Continuous-Time Fixed-Lag Smoothing for LiDAR-Inertial-Camera SLAM

Jiajun Lv, Xiaolei Lang, Jinhong Xu, Mengmeng Wang, Yong Liu, and Xingxing Zuo

***Abstract*—Localization and mapping with heterogeneous multisensor fusion have been prevalent in recent years. To adequately fuse multimodal sensor measurements received at different time instants and different frequencies, we estimate the continuous-time trajectory by fixed-lag smoothing within a factor-graph optimization framework. With the continuous-time formulation, we can query poses at any time instants corresponding to the sensor measurements. To bound the computation complexity of the continuous-time fixed-lag smoother, we maintain temporal and keyframe sliding windows with constant size, and probabilistically marginalize out control points of the trajectory and other states, which allows preserving prior information for future sliding-window optimization. Based on continuous-time fixed-lag smoothing, we design tightly coupled multimodal SLAM algorithms with a variety of sensor combinations, like the LiDAR-inertial and LiDAR-inertial-camera SLAM systems, in which online time offset calibration is also naturally supported. More importantly, benefiting from the marginalization and our derived analytical Jacobians for optimization, the proposed continuous-time SLAM systems can achieve real-time performance regardless of the high complexity of continuous-time formulation. The proposed multimodal SLAM systems have been widely evaluated on three public datasets and self-collect datasets. The results demonstrate that the proposed continuous-time SLAM systems can achieve high-accuracy pose estimations and outperform existing state-of-the-art methods. To benefit the research community, we will open source our code at https://github.com/APRIL-ZJU/clic.**

***Index Terms*—Continuous-time trajectory, fixed-lag smoothing, multisensor fusion, simultaneous localization and mapping (SLAM).**

Manuscript received 1 August 2022; revised 19 November 2022; accepted 3 January 2023. Date of publication 15 February 2023; date of current version 16 August 2023. Recommended by Technical Editor C. C. L. Wang and Senior Editor K. J. Kyriakopoulos. This work was supported by NSFC under Grant 62088101, Autonomous Intelligent Unmanned Systems. *(Corresponding authors: Yong Liu; Xingxing Zuo.)*

Jiajun Lv, Xiaolei Lang, Jinhong Xu, Mengmeng Wang, and Yong Liu are with the State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China, and also with the Huzhou Institute of Zhejiang University, Hangzhou 310027, China (e-mail: yongliu@iipc.zju.edu.cn).

Xingxing Zuo is with the School of Computation, Information and Technology, Technical University of Munich, 80333 Munich, Germany (e-mail: xingxingzuo@zju.edu.cn).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TMECH.2023.3241398.

Digital Object Identifier 10.1109/TMECH.2023.3241398

I. Introduction

SIMULTANEOUS localization and mapping (SLAM) are fundamental for mobile robots to navigate autonomously in various applications. Especially, in scenarios where external signals are unavailable, e.g., the GPS-denied environment, localization, and mapping with only onboard sensors can be a practical solution. Plenty of sensors have been used for SLAM purposes, including the proprioceptive sensors and exteroceptive sensors. The common proprioceptive sensors measure the state of robot, e.g., wheel encoders, inertial measurement unit (IMU), magnetometer, etc. While the exteroceptive sensors perceive the surrounding environment information, e.g., radar, sonar, LiDAR, camera, barometer, altimeter, etc. Among all of the sensors, camera, IMU and LiDAR are three of the most ubiquitous sensors leveraged in SLAM algorithms [1], [2], [3], [4], [5], [6].

Recently, LiDAR-inertial-camera (LIC)-based localization and mapping systems [4], [6], [7] have attracted significant attention, due to their versatility, high accuracy, and robustness. By combining the three sensor modalities, LIC systems have more applicable scenarios than using only the component sensors. Besides, since a single LiDAR only has a limited field of view (FOV), multiple LiDARs fusion [8] is a rational choice for highly accurate localization and efficient mapping.

Multimodal sensor measurements are assigned with timestamps by individual sensors, and timeoffsets generally exist among different sensors [9], [10] without hardware synchronization. Even the timeoffsets between sensors can be removed, measurements from various sensor modalities are usually received at different rates and different time instants. In order to fuse the heterogeneous sensors, accurate alignment of the asynchronous sensors is the prerequisite. Without special hardware support for synchronization, time offset can also be online calibrated in estimators, such as online timeoffset estimation in visual-inertial (VI) system [9], [11], [12], [13] and LIC system [4]. Further efforts are required to deal with different sensor frequencies and sampling time instants. For example, sensors like IMU and LiDAR consecutively provide abundant measurements at high frequencies, and it is challenging to accurately align thousands of points in LiDAR scans to the asynchronous IMU measurements. Some approaches [6], [10], [14] linearly interpolate the integrated discrete-time IMU poses at the sampling time instants of LiDAR points, in order to compensate for motion distortion in LiDAR scans.

Another alternative way is to parameterize the trajectory in a continuous-time representation [15], [16], [17], which allows

TABLE I
RELATED WORKS OF LIC FUSION

Paper	Year	IMU[1]	Camera[2]	LiDAR[3]	Method[4]
V-LOAM [25]	2018-JFR	I1	C1, C3	L1	F1
Wang. [26]	2019-IROS	I2	C1	L1	F2
Khattak. [27]	2019-ICUAS	ROVIO [28]		L1	F2
Lowe. [16]	2018-RAL	I3	C1, C4	L5	F4
LIC-Fusion [4]	2019-IROS	I1	C1	L1	MSCKF
LIC-Fusion2.0 [10]	2020-IROS	I1	C1	L1, L2	MSCKF
LVI-SAM [6]	2021-ICRA	I2	C1, C3	L1	F3
VILENS [29]	2021-RAL	I2	C1, C3	L2	F4
VILENS [30]	2021-Arxiv	I2	C1, C3	L2, L3	F4
R2live [31]	2021-RAL	I1, I2	C1	L1	ESIKF, F4
R3live [7]	2021-Arxiv	I1	C2, C3	L1	ESIKF
Super Odom. [32]	2021-IROS	I2	C1, C3	L4	F3
Lvio-Fusion [33]	2021-IROS	I2	C1	L1	F4

[1] I1=Integration; I2=Preintegration; I3=Raw measurements.
[2] C1=Indirect; C2=Direct; C3=Depth From LiDAR; C4=Depth From Surfel.
[3] L1=LOAM Feature; L2=Tracked Plane/Line; L3=PCA based Feature; L4=PCA based Feature; L5=Surfel.
[4] See Fig. 1 for details.

pose querying at any time instants without interpolations. Formulating the trajectory in continuous-time with B-spline gains popularity in calibration tasks [18], [19], visual odometry [20], as well as LiDAR odometry [17], [21], due to its versatility and convenience in aligning asynchronous sensor measurements. However, there are two main challenges preventing continuous-time trajectory from being widely deployed: 1) *Real-time performance:* In conventional discrete-time state estimation, the pose in the residual is just the state to be estimated. However, in continuous-time state estimation, the pose is computed from multiple control points on the Lie group, and the state to be estimated in the residual switches to multiple control points (depend on the B-spline order). This fact not only increases the computation load but also generally increases the complexity of the Jacobian computation. Existing continuous-time odometry methods [17] rarely achieve real-time performance. 2) *Fixed-lag smoothing:* Discrete-time methods typically estimate states by fixed-lag smoothing [3], [22], [23] and preserve information of the old measurements/states with marginalization, however, continuous-time methods like VIO [24] or LIO [21] rarely consider how to preserve information of old measurements/states. In fact, some of informative measurements/states are directly discarded without leaving prior information for the estimation of remaining states. There is little work about marginalization for continuous-time trajectory optimization, probably because of the high complexity of optimizing continuous-time trajectory, and the sliding strategy and marginalization strategy are significantly different from discrete-time methods.

In general, although B-spline-based continuous-time trajectory optimization methods have been studied in many existing research works [17], [21], [24], the continuous-time fixed-lag smoothing within a sliding window and with probabilistic state marginalization is rarely investigated in existing literature. To the best of our knowledge, this article is among the *first* to utilize continuous-time fixed-lag smoothing with probabilistic marginalization for multisensor fusion, and even better, we can achieve real-time performance.

The contributions can be summarized as follows.

1) We adopt continuous-time fixed-lag smoothing method for multisensor fusion in a factor-graph optimization framework. Specifically, we estimate B-spline-based continuous-time trajectory within a constant size of sliding window by fusing asynchronous heterogeneous sensor measurements published at various frequencies and different time instants. To attain a bounded computation complexity, old control points of the continuous-time trajectory are strategically and probabilistically marginalized out of the sliding window.
2) Powered by continuous-time fixed-lag smoothing, we design some LiDAR-inertial (LI) SLAM and LIC SLAM systems at a variety of sensor combinations (even with multiple LiDARs), and derive the analytical Jacobians for efficient factor-graph optimization. Benefiting from easy accessibility of continuous-time trajectory derivatives, timeoffsets between different sensors can be online calibrated. With careful and strategic implementation, our proposed continuous-time LI SLAM and LIC SLAM systems can achieve real-time performance.
3) The proposed continuous-time LI SLAM and LIC SLAM systems are extensively evaluated on three publicly available datasets and self-collected datasets. The experimental results show that the proposed LI system has competitive accuracy and the proposed LIC system outperforms several state-of-the-art methods and works well in degenerated sequences.

The rest of this article is organized as follows. Section II reviews the relevant literature. Then, with the continuous-time trajectory preliminary provided in Section III, we present continuous-time fixed-lag smoothing method leveraged in Section IV. In Section V, we further detail multisensor fusion SLAM systems enabled by continuous-time fixed-lag smoothing with different sensor configurations, such as LI systems and LIC systems. Section VI demonstrates the performance of different sensor combinations and discloses the runtime of the different systems. Finally, Section VII concludes the article and discusses future work.

