# FF-LINS: A Consistent Frame-to-Frame Solid-State-LiDAR-Inertial State Estimator

Hailiang Tang, Tisheng Zhang, Xiaoji Niu, Liqiang Wang, Linfu Wei, and Jingnan Liu

***Abstract*—Most of the existing LiDAR-inertial navigation systems are based on frame-to-map registrations, leading to inconsistency in state estimation. The newest solid-state LiDAR with a non-repetitive scanning pattern makes it possible to achieve a consistent LiDAR-inertial estimator by employing a frame-to-frame data association. In this letter, we propose a robust and consistent frame-to-frame LiDAR-inertial navigation system (FF-LINS) for solid-state LiDARs. With the INS-centric LiDAR frame processing, the keyframe point-cloud map is built using the accumulated point clouds to construct the frame-to-frame data association. The LiDAR frame-to-frame and the inertial measurement unit (IMU) preintegration measurements are tightly integrated using the factor graph optimization, with online calibration of the LiDAR-IMU extrinsic and time-delay parameters. The experiments on the public and private datasets demonstrate that the proposed FF-LINS achieves superior accuracy and robustness than the state-of-the-art systems. Besides, the LiDAR-IMU extrinsic and time-delay parameters are estimated effectively, and the online calibration notably improves the pose accuracy. The proposed FF-LINS and the employed datasets are open-sourced on GitHub (https://github.com/i2Nav-WHU/FF-LINS).**

***Index Terms*—LiDAR-inertial navigation, state estimation, factor graph optimization, multi-sensor fusion navigation.**

## I. Introduction

LIGHT detection and ranging (LiDAR) navigation system has been widely used in navigation and mapping in this century. Conventionally, the iteration closest point (ICP)-based methods [1], [2] and the normal distributions transform (NDT)-based methods [3], [4] have been adopted for pose estimation, but they are mainly for dense point-cloud registration. The LiDAR sensors employed in autonomous vehicles and robots are commonly low-cost, and we can only obtain sparse point clouds from a LiDAR frame. Besides, initial pose estimation is also required to achieve successful iterations [5]. Moreover, these methods are computationally intensive and may cost many computational resources. Due to these shortcomings, ICP-based and NDT-based methods are usually unsuitable for real-time navigation applications.

The real-time LiDAR odometry and mapping (LOAM) [6] is proposed without using the ICP or NDT. The edge and plane feature points are first extracted from a LiDAR frame by judging the smoothness of the local surface [6]. The LiDAR odometry is achieved by employing a frame-to-frame correspondence. The LiDAR frame is registered to the global feature map using the non-linear optimization method. The feature-point detection and the frame-to-frame methods in LOAM [6] are mainly designed for rotated 2-dimensional (2D) and 3-dimensional (3D) spinning LiDARs. LeGO-LOAM [7] further segments ground plane points and adopts a two-step optimization, yielding higher computational performance.

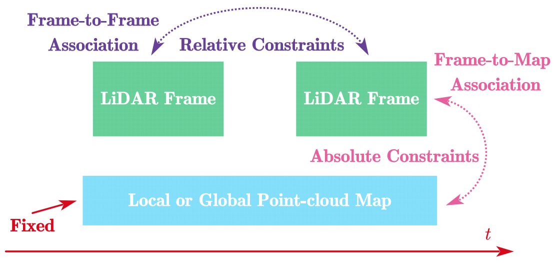

Fig. 1. An illustration of the frame-to-frame and the frame-to-map association in LiDAR navigation.

In state estimation, if the covariance of the estimated states cannot reflect the real estimation errors, then the estimator is inconsistent [8], [9]. Without a prebuilt map, the LiDAR navigation system should be a dead-reckoning (DR) system, and thus the yaw and position may drift over time with growing covariance [10]. However, a self-built map, such as a local point-cloud map [11]–[14] or a global point-cloud map [15], [16], has been used to build up the frame-to-map association, as depicted in Fig. 1. As the self-built map is fixed, the frame-to-map method constructs an absolute constraint between the current frame and the self-built map. Hence, in the frame-to-map state estimator, the covariance of the yaw and position will not grow, resulting in inconsistency in state estimation. This problem may be more significant in multi-sensor fusion navigation with a tightly-coupled formulation, as it is impossible to incorporate other absolute-positioning sensors, such as the global navigation satellite system (GNSS) [17].

The LiDAR point clouds are usually sampled at different times, which results in motion distortion. Hence, the micro-electro-mechