. The main reason of this result is that the assumption, which is necessary to estimate the camera-IMU time offset, of constant velocity of the camera motion in the period of time between consecutive camera frames does not always hold. The findings of the hardware-in-the-loop simulation agree with the results of the experiments on the real-world datasets containing data from aerial and ground robots. In addition, we evaluated the contribution of each sensor modality in an ablation study. We found that on the EuRoC dataset the most important sensor modality is vision. In most of the sequences, the lowest ATE_P is obtained by aligning the camera trajectory obtained from COLMAP to a gravity aligned frame by using the noisy simulated GPS measurements. Including inertial measurements does not give any significant advantage.

VI. Acknowledgments

The authors would like to thank Torsten Sattler and Antonio Loquercio for the fruitful discussions, and the team at Fixposition for the ground robot dataset.

References

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. D. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *IEEE robotics & automation magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[3] T. Qin and S. Shen, "Online temporal calibration for monocular visual-inertial systems," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2018.

[4] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual-inertial odometry," *IEEE Trans. Robot.*, vol. 33, no. 1, pp. 1–21, 2017.

[5] P. Furgale, T. D. Barfoot, and G. Sibley, "Continuous-time batch estimation using temporal basis functions," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2012.

[6] H. Ovrén and P.-E. Forssén, "Trajectory representation and landmark projection for continuous-time structure from motion," *Int. J. Robot. Research*, vol. 38, no. 6, pp. 686–701, 2019.

[7] E. Mueggler, G. Gallego, H. Rebecq, and D. Scaramuzza, "Continuous-time visual-inertial odometry for event cameras," *IEEE Trans. Robot.*, vol. 34, no. 6, pp. 1425–1440, Dec. 2018.

[8] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2013.

[9] W. Ding, W. Gao, K. Wang, and S. Shen, "An efficient b-spline-based kinodynamic replanning framework for quadrotors," *IEEE Trans. Robot.*, vol. 35, no. 6, pp. 1287–1306, 2019.

[10] A. J. Yang, C. Cui, I. A. Bârsan, R. Urtasun, and S. Wang, "Asynchronous multi-view slam," *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2021.

[11] H. Ovrén and P.-E. Forssén, "Spline error weighting for robust visual-inertial fusion," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018.

[12] C. Sommer, V. Usenko, D. Schubert, N. Demmel, and D. Cremers, "Efficient derivative computation for cumulative b-splines on lie groups," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2020.

[13] S. Anderson, F. Dellaert, and T. D. Barfoot, "A hierarchical wavelet decomposition for continuous-time SLAM," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2014.

[14] 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.

[15] D. Hug and M. Chli, "On conceptualizing a framework for sensor fusion in continuous-time simultaneous localization and mapping," in *3D Vision (3DV)*, 2020.

[16] W. Lee, K. Eckenhoff, P. Geneva, and G. Huang, "Intermittent gps-aided vio: Online initialization and calibration," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2020.

[17] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2016.

[18] C. de Boor, *A practical guide to splines*. Springer Verlag New York, 2001.

[19] K. Qin, "General matrix representations for B-splines," *The Visual Computer*, vol. 16, no. 3–4, pp. 177–186, 2000.

[20] H. Sommer, J. R. Forbes, R. Siegwart, and P. Furgale, "Continuous-time estimation of attitude using b-splines on lie groups," *Journal of Guidance, Control, and Dynamics*, vol. 39, no. 2, pp. 242–261, 2016.

[21] A. Haarbach, T. Birdal, and S. Ilic, "Survey of higher order rigid body motion interpolation methods for keyframe animation and continuous-time trajectory estimation," in *3D Vision (3DV)*, 2018.

[22] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 4, 1991.

[23] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, 2018.

[24] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2018.

[25] A. Agarwal, K. Mierle, and Others, "Ceres solver," http://ceres-solver.org.

[26] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *Int. J. Robot. Research*, vol. 35, no. 10, pp. 1157–1163, 2015.

[27] J. Surber, L. Teixeira, and M. Chli, "Robust Visual-Inertial Localization with Weak GPS Priors for Repetitive UAV Flights," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2017.