II. RELATED WORKS

There is a rich body of literature on multisensor fusion for localization and mapping. Instead of providing a comprehensive literature review of SLAM with multisensor fusion, we extensively review the existing LIC systems, multi-LiDAR systems, and continuous-time trajectory-based SLAM systems, which are most relevant to this article.

A. Multisensor Fusion for SLAM

1) LIC Fusion for SLAM: State estimation in LIC systems has been achieved either by graph-based optimization [6], [16], [25], [26], [29], [30], [32], [33] or filter-based methods [4], [7], [10], while [31] also combines both the graph optimization and filter for localization and mapping. We summarize typical LIC systems in Table I, which depicts the processing methods of raw sensory measurements as well as the state estimation framework. Fig. 1 summarizes four typical pipelines using factor-graph

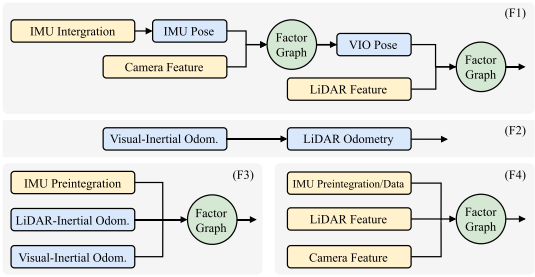


Fig. 1. Four different pipelines of LIC SLAM systems using factor-graph optimization framework. F1, F2, F3 are loosely-coupled methods, and F4 is a tightly-coupled method.

optimization, including the loosely coupled ones (F1, F2, and F3), as well as the tightly coupled methods (F4). V-LOAM [2] is a loosely coupled method, composed of IMU integration, vision and integrated IMU poses fusion with factor-graph optimization, as well as LiDAR and VIO poses fusion with factor-graph optimization (see pipeline F1 in Fig. 1). All the three submodules can output pose estimation at different frequencies, and the estimated poses are refined step by step within the submodules. Some works [26], [27], utilizing VI odometry to provide initial pose guess for LiDAR point cloud registration (see F2 of Fig. 1), adopt another way to loosely couple sensor measurements. Some other works [6], [32] get pose estimation via IMU preintegration, VI odometry and LiDAR-inertial odometry separately, then fuse the three types of pose estimation in a pose graph, which can be solved by factor-graph optimization. In contrast to loosely coupled methods, which somehow fuse intermediate pose estimation from sensor measurements, tightly coupled approaches [16], [29], [30], [33] directly integrate sensor data, estimating system state with IMU data (preintegration/raw measurements), image features, and LiDAR features (illustrated in F4 of Fig. 1).

IMU measurement: Regarding the measurement processing for different sensor modalities, filter-based methods usually use IMU measurements to propagate system states, and optimization-based methods widely adopt preintegration [34] technique, which integrates high-rate IMU data into a low-rate preintegration factor; continuous-time trajectory-based methods naturally couple raw IMU measurements in the batch optimization.

Visual measurement: Visual algorithms could be categorized into the indirect and the direct upon the visual residual models [35]. Indirect methods extract and track features from images and construct geometric constraints in the estimation, while direct methods directly use the actual image values to formulate photometric error, allowing for a more finely grained geometry representation. Both methods rely on accurate depth estimation of landmarks, which can be enhanced by the highly accurate clouds of LiDAR. Specifically, Lowe et al. [16] set the depth of a feature to the nearest surfel along the feature ray, and set the depth uncertainty to the surfel's covariance in the ray direction. Other methods [6], [25], [30], [32] first project the LiDAR cloud onto the spherical coordinates of the image, then associate the image feature with the nearest local planar patch that is formed by several nearest LiDAR points, and finally set the feature depth to the depth of the intersection between the ray (from the camera center to the feature) and the plane.

LiDAR measurement: Line and plane features based on local patch's smoothness firstly defined in LOAM algorithm [2] are popular in data association of LiDAR clouds, followed by several methods [4], [6], [25], [26]. Zuo et al. [10] extend to track consecutive plane feature and VILENS [29], [30] tracks both line and plane feature.

2) Multi-LiDAR Fusion for SLAM: Multi-LiDAR odometry LOCUS [8] and its following [36] combine motion-corrected scans from each LiDAR into a single point cloud, which is registered by generalized iterative closest point (GICP). MILIOM [37] chooses to extract features from raw organized scan first and merges feature cloud of each LiDAR instead of merging the full scan, the feature cloud is registered through feature-to-map matching method.

B. SLAM With Continuous-Time Trajectory

Continuous-time trajectory representation includes linear interpolation, wavelets, Gaussian process, and splines. In this article, we focus on B-spine-based representation as explained in Section III-A. Pioneering work on solving B-spline-based continuous-time SLAM problem is first systematically derived in [38], where the authors propose to represent the states as a weighted sum of continuous temporal basis function and illustrate its application case in the extrinsic calibration between IMU and camera. Thereafter, B-spline-based continuous-time trajectory formulation has been widely applied to SLAM-relevant applications, such as event camera odometry [20], and LiDAR-inertial odometry [17], [21], multicamera SLAM system [24], and LIC fusion [16]. Lowe et al. [16] tightly coupled LIC measurements to estimate state in continuous-time trajectory, however, they assume no timeoffset between sensors, which does not hold well in real world application, and prior information is discarded without any explanation. Our previous work [17] proposed a continuous-time based method for LI system, which managed the measurements within a local window and presented a two-stage loop closure strategy to obtain global-consistent trajectory. Some IMU measurements older than current scan are included in local window rather than applying marginalization technique to retain information about the old measurements, and in addition, it uses automatic derivation to solve the NLS problem. Due to these two aspects, the system is not able to run in real time.

In this article, we propose a continuous-time based method that supports fusing measurements from LiDAR, IMU, and camera, which is very easy to expand to fuse more other sensors to improve the accuracy of localization and mapping, such as fusing GPS in outdoor cases and RFID [39] in indoor cases.

III. Preliminary on Continuous-Time Trajectory

This section presents in detail continuous-time trajectory representation based on B-spline and its time derivatives.

A. B-Spline Trajectory Representation

We employ B-spline to parameterize continuous-time trajectory as it has the good property of locality and closed-form analytic time derivatives, C^{k-1} smoothness for a spline of order k (degree $k-1$) [40]. Specifically, we parameterize the continuous-time 6-DoF trajectory with uniform cumulative B-splines in a split representation format [41]. The translation $\mathbf{p}(t)$ of k order over time $t \in [t_i, t_{i+1})$ is controlled by the temporally uniformly distributed translational control points $\mathbf{p}_i, \mathbf{p}_{i+1}, \ldots, \mathbf{p}_{i+k-1}$, and the matrix format [42] could be written as

$$\underbrace{\mathbf{p}(t)}_{3\times 1} = \underbrace{\begin{bmatrix}\mathbf{p}_i & \mathbf{d}_1^i & \cdots & \mathbf{d}_{k-1}^i\end{bmatrix}}_{3\times k}\underbrace{\widetilde{\mathbf{M}}^{(k)}}_{k\times k}\underbrace{\mathbf{u}}_{k\times 1}$$
$$\mathbf{u} = \begin{bmatrix}1 & u & \cdots & u^{k-1}\end{bmatrix}, u = (t-t_i)/(t_{i+1}-t_i) \quad (1)$$

with difference vectors $\mathbf{d}_j^i = \mathbf{p}_{i+j} - \mathbf{p}_{i+j-1} \in \mathbb{R}^3$. The cumulative spline matrix $\widetilde{\mathbf{M}}^{(k)}$ of uniform B-spline only depends on the B-spline order. We further define $\boldsymbol{\lambda}(t) = \widetilde{\mathbf{M}}^{(k)}\mathbf{u}$; thus, (1) can be written as

$$\mathbf{p}(t) = \mathbf{p}_i + \sum_{j=1}^{k-1} \lambda_j(t) \cdot \mathbf{d}_j^i \,. \quad (2)$$

To parameterize the 3-D rotation in $SO(3)$, we adopt the cumulative B-splines in Lie groups with the following expression over time $t \in [t_i, t_{i+1})$:

$$\mathbf{R}(t) = \mathbf{R}_i \cdot \prod_{j=1}^{k-1} \mathrm{Exp}\left(\lambda_j(t) \cdot \mathrm{Log}\left(\mathbf{R}_{i+j-1}^{-1}\mathbf{R}_{i+j}\right)\right) \quad (3)$$

where $\mathbf{R}_i \in SO(3)$ are the control points for rotation. The difference vector between two rotations is defined as $\mathbf{d}_j^i = \mathrm{Log}(\mathbf{R}_{i+j-1}^{-1}\mathbf{R}_{i+j}) \in \mathbb{R}^3$ and $\mathbf{A}_j(t) = \mathrm{Exp}(\lambda_j(t) \cdot \mathbf{d}_j)$ where omitting the i to simplify notation, (3) can be written in the following concise equation:

$$\mathbf{R}(t) = \mathbf{R}_i \cdot \prod_{j=1}^{k-1} \mathbf{A}_j(t) \,. \quad (4)$$

In this article, we select cubic (degree $= 3$) B-spline, and the corresponding cumulative spline matrix $\widetilde{\mathbf{M}}^{(k)}$ is

$$\widetilde{\mathbf{M}}^{(4)} = \frac{1}{6}\begin{bmatrix}6 & 5 & 1 & 0\\ 0 & 3 & 3 & 0\\ 0 & -3 & 3 & 0\\ 0 & 1 & -2 & 1\end{bmatrix}. \quad (5)$$

B. Time Derivatives of B-Spline

As mentioned before, B-spline provides closed-form analytic derivatives, enabling the proposed system to fuse high-frequency IMU measurements seamlessly. Continuous-time trajectory of IMU in global frame $\{G\}$ is denoted as ${}^G_I\mathbf{T}(t) = [{}^G_I\mathbf{R}(t), {}^G\mathbf{p}_I(t)]$, and its time derivatives can be derived as

$${}^G\mathbf{v}(t) = {}^G\dot{\mathbf{p}}_I(t) = \sum_{j=1}^{3} \dot{\lambda}_j(t) \cdot \mathbf{d}_j^i, \quad (6)$$

$${}^G\mathbf{a}(t) = {}^G\ddot{\mathbf{p}}_I(t) = \sum_{j=1}^{3} \ddot{\lambda}_j(t) \cdot \mathbf{d}_j^i, \quad (7)$$

$${}^G_I\dot{\mathbf{R}}(t) = \mathbf{R}_i\left(\dot{\mathbf{A}}_1\mathbf{A}_2\mathbf{A}_3 + \mathbf{A}_1\dot{\mathbf{A}}_2\mathbf{A}_3 + \mathbf{A}_1\mathbf{A}_2\dot{\mathbf{A}}_3\right) \quad (8)$$

where $\dot{\mathbf{A}}_j = \mathrm{Exp}(\dot{\lambda}_j(t) \cdot \mathbf{d}_j)$. Naturally, it is straightforward to compute the linear accelerations and angular velocities in local IMU frame

$${}^I\mathbf{a}(t) = {}^G_I\mathbf{R}^\top(t)\left({}^G\mathbf{a}(t) - {}^G\mathbf{g}\right) \quad (9)$$

$${}^I\boldsymbol{\omega}(t) = {}^G_I\mathbf{R}^\top(t) \cdot {}^G_I\dot{\mathbf{R}}(t) \quad (10)$$

where ${}^G\mathbf{g} \in \mathbb{R}^3$ denotes the gravity vector in global frame.

In the following section, we model the continuous-time trajectory of IMU sensor in $\{G\}$ frame, termed as ${}^G_I\mathbf{T}(t)$. The global frame $\{G\}$ is determined by the first IMU measurement after system initialization by aligning its z-axis with the gravity direction, and the gravity can be denoted as [0,0,9.8] in $\{G\}$. We assume the extrinsic rigid transformations between sensors are precalibrated [43], and we can get the camera trajectory ${}^G_C\mathbf{T}(t)$, and LiDAR trajectory ${}^G_L\mathbf{T}(t)$ handily by transferring IMU trajectory ${}^G_I\mathbf{T}(t)$ with the known extrinsic transformations.

TABLE II
NOTATIONS GLOSSARY

Symbols	Meaning
$\boldsymbol{\Phi}(t_{\kappa-1}, t_\kappa)$	Control points of B-splines in $[t_{\kappa-1}, t_\kappa)$
$\boldsymbol{\Phi}_R(t_\kappa)/\boldsymbol{\Phi}_p(t_\kappa)$	Involved orientation/position control points at t_κ
$\mathbf{b}_\omega^\kappa, \mathbf{b}_a^\kappa$	Biases of temporal sliding window in $[t_{\kappa-1}, t_\kappa)$
t_L, t_I, t_C	Timeoffsets of LiDAR, IMU and camera, respectively
t	Timestamp of trajectory or sensor measurements
$\tau = t + t_{\text{offset}}$	Corrected timestamp of sensor measurements

IV. CONTINUOUS-TIME FIXED-LAG SMOOTHING

This section presents the continuous-time fixed-lag smoothing applied in factor-graph optimization, which is the core of estimator design. Before diving into details, we introduce some notations used in this article, which is summarized in Table II.

A. Factor-Graph Optimization

We fuse heterogeneous IMU, LiDAR, and camera measurements in a factor graph optimization framework with the continuous-time trajectory formulation. Fig. 2(a) and (b) shows the factor graphs of the LI system and LIC system, respectively. Here, we only illustrate the factor graph for LIC system, and it is straightforward to apply to LI system with minor adaptations. Specifically, our estimator in the temporal sliding window (detailed in Section V-A2) over $[t_{\kappa-1}, t_\kappa)$ aims to estimate the following states:

$$\mathcal{X}^\kappa = \{\mathbf{x}_R^\kappa, \mathbf{x}_p^\kappa, \mathbf{x}_{I_b}^\kappa, \mathbf{x}_\lambda^\kappa, t_I, t_C\},$$

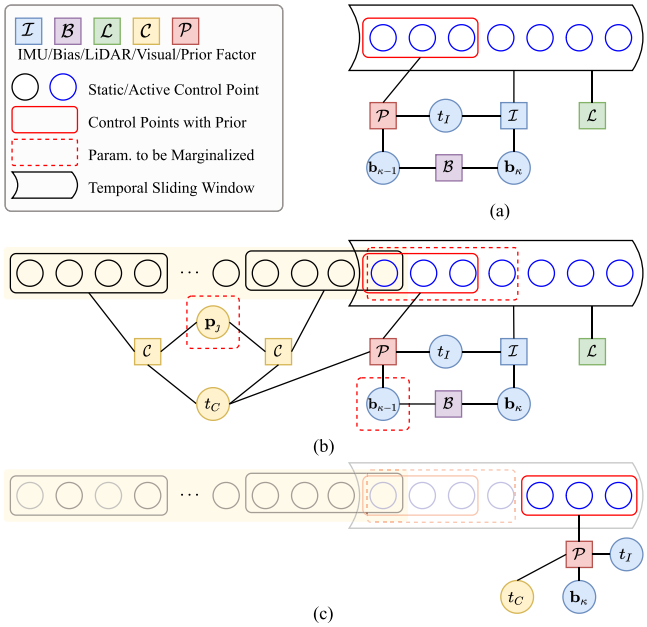


Fig. 2. Factor graphs of multi-sensor fusion. (a) A typical factor graph of LI system fusion. (b) A typical factor graph of LIC system fusion. Active control points are to be optimized, while static control points remain constant. The control points with yellow background are involved in visual keyframe sliding window. (c) After marginalization of *(b)*, the induced prior factor is involved with the latest control points, latest bias and timeoffsets.

$$\mathbf{x}_{I_b}^{\kappa} = \{\mathbf{b}_{\omega}^{\kappa-1},\ \mathbf{b}_{a}^{\kappa-1},\ \mathbf{b}_{\omega}^{\kappa},\ \mathbf{b}_{a}^{\kappa}\} \tag{11}$$

which include active control points of B-splines $\mathbf{\Phi}(t_{\kappa-1}, t_{\kappa}) = \{\mathbf{x}_R^{\kappa}, \mathbf{x}_p^{\kappa}\}$, the IMU biases $\mathbf{x}_{I_b}^{\kappa}$, the parameters of visual landmarks $\mathbf{x}_{\lambda}^{\kappa}$, and the temporal offset between LiDAR and IMU t_I or camera t_C. Notably, LiDAR is taken as the base sensor in our multisensor fusion system, thus we need to align IMU and camera timestamps to LiDAR. Section V-C2 provides more details about the timeoffset estimation.

The factor graph needed to be solved consists of LiDAR factors $\mathbf{r}_L$, IMU factors $\mathbf{r}_I$, one bias factor $\mathbf{r}_{I_b}$, visual factors $\mathbf{r}_C$, and one prior factor $\mathbf{r}_{\text{prior}}$ induced from marginalization; the details of factors are provided in the following sections. With the LiDAR-inertial measurements during $[t_{\kappa-1}, t_{\kappa})$ and tracked visual features in the keyframe sliding window (detailed in Section V-B2), we formulate the following nonlinear least-squares (NLS) problem:

$$\hat{\mathcal{X}}^{\kappa} = \underset{\mathcal{X}^{\kappa}}{\operatorname{argmin}}\ \mathbf{r}, \quad \mathbf{r} = \mathbf{r}_I + \mathbf{r}_{I_b} + \mathbf{r}_L + \mathbf{r}_c + \mathbf{r}_{\text{prior}} \tag{12}$$

and solve the NLS problem by iterative optimization methods. At each iteration, the system is linearized at current estimate $\hat{\mathbf{x}}$, and we define its error state as $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$, where $\mathbf{x}$ is the true state. In practice, we adopt the Levenberg–Marquardt algorithm from the Ceres Solver [44] library and employ analytical derivatives to speed up the NLS problem-solving.

B. LiDAR Factor

A LiDAR point measurement ${}^{L}\mathbf{p}_t$ with noise $\mathbf{n}_L$, measured at time t_ℓ, is associated with a 3-D plane in closest point parameterization [14], [45], ${}^{G}\boldsymbol{\pi} = {}^{G}d_{\pi}\,{}^{G}\mathbf{n}_{\pi}$, where ${}^{G}d_{\pi}$ and ${}^{G}\mathbf{n}_{\pi}$ denote the distance of the plane to origin and unit normal vector, respectively. We can transform LiDAR point to global frame by

$${}^{G}\hat{\mathbf{p}}_{\ell} = {}^{G}_{L}\mathbf{R}(\tau_\ell)\left({}^{L}\mathbf{p}_{\ell} + \mathbf{n}_L\right) + {}^{G}\mathbf{p}_L(\tau_\ell) \tag{13}$$

and the point-to-plane distance is given by

$$\begin{aligned}\mathbf{r}_L(\tau_\ell, \hat{\mathcal{X}}^{\kappa}, {}^{L}\mathbf{p}_{\ell}, {}^{G}\boldsymbol{\pi}) &= {}^{G}\mathbf{n}_{\pi}^{\top}\,{}^{G}\hat{\mathbf{p}}_{\ell} + {}^{G}d_{\pi} \\ &\approx \mathbf{H}_{\ell}\cdot\tilde{\mathbf{x}} + \mathbf{G}_{\ell}\cdot\mathbf{n}_L\end{aligned} \tag{14}$$

where $\tau_\ell = t_\ell$ is the measure time of LiDAR point; $\mathbf{H}_\ell$ is Jacobian matrix w.r.t. error state $\tilde{\mathbf{x}}$ (see our supplementary file for detail). $\mathbf{n}_L$ is assumed to be under independent and identically distributed (i.i.d.) white Gaussian noise in our experiments. $\mathbf{G}_\ell$ is the Jacobian with respect to $\mathbf{n}_L$, which can be easily computed.

C. IMU Factor and Bias Factor

Considering raw IMU measurements at t_m with angular velocity ${}^{I}\boldsymbol{\omega}_m$ and linear acceleration ${}^{I}\mathbf{a}_m$, and the true angular velocity and linear acceleration are denoted by ${}^{I}\boldsymbol{\omega}$ and ${}^{I}\mathbf{a}$, respectively. The following equations hold:

$${}^{I}\boldsymbol{\omega}_m = {}^{I}\boldsymbol{\omega}(t) + \mathbf{b}_{\omega}(t) + \mathbf{n}_{\omega} \tag{15}$$

$${}^{I}\mathbf{a}_m(t) = {}^{G}_{I}\mathbf{R}^{\top}(t)\left({}^{G}\mathbf{a}(t) - {}^{G}\mathbf{g}\right) + \mathbf{b}_a(t) + \mathbf{n}_a \tag{16}$$

$$\dot{\mathbf{b}}_{\omega}(t) = \mathbf{n}_{b_\omega}, \quad \dot{\mathbf{b}}_a(t) = \mathbf{n}_{b_a} \tag{17}$$

where $\mathbf{n}_{\omega}$, $\mathbf{n}_a$ are zero-mean Gaussian white noise. The gyroscope bias $\mathbf{b}_{\omega}$ and accelerometer bias $\mathbf{b}_a$ are modeled as random walks, driving by the white Gaussian noises $\mathbf{n}_{b_\omega}$ and $\mathbf{n}_{b_a}$, respectively. With the time offset t_I of IMU between LiDAR, we have the following IMU factor:

$$\begin{aligned}&\mathbf{r}_I(\tau_m, \hat{\mathcal{X}}^{\kappa}, {}^{I}\boldsymbol{\omega}_m, {}^{I}\mathbf{a}_m) \\ &= \begin{bmatrix} {}^{I}\boldsymbol{\omega}(\tau_m) - {}^{I}\boldsymbol{\omega}_m + \mathbf{b}_{\omega}^{\kappa} \\ {}^{I}\mathbf{a}(\tau_m) - {}^{I}\mathbf{a}_m + \mathbf{b}_{a}^{\kappa} \end{bmatrix} + \begin{bmatrix} \mathbf{n}_{\omega} \\ \mathbf{n}_{a} \end{bmatrix} \\ &\approx \mathbf{H}_{I_m}^{\kappa}\cdot\tilde{\mathbf{x}} + \mathbf{G}_{I_m}\cdot\mathbf{n}_I\end{aligned} \tag{18}$$

and bias factor

$$\begin{aligned}\mathbf{r}_{I_b}(\hat{\mathcal{X}}^{\kappa}) &= \begin{bmatrix} \mathbf{b}_{\omega}^{\kappa} - \mathbf{b}_{\omega}^{\kappa-1} \\ \mathbf{b}_{a}^{\kappa} - \mathbf{b}_{a}^{\kappa-1} \end{bmatrix} + \begin{bmatrix} \mathbf{n}_{b_\omega} \\ \mathbf{n}_{b_a} \end{bmatrix} \\ &\approx \mathbf{H}_{I_b}^{\kappa}\cdot\tilde{\mathbf{x}} + \mathbf{G}_{I_b}\cdot\mathbf{n}_{I_b}\end{aligned} \tag{19}$$

where $\tau_m = t_m + t_I$ is the corrected IMU timestamp, and $\mathbf{H}_{I_m}^{\kappa}$ and $\mathbf{H}_{I_b}^{\kappa}$ are Jacobian matrices with respect to states (see supplementary file), and $\mathbf{G}_{I_m}$ and $\mathbf{G}_{I_b}$ are Jacobian matrices with respect to noise. By substituting the derivative of continuous-time trajectory (9) at time instant τ_m into (18), we can optimize the continuous-time trajectory by raw IMU measurements directly, avoiding the efforts of IMU propagation or preintegration.

D. Visual Factor

A landmark $\mathbf{p}_j$ observed in its anchor keyframe $\mathcal{F}_a$ at timestamp t_a and observed again in frame $\mathcal{F}_b$ at timestamp t_b, can be

given the initialized inverse depth as λ_j through triangulation (as Section V-B1). Let $\boldsymbol{\rho}_j^a$ denote 2-D raw observation in $\mathcal{F}_a$ with noise $\mathbf{n}_c$, the estimated position of landmark in frame $\mathcal{F}_b$ is

$$\hat{\mathbf{p}}_j^b = {}_C^G\mathbf{T}(\tau_b)^\top \cdot {}_C^G\mathbf{T}(\tau_a) \cdot \frac{1}{\lambda_j}\pi_c(\boldsymbol{\rho}_j^a + \mathbf{n}_c) \tag{20}$$

where $\pi_c(\cdot)$ denotes the back projection, which transforms a pixel to the normalized image plane. τ_a, τ_b are corrected timestamps to remove timeoffsets. The corresponding visual factor based on reprojection error is defined as

$$\begin{aligned}\mathbf{r}_c(\tau_b, \hat{\mathcal{X}^\kappa}, \boldsymbol{\rho}_j^b) &= \begin{bmatrix}\mathbf{e}_1^\top \\ \mathbf{e}_2^\top\end{bmatrix}\left(\frac{\hat{\mathbf{p}}_j^b}{\mathbf{e}_3^\top \hat{\mathbf{p}}_j^b} - \pi_c\left(\boldsymbol{\rho}_j^b + \mathbf{n}_c\right)\right) \\ &\approx \mathbf{H}_j^b \cdot \tilde{\mathbf{x}} + \mathbf{G}_j^b \cdot \mathbf{n}_c\end{aligned} \tag{21}$$

where $\mathbf{e}_i$ denotes a 3×1 vector with its ith element to be 1 and the others to be 0, and $\boldsymbol{\rho}_j^b$ describes 2-D raw observation in $\mathcal{F}_b$. $\mathbf{H}_j^b$ denotes Jacobian matrix (see supplementary file).

E. Marginalization

To bound the size of sliding window in our continuous-time fixed-lag smoother, marginalization has to be resorted to. Although marginalization is universally leveraged in discrete-time sliding-window estimator [3], [23], it is rarely investigated in continuous-time sliding-window estimator. By utilizing probabilistic marginalization, information on the active states can be well reserved in the estimator, when measurements and old states are removed from the sliding window. Linearizing the factors at the best estimate of the state gives

$$\begin{bmatrix}\mathbf{H}_{\alpha\alpha} & \mathbf{H}_{\alpha\beta} \\ \mathbf{H}_{\beta\alpha} & \mathbf{H}_{\beta\beta}\end{bmatrix}\begin{bmatrix}\mathbf{x}_\alpha \\ \mathbf{x}_\beta\end{bmatrix} = \begin{bmatrix}\mathbf{b}_\alpha \\ \mathbf{b}_\beta\end{bmatrix} \tag{22}$$

where we organize the reserved parameters in $\mathbf{x}_\alpha$, and $\mathbf{x}_\beta$ will be marginalized. Using the Schur complement [46], the following equation holds:

$$\left(\mathbf{H}_{\alpha\alpha} - \mathbf{H}_{\alpha\beta}\mathbf{H}_{\beta\beta}^{-1}\mathbf{H}_{\beta\alpha}\right)\mathbf{x}_\alpha = \left(\mathbf{x}_\alpha - \mathbf{H}_{\alpha\beta}\mathbf{H}_{\beta\beta}^{-1}\mathbf{x}_\beta\right) \tag{23}$$

and by introducing new notations, we can denote the above equation by

$$\hat{\mathbf{H}}_{\alpha\alpha}\mathbf{x}_\alpha = \hat{\mathbf{b}}_\alpha \tag{24}$$

which is exactly the prior factor. Fig. 2(b) displays the entries needed to be marginalized out of the LI temporal- and visual keyframe-sliding windows. The entries needed to be marginalized vary at different statuses. We marginalize out the control points and IMU biases when sliding the LI temporal window, and marginalize out the inverse depths of landmarks when sliding the oldest keyframe out of visual keyframe sliding window, producing a prior on

$$\mathcal{X}_{\text{prior}}^\kappa = \{\boldsymbol{\Phi}(t_{\kappa-1}, t_\kappa) \cap \boldsymbol{\Phi}(t_\kappa, t_{\kappa+1}),\ \mathbf{b}_\omega^\kappa, \mathbf{b}_a^\kappa, t_I, t_C\}.$$

where $\boldsymbol{\Phi}(t_{\kappa-1}, t_\kappa) \cap \boldsymbol{\Phi}(t_\kappa, t_{\kappa+1})$ denotes the affected control points in next sliding window. The prior factor induced from marginalization is shown in Fig. 2(c) and will be involved in future factor-graph optimization.

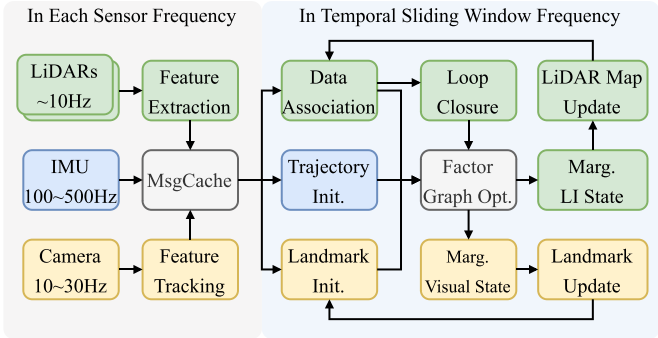


Fig. 3. Pipeline of the proposed LIC fusion system. Raw IMU measurements, features of each LiDAR and tracked features of camera are cached in the MsgCache module, and measurements are fed to the sliding window at $\frac{1}{\eta\Delta t}$ Hz. After factor-graph optimization, we separately marginalize LI state and visual state, and update local LiDAR map and visual landmarks. More details are provided in Section V.

V. Multisensor Fusion

This section presents in detail the multisensor fusion method powered by the versatile continuous-time fixed-lag smoothing presented above. Fig. 3 shows the overall architecture of an LIC fusion system. In this section, we first introduce the LiDAR-IMU system (Section V-A) and visual system (Section V-B). Then, we reveal some implementation details (Section V-C) to accommodate the particularity of continuous-time trajectory estimation, which is with significant distinction from discrete-time methods.

A. LiDAR-Inertial System

1) LiDAR Measurement Processing: There are three main stages for LiDAR point cloud processing: feature extraction, data association, and local map management. Inspired by [2], for each new incoming LiDAR scan, we first compute the curvature of each point according to its neighboring points, and select points with small curvature as planar points. The motion distortion in raw LiDAR scan is inevitable due to the motion when the LiDAR is scanning. Since we have formulated the continuous-time trajectory, it is handy to query poses at every time instant, which allows us to compensate for the motion distortion by aligning all the LiDAR points in one scan to the sweeping start time of that scan. After removing the incidental motion distortion in raw LiDAR scan and projecting undistorted LiDAR scan to the map frame using the initialized (or optimized) continuous-time trajectory, we can perform the point-to-plane data association by associating planar LiDAR points to tiny planes in the map. The tiny planes are found by fitting neighboring planar points on the map. The point-to-plane distance [see (14)] will be minimized in the continuous-time fixed-lag smoother. After finishing the first optimization, we update the data association using the optimized trajectory and optimize again to refine the estimation. Upon completion of the optimization, we individually transform each LiDAR point into the scan's start time according to queried pose from the optimized trajectory, and select keyscans based on the displacement of poses or time span. Keyscans are leveraged to build the local LiDAR map in $\{G\}$ frame, and the local LiDAR

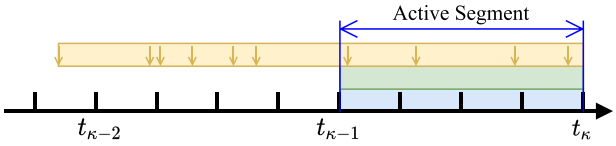


Fig. 4. Involved measurements from IMU (blue block) and LiDAR (green block) sensors in a temporal sliding window in $[t_{\kappa-1}, t_\kappa)$, and measurements of camera (yellow block, keyframes denoted by arrows) in a keyframe sliding window. The active trajectory segment of which control points will be optimized is within the LI temporal sliding window.

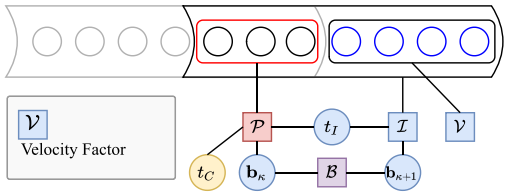


Fig. 5. Factor graph of trajectory initialization with details provided in Section V-C1.

map composed of certain keyscans keeps updated upon newly added scan.

2) LI Temporal Sliding Window: For the LI system, we maintain a temporal sliding window within a constant time duration, $\eta\Delta t$, where Δt denotes the B-spline temporal knot distance and η is an integer. The continuous-time trajectory of LI system is optimized and updated every $\eta\Delta t$ seconds. Compared to optimizing the trajectory within every LiDAR scan individually [17], the temporal sliding-window optimization (fixed-lag smoothing) makes full use of all measurements within the sliding window (see Fig. 4) and prior information from marginalization, which could achieve higher accuracy. Note that the IMU bias sampling frequency is set as $\frac{1}{\eta\Delta t}$Hz, that is to say, we add a new IMU bias state when sliding the temporal window forward, and the discrete noise of bias factor [see (19)] is determined accordingly.

B. Visual System

The main purpose of incorporating camera to the multisensor fusion estimator is to improve the robustness of LI system. We expect the LIC system not to have degraded performance in structureless scenarios, which is a significant challenge for LI systems.

1) Visual Front End: We extract corner features [47] from the images with KLT tracking [48] and determine image keyframe based on parallax variation and the number of features tracked, similar to [3]. Furthermore, we triangulate the landmarks tracked throughout the whole keyframe sliding window (see Section V-B2) using keyframe poses queried from current best-estimated continuous-time trajectory, and only keep the landmarks with low reprojection errors. Landmarks are parameterized by inverse depth represented in anchor frame, which in our case is always the oldest keyframe in the keyframe sliding window.

2) Visual Keyframe Sliding Window: When processing images, we maintain a visual keyframe sliding window with a constant number of keyframes, in contrast to the constant time duration for LI temporal sliding window. This is due to the consideration that visual landmark triangulation needs observations across multiple keyframes with sufficient parallaxes. The duration of the keyframe sliding window is normally longer than the LI temporal sliding window, and estimating the entire trajectory covered by all the keyframes is time-consuming. In practice, we determine the trajectory optimization range based on the temporal sliding window of the LI system and define the control points to be optimized as active control points (shown in Fig. 4). We only optimize the control points of active trajectory segment within the LI temporal sliding window, while the other control points are involved in the optimization but kept static in the optimization. That formulation of visual system may not be the best choice of high-accuracy estimation, but rather a tradeoff to achieve real-time. While for the marginalization of visual keyframe sliding window, the oldest keyframe and the landmarks in it are marginalized out.

The strategy of sliding the keyframe window is similar to [3], where the newest frame in the sliding window is always the latest image of the system. If the second newest frame is determined as keyframe, we will marginalize out the oldest keyframe and landmarks in it from the sliding window, in order to keep a constant number of keyframes. Otherwise, we discard the second newest frame directly. Note that, we need to transfer the anchor frame of visual landmark if its anchor frame is marginalized.

C. Extra Implementation Details

1) Initialization: We initialize the IMU bias and gravity direction using the raw IMU measurements, assuming that the system is stationary at the start, which shares the same spirit as [49]. The system appends new IMU biases and control points when sliding the LI temporal window. The new IMU biases are initialized to the value of the previous temporal sliding window bias, and the new control points are first assigned values of the neighboring control point and further initialized via factor graph optimization as shown in Fig. 5. The velocity factor is defined as

$$\mathbf{r}_v = {}^G\hat{\mathbf{v}}_t - {}^G\mathbf{v}(t) \tag{25}$$

where ${}^G\mathbf{v}(t)$ is defined in (7), which is derivative of continuous-time trajectory. ${}^G\hat{\mathbf{v}}_t$ is an estimate of global linear velocity at the end of current LI temporal window, and is derived by forward integrating IMU measurements from the pose at the start of temporal window. When solving the initialization problem, only the newly added control points are optimized while all the other states remain constant during optimization. After initialization, we remove the distortion of LiDAR scans with that initialized trajectory for better association results.

2) Online Calibration of Timeoffset: Estimating timeoffsets between sensors is natural and convenient when modeling the trajectory in continuous time. In this article, we choose the LiDAR sensor as the time baseline. Thus, the timeoffsets of the camera and IMU with respect to the LiDAR need to be known

or calibrated online. The main reason of choosing LiDAR as the base sensor is that local LiDAR map needs to be maintained, and once the timeoffset of LiDAR trajectory is changed, LiDAR map needs to be transformed accordingly, which is time-consuming. In contrast, the extra computation arising from the change of camera or IMU timestamps is insignificant.

3) Loop Closure: We utilize Euclidean distance-based loop closure detection method [50] and adopt the two-stage continuous-time trajectory correction method [17] to tackle loop closures. After a loop closure optimization, we will remove the prior information of states since the current best-estimated states may be away from the linearized points, resulting in inappropriate prior constraints.

VI. Experiment

In the experiments, we evaluate the proposed continuous-time fixed-lag smoother in terms of pose estimation accuracy in various scenarios, systematically analyze the convergence speed and the accuracy of online timeoffset calibration, and disclose the runtime of the main stages in the proposed method. We assess a variety of sensor combinations powered by the continuous-time fixed-lag smoothing method, including the following:

1) *CLIO:* one LIDAR and one IMU;
2) *CLIO2:* two LiDARs and one IMU;
3) *CLIC:* one LiDAR, one IMU and one camera;
4) *CLIC2:* two LiDAR, one IMU and one camera.

We adopt the absolute pose error (APE) as evaluation metric to compare the proposed method against three LI systems (CLINS [17], LIO-SAM [50]), and three LIC systems (LVI-SAM [6], VIRAL-SLAM [51], LIC-Fusion 2.0 [10]) and two multi-LiADR systems (VIRAL-SLAM, MILIOM [37]). In all experiments, the temporal knot distance Δt is 0.03 s, and the duration of LI temporal sliding window length is 0.12 s while the visual keyframe sliding window size is 10.

A. Evaluation Datasets

We evaluate the following three publicly available datasets and our self-collected datasets.

1) *VIRAL [52]:* The Visual-Inertial-RAnging-Lidar Dataset contains a variety of sensors, including two 16-beam Ouster LiDARs at 10 Hz, two monocular cameras at 10 Hz, and a VectorNav VN100 IMU at 385 Hz, etc. The dataset comprises nine sequences collected indoors and outdoors by an MAV (Micro Aerial Vehicle).
2) *NCD [53]:* The Newer College Dataset is collected using a handheld device that consists of a 64-beam Ouster LiDAR at 10 Hz with its internal IMU at 100 Hz, and a stereo camera at 30 Hz. The dataset combines built environments, open spaces, and vegetated areas.
3) *LVI-SAM [6]:* The LVI-SAM Dataset features both handheld and vehicle (Jackal) platforms in outdoor open vegetated environments including geometrically degenerate surroundings with 16-beam LiDAR at 10 Hz, camera at 20 Hz, and IMU at 500 Hz.
4) *YQ:* The self-collected YuQuan Dataset collected on our university campus consists of seven sequences using an electric car with a sensor rig mounted on the top shown in Fig. 6. The sensor rig comprises a 16-beam LiDAR at 10 Hz, a camera at 20 Hz, and an IMU at 400 Hz. The trajectories of different sequences overlaid on Google map are shown in Fig. 7. GPS measurements are collected to provide groundtruth for this outdoor dataset.
5) *Vicon Room:* The self-collected Vicon Room Dataset shares the identical sensor rig as YQ dataset. This indoor dataset is collected with hand-held random motion, and a motion capture system is leveraged to provide groundtruth trajectories.

Fig. 6. Sensor rig and ground vehicle to collect YQ dataset.

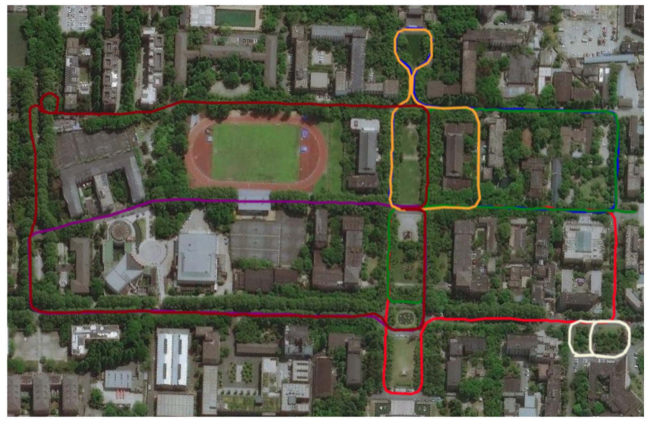


Fig. 7. Seven trajectories of YQ dataset.

B. LiDAR-Inertial Fusion: CLIO

In this experiment, we evaluate the accuracy of continuous-time trajectory estimation of the CLIO system enabled by the proposed continuous-time fixed-lag smoothing. The results on VIRAL dataset are shown in Table III; CLIO outperforms other LI methods in most sequences and achieves an average RMSE of 0.034 m. CLINS is much more accurate than all the discrete-time methods, including LIO-SAM [50], MILIOM [37], and VIRAL [51]. Compared to continuous-time method, CLINS [17], which only optimizes control points of trajectory within the time span of the newest LiDAR scan and lacks probabilistic marginalization, the proposed CLIO, optimizing all the control points in a sliding window involved with

TABLE III
APE (RMSE, METER) RESULTS ON VIRAL DATASET

Method	Sensor[1]	eee_01 (237m)	eee_02 (171m)	eee_03 (128m)	nya_01 (160m)	nya_02 (249m)	nya_03 (315m)	sbs_01 (202m)	sbs_02 (184m)	sbs_03 (199m)	average
LIO-SAM[2] [50]	L, I	0.075	0.069	0.101	0.076	0.090	0.137	0.089	0.083	0.140	0.096
MILIOM (horz. LiDAR)[2] [37]	L, I	0.104	0.065	0.063	0.083	0.072	0.058	0.076	0.081	0.088	0.077
VIRAL (horz. LiDAR)[2] [51]	L, I	0.064	0.051	0.060	0.063	0.042	0.039	0.051	0.056	0.060	0.054
CLINS (w/o loop) [17]	L, I	0.059	0.030	0.029	**0.034**	**0.040**	**0.039**	0.029	0.031	0.033	0.036
CLIO (w/o loop)	L, I	**0.030**	**0.023**	0.028	0.042	0.053	0.042	**0.028**	0.032	**0.030**	**0.034**
CLIC (w/o loop)	L, I, C	0.030	0.029	**0.028**	0.040	0.054	0.041	0.029	**0.031**	0.033	0.035
MILIOM (2 LiDARs)[2] [37]	L2, I	0.067	0.066	0.052	0.057	0.067	0.042	0.066	0.082	0.093	0.066
VIRAL (2 LiDARs)[2] [51]	L2, I, C	0.060	0.058	0.037	0.051	0.043	**0.032**	0.048	0.062	0.054	0.049
CLIO2 (w/o loop)	L2, I	0.040	**0.021**	0.031	0.030	0.037	0.034	**0.033**	0.037	0.044	0.034
CLIC2 (w/o loop)	L2, I, C	**0.038**	0.025	**0.030**	**0.029**	**0.036**	0.035	0.034	**0.035**	**0.043**	**0.034**

[1] Sensors L,I,C are abbreviations of LiDAR, IMU, camera, respectively. L2 represents using two LiDAR sensors.
[2] Results are from [51]. The horz. LiDAR in table means only the horizontal LiDAR is used for odometry.

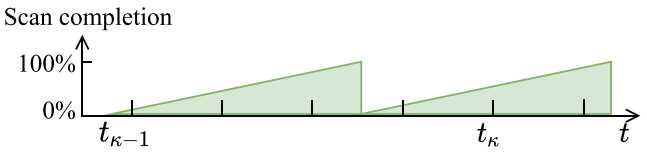


Fig. 8. LiDAR-inertial temporal sliding window over $[t_{\kappa-1}, t_\kappa)$ consists of multiple LiDAR scans (green blocks), which can be an incomplete scan.

TABLE IV
APE (RMSE, METER) RESULTS ON NCD DATASET

Method	NCD_01 (1530s / 1609m)	NCD_02 (2656s / 3063m)	NCD_06 (120s / 97m)
LIO-SAM (w/o loop)	1.660	**2.305**	0.272
CLIO (w/o loop)	**0.792**	2.686	**0.091**
LIO-SAM (w/ loop)	0.544	0.592	0.272
CLIO (w/ loop)	**0.408**	**0.381**	**0.091**

CThe best result is in bold.

TABLE V
THE APE (RMSE, METER) RESULTS ON LVI-SAM DATASET

Method	Sensor	Handheld (1642s)	Jackal (2182s)
LIO-SAM (w/o loop)	L, I	53.62	3.54
CLIO (w/o loop)	L, I	fail	3.43
LVI-SAM (w/o loop)	L, I, C	7.87	4.05
CLIC (w/o loop)	L, I, C	**2.56**	**2.55**
LIO-SAM (w/ loop)	L, I	fail	1.52
CLIO (w/ loop)	L, I	fail	1.01
LVI-SAM (w/ loop)	L, I, C	0.83	**0.67**
CLIC (w/ loop)	L, I, C	0.65	0.88
CLIC (w/ loop, w/ calib)	L, I, C	**0.56**	0.84

The best result is in bold.

several LiDAR scans (see Figs. 4 and 8) and benefiting from marginalization, can achieve higher accuracy.

We further conduct evaluations in large-scale scenarios on the LVI-SAM dataset and NCD dataset (Tables V and IV); CLIO achieves higher accuracy on most sequences compared to LIO-SAM. Especially, it is worth noting that on the NCD_06 sequence with the handheld sensor shaking vigorously, CLIO shows significant superiority over the discrete-time LIO-SAM. The improvement in accuracy could be attributed to our continuous-time trajectory formulation as it adequately

TABLE VI
APE (RMSE, METER) RESULTS ON YQ DATASET (OUTDOOR)

Sequence	LVI-SAM (w/o loop)	LIC-Fusion 2.0 (w/o loop)	CLIC (w/o loop)	LVI-SAM (w/ loop)	CLIC (w/ loop)
YQ-01 (1005m)	1.614	3.300	1.826	**1.227**	1.537
YQ-02 (1021m)	1.790	1.804	1.626	1.610	**1.363**
YQ-03 (1058m)	3.017	2.798	2.616	1.886	**1.701**
YQ-04 (1233m)	3.163	2.697	2.450	2.402	**1.917**
YQ-05 (673m)	1.721	1.550	1.537	8.465	**1.439**
YQ-06 (1644m)	3.753	3.862	3.082	3.682	**1.607**
YQ-07 (414m)	0.800	1.306	0.761	0.795	**0.761**

The best result of LIC system without (or with) loop closure is underlined (or in bold).

TABLE VII
APE (RMSE, METER) RESULTS ON VICON ROOM DATASET (INDOOR)

Seq.	LVI-SAM (w/o loop)	LIC-Fusion 2.0 (w/o loop)	CLIC (w/o loop)
Seq1 (43m)	0.393	**0.033**	0.080
Seq2 (84m)	0.232	0.096	**0.073**
Seq3 (34m)	0.364	**0.052**	0.073
Seq4 (53m)	0.395	0.092	**0.089**
Seq5 (50m)	0.155	**0.044**	0.171
Seq6 (88m)	0.459	**0.046**	0.128

The best result is in bold.

addresses the motion distortion of LiDAR point cloud, which is of significant importance in highly dynamic motion scenarios.

The presented continuous-time fixed-lag smoothing method supports fusion of multiple sensors conveniently. In Table III, we also showcase the pose estimation accuracy of a system with the fusion of IMU and two LiDARs, dubbed CLIO2. We can see that CLIO2 has on-par accuracy with CLIO on most of the sequences, and shows more accurate pose estimation over the outdoor sequences (nya_xx).

C. LiDAR-Inertial-Camera Fusion: CLIC

In this section, we discuss the accuracy of the LiDAR-Inertial-Camera pose estimation system enabled by continuous-time fixed-lag smoothing. The experimental results on the VIRAL and LVI datasets are summarized in Tables III and V. In addition, the evaluation results on our self-collected indoor and outdoor datasets are shown in Tables VI and VII. CLIC tightly fuses LiDAR, IMU, and camera measurements, and successfully generates good pose estimation in Handheld sequences collected

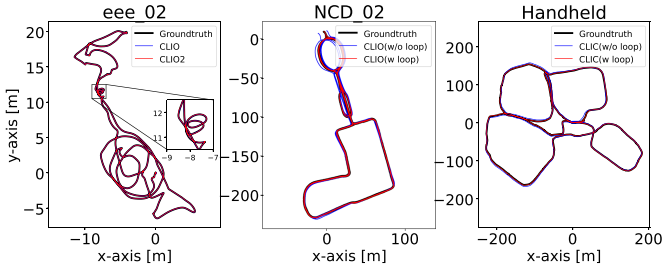


Fig. 9. Estimated trajectories compared to the groundtruth in eee_02 sequence of VIRAL dataset, NCD_02 sequence of NCD dataset, and Handheld sequence of LVI-SAM dataset.

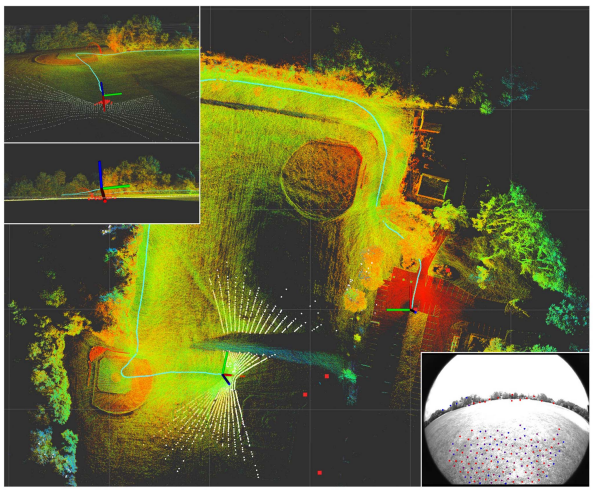


Fig. 10. LiDAR map (color points) and visual landmarks (red squares) during the system passing open areas when running Handheld sequence. The current scan (white points) only observes the ground while camera can track stable visual features.

over large open areas while CLIO fails, achieving significantly higher accuracy compared to the discrete-time method LVI-SAM [6]. It indicates that visual factors fused into CLIO can help improve the system's applicability. Fig. 10 showcases a snapshot on the Handheld sequence where the LiDAR could only detect the ground while camera can track stable visual features to further constrain the optimization problem. As discussed in Section V-B2, we only optimize the control points within the LI temporal sliding window for computation efficiency reasons, and the other control points involved in visual keyframe sliding window are kept fixed during optimization, which might degrade the effects of visual factors. In Tables VI and VII, LIC-Fusion 2.0 [10] shows pleasing performance in both indoor and outdoor scenarios, which benefits from reliable sliding-window plane-feature tracking. However, we can see that LIC-Fusion 2.0 has degraded performance in outdoor scenarios (see Table VI). The possible reason is that LIC-Fusion 2.0 relies on stably tracked plane features to update LiDAR poses, while the outdoor scenarios are filled with tree clumps, and can fail to provide sufficient structural planes compared to indoors.

The accuracy of CLIC can be further improved when enabling online timeoffset calibration between different sensors (see Table V). In Fig. 9, we also show some representative estimated

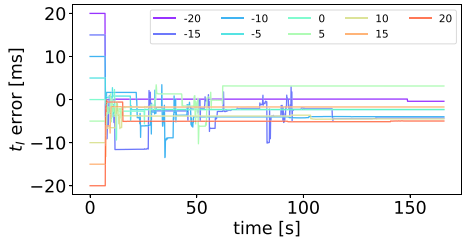


Fig. 11. Temporal calibration error of CLIC system in NCD_01 sequence of NCD dataset. Although we start from various initial values of timeoffsets, the estimates of timeoffsets are able to quickly converge.

TABLE VIII
TIMING OF DIFFERENT MODULES OF CLINS, CLIO AND CLIC IN EEE_01 SEQUENCE WITH A DURATION OF 397 SECONDS

	CLINS	CLIO	CLIC
Update local map	17.39	11.56	11.52
Update trajectory	1184.34	58.55	80.64
Update prior	0.00	8.45	8.44
Others	400.13	139.27	193.97
Total time cost	1601.86	217.82	294.57

trajectories of different methods aligned with the groundtruth trajectories.

D. Online Temporal Calibration

In this section, we examine the performance of online temporal calibration in the proposed multisensor fusion systems. Experiments are conducted on the NCD_01 sequence of NCD dataset [53]. We manually add additional timeoffsets of $-20\sim20$ ms to the raw timestamps, and the timeoffest t_I provided from NCD dataset is 0 ms. Fig. 11 shows the error of estimated timeoffset over time. We start to estimate the timeoffset 5 s after the start of the system, and the IMU timeoffset converges quickly, with most of the trials converging within 3 s. In addition, the final estimated timeoffset mean(std) is -2.0 ms (2.7 ms).

E. Runtime Analysis

We investigate runtime of the main modules in CLIO and CLIC in eee_01 sequence of VIRAL dataset [51]. Importantly, we notice the average runtime of proposed method remains almost constant whether in a long sequence or a short sequence. The method is implemented in C++ and executed on the desktop PC with an Intel i7-7700 K and 32-GB RAM, and Table VIII summarizes the time consumption of CLINS [17], CLIO, and CLIC. The term *Update Local Map* in Table VIII represents update local LiDAR map for associating LiDAR feature; *Update Trajectory* represents solving the problem in 12; and *Update Trajectory* represents the marginalization process. The term *Others* includes feature extraction of image and LiDAR cloud, trajectory initialization, data association, and so on. With analytical Jacobian computations for optimization and using sliding window and marginalization to bound computation complexity by the continuous-time fixed-lag smoothing, the enabled multisensor fusion methods (CLIO and CLIC) achieve real-time capability. In contrast, the continuous-time method, CLINS [17], consumes much more time and fails to run in real time. Here,

real-time performance refers to the total elapsed time to process measurements from the sensors is less than the sensor data collection time. We also note that our current implementation is not optimal, and there is still significant space to further improve efficiency.

VII. Conclusion

This article exploits continuous-time fixed-lag smoothing for asynchronous multisensor fusion in a factor-graph framework. Specifically, we propose to probabilistically marginalize old states and measurements out of the sliding window, and derive analytic Jacobians for continuous-time optimization. Benefiting from the nature of continuous-time trajectory formulation, heterogeneous multisensor measurements at any time instants can be seamlessly fused. Empowered by the continuous-time fixed-lag smoothing, we design the estimators at different sensor configurations, for tight fusion of multiple LiDARs, IMU, and camera sensors. Our estimators, including LI systems and LIC systems, show significant advances in pose estimation accuracy over the existing state-of-the-art methods. Online temporal calibration between sensors is also naturally supported in the continuous-time estimator. We demonstrate the accuracy and applicability of our method on three available public datasets and compare it with the state-of-the-art LI, LIC, and multi-LiDAR systems, respectively. Future work includes more efficient management of LiDAR map [54], utilizing more reliable LiDAR feature tracking and estimation method [10], and exploring the potential benefits of nonuniform B-spline.

Acknowledgment

The authors would like to thank Kewei Hu and Xiangrui Zhao for fruitful discussion.

References

[1] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[2] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," *Robot.: Sci. Syst.*, vol. 2, no. 9, 2014.

[3] 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, Aug. 2018.

[4] X. Zuo, P. Geneva, W. Lee, Y. Liu, and G. Huang, "LIC-Fusion: LiDAR-inertial-camera odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 5848–5854.

[5] X. Zuo et al., "Multimodal localization: Stereo over LiDAR map," *J. Field Robot.*, vol. 37, no. 6, pp. 1003–1026, 2020.

[6] T. Shan, B. Englot, C. Ratti, and D. Rus, "LVI-SAM: Tightly-coupled LiDAR-visual-inertial odometry via smoothing and mapping," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 5692–5698.

[7] J. Lin and F. Zhang, "R3LIVE: A robust, real-time, RGB-colored, LiDAR-inertial-visual tightly-coupled state estimation and mapping package," 2021, *arXiv:2109.07982*.

[8] M. Palieri et al., "LOCUS—A multi-sensor LiDAR-centric solution for high-precision odometry and 3D mapping in real-time," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 421–428, Apr. 2020.

[9] M. Li and A. I. Mourikis, "Online temporal calibration for camera–IMU systems: Theory and algorithms," *Int. J. Robot. Res.*, vol. 33, no. 7, pp. 947–964, 2014.

[10] X. Zuo et al., "LIC-Fusion 2.0: LiDAR-inertial-camera odometry with sliding-window plane-feature tracking," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5112–5119.

[11] T. Qin and S. Shen, "Online temporal calibration for monocular visual-inertial systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 3662–3669.

[12] Y. Yang, P. Geneva, X. Zuo, and G. Huang, "Online IMU intrinsic calibration: Is it necessary?," in *Proc. Robot.: Sci. Syst.*, Corvalis, OR, USA, Jul. 2020, doi: 10.15607/RSS.2020.XVI.026. [Online]. Available: http://www.roboticsproceedings.org/rss16/p026.html

[13] X. Lang, J. Lv, J. Huang, Y. Ma, Y. Liu, and X. Zuo, "Ctrl-vio: Continuous-time visual-inertial odometry for rolling shutter cameras," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 11537–11544, Aug. 2022.

[14] P. Geneva, K. Eckenhoff, Y. Yang, and G. Huang, "Lips: LiDAR-inertial 3D plane SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 123–130.

[15] C. Sommer, V. Usenko, D. Schubert, N. Demmel, and D. Cremers, "Efficient derivative computation for cumulative B-splines on lie groups," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11148–11156.

[16] T. Lowe, S. Kim, and M. Cox, "Complementary perception for handheld SLAM," *IEEE Robot. Automat. Lett.*, vol. 3, no. 2, pp. 1104–1111, Apr. 2018.

[17] J. Lv, K. Hu, J. Xu, Y. Liu, X. Ma, and X. Zuo, "CLINS: Continuous-time trajectory estimation for LiDAR-inertial system," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 6657–6663.

[18] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart, "Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 4304–4311.

[19] J. Lv, J. Xu, K. Hu, Y. Liu, and X. Zuo, "Targetless calibration of LiDAR-IMU system based on continuous-time batch estimation," 2020, *arXiv:2007.14759*.

[20] 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.

[21] C. Park, P. Moghadam, J. L. Williams, S. Kim, S. Sridharan, and C. Fookes, "Elasticity meets continuous-time: Map-centric dense 3D LiDAR slam," *IEEE Trans. Robot.*, vol. 38, no. 2, pp. 978–997, Apr. 2022.

[22] T.-C. Dong-Si and A. I. Mourikis, "Motion tracking with fixed-lag smoothing: Algorithm and consistency analysis," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 5655–5662.

[23] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *Int. J. Robot. Res.*, vol. 34, no. 3, pp. 314–334, 2015.

[24] A. J. Yang, C. Cui, I. A. Bârsan, R. Urtasun, and S. Wang, "Asynchronous multi-view slam," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 5669–5676.

[25] J. Zhang and S. Singh, "Laser–visual–inertial odometry and mapping with high robustness and low drift," *J. Field Robot.*, vol. 35, no. 8, pp. 1242–1264, 2018.

[26] Z. Wang, J. Zhang, S. Chen, C. Yuan, J. Zhang, and J. Zhang, "Robust high accuracy visual-inertial-laser slam system," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 6636–6641.

[27] S. Khattak, H. Nguyen, F. Mascarich, T. Dang, and K. Alexis, "Complementary multi–modal sensor fusion for resilient robot pose estimation in subterranean environments," in *Proc. IEEE Int. Conf. Unmanned Aircr. Syst.*, 2020, pp. 1024–1029.

[28] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback," *Int. J. Robot. Res.*, vol. 36, no. 10, pp. 1053–1072, 2017.

[29] D. Wisth, M. Camurri, S. Das, and M. Fallon, "Unified multi-modal landmark tracking for tightly coupled LiDAR-visual-inertial odometry," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 1004–1011, Feb. 2021.

[30] D. Wisth, M. Camurri, and M. Fallon, "VILENS: Visual, inertial, LiDAR, and leg odometry for all-terrain legged robots," 2021, *arXiv:2107.07243*.

[31] J. Lin, C. Zheng, W. Xu, and F. Zhang, "R2LIVE: A robust, real-time, LiDAR-inertial-visual tightly-coupled state estimator and mapping," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 7469–7476, Feb. 2021.

[32] S. Zhao, H. Zhang, P. Wang, L. Nogueira, and S. Scherer, "Super odometry: IMU-centric LiDAR-visual-inertial estimator for challenging environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 8729–8736.

[33] Y. Jia et al., "LVIO-fusion: A self-adaptive multi-sensor fusion SLAM framework using actor-critic method," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 286–293.

[34] 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, Aug. 2016.
[35] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 611–625, 2017.
[36] A. Reinke et al., "Locus 2.0: Robust and computationally efficient LiDAR odometry for real-time 3D mapping," *IEEE Robot. Automat. Lett.*, pp. 1–8, Oct. 2022.
[37] T.-M. Nguyen, S. Yuan, M. Cao, L. Yang, T. H. Nguyen, and L. Xie, "MILIOM: Tightly coupled multi-input LiDAR-inertia odometry and mapping," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 5573–5580, Jul. 2021.
[38] P. Furgale, T. D. Barfoot, and G. Sibley, "Continuous-time batch estimation using temporal basis functions," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 2088–2095.
[39] S. Li, S. Liu, Q. Zhao, and Q. Xia, "Quantized self-supervised local feature for real-time robot indirect vslam," *IEEE/ASME Trans. Mechatronics*, vol. 27, no. 3, pp. 1414–1424, Jun. 2022.
[40] T. Lyche and K. Morken, "Spline methods draft," *Dept. Inform., Center Math. Appl., Univ. Oslo*, Oslo, pp. 3–8, 2008.
[41] 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 *Proc. IEEE Int. Conf. 3D Vis.*, 2018, pp. 381–389.
[42] K. Qin, "General matrix representations for b-splines," in *Proc. IEEE Pacific Graph. 98. 6th Pacific Conf. Comput. Graph. Appl.*, 1998, pp. 37–43.
[43] J. Lv, X. Zuo, K. Hu, J. Xu, G. Huang, and Y. Liu, "Observability-aware intrinsic and extrinsic calibration of LiDAR-imu systems," *IEEE Trans. Robot.*, vol. 38, no. 6, pp. 3734–3753, Dec. 2022.
[44] S. Agarwal, K. Mierle, and T. C. S. Team, "Ceres solver," 2022. [Online]. Available: https://github.com/ceres-solver/ceres-solver
[45] Y. Yang, P. Geneva, X. Zuo, K. Eckenhoff, Y. Liu, and G. Huang, "Tightly-coupled aided inertial navigation with point and plane features," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 6094–6100.
[46] G. Sibley, L. Matthies, and G. Sukhatme, "Sliding window filter with application to planetary landing," *J. Field Robot.*, vol. 27, no. 5, pp. 587–608, 2010.
[47] J. Shi et al., "Good features to track," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 1994, pp. 593–600.
[48] B. D. Lucas et al., *An Iterative Image Registration Technique With an Application to Stereo Vision*, vol. 81. Vancouver, 1981.
[49] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "OpenVINS: A research platform for visual-inertial estimation," in *Proc. IEEE Int. Conf. Robot. Automat.*, Paris, France, 2020.
[50] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "Lio-sam: Tightly-coupled LiDAR inertial odometry via smoothing and mapping," 2020, *arXiv:2007.00258*.
[51] T.-M. Nguyen, S. Yuan, M. Cao, T. H. Nguyen, and L. Xie, "Viral SLAM: Tightly coupled camera-IMU-UWB-LiDAR SLAM," 2021, *arXiv:2105.03296*.
[52] T.-M. Nguyen, S. Yuan, M. Cao, Y. Lyu, T. H. Nguyen, and L. Xie, "NTU viral: A visual-inertial-ranging-LiDAR dataset, from an aerial vehicle viewpoint," *Int. J. Robot. Res.*, vol. 41, no. 3, pp. 270–280, 2022.
[53] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon, "The newer college dataset: Handheld LiDAR, inertial and vision with ground truth," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 4353–4360.
[54] C. Bai, T. Xiao, Y. Chen, H. Wang, F. Zhang, and X. Gao, "Faster-lio: Lightweight tightly coupled LiDAR-inertial odometry using parallel sparse incremental voxels," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4861–4868, Apr. 2022.

Jiajun Lv received the B.Eng. degree in automation from the Zhejiang University of Technology, Hangzhou, China, in 2018. She is currently working toward the Ph.D. degree in electronic and information engineering with the College of Control Science and Engineering, Zhejiang University, Hangzhou.

Her major research interests include sensor calibration, sensor fusion, spatial perception, and cognition.

Xiaolei Lang received the B.Eng. degree in automation from the Zhejiang University of Technology, Hangzhou, China, in 2020. He is currently working toward the M.S. degree in control science and engineering with the College of Control Science and Engineering, Zhejiang University, Hangzhou.

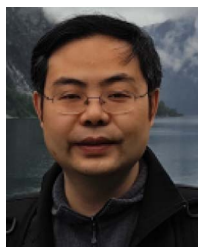
His major research interests include multisensor based calibration, odometry, and mapping.

Jinhong Xu received the MA.Sc. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2018.

He is currently a Researcher with the Institute of Cyber Systems and Control, Department of Control Science and Engineering, Zhejiang University. His latest research interests include SLAM, information processing, and robotic control.

Mengmeng Wang received the B.S. and M.S. degrees, in 2015 and 2018, respectively, in control science and engineering from Zhejiang University, Zhejiang, China, where she is currently working toward the Ph.D degree with the Laboratory of Advanced Perception on Robotics and Intelligent Learning, College of Control Science and Engineering.

Her research interests include visual tracking, action recognition, computer vision, and deep learning.



Yong Liu received the B.S. degree in computer science and engineering, and the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China, in 2001 and 2007, respectively.

He is currently a Professor with the Institute of Cyber-Systems and Control, College of Control Science and Engineering, Zhejiang University. His research interests include machine learning, robotics vision, multiple-sensor fusion, and intelligent systems.

Xingxing Zuo received the B.Eng. degree in mechanical engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2016, and the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2021.

He is currently a Postdoc Researcher with the Technical University of Munich, Munich, Germany. His research interests include computer vision, state estimation, sensor fusion, deep learning, localization, and mapping for autonomous robots in complex environments.

Dr. Zuo was the Finalist for the ICRA 2021 Best Paper Award in Robot Vision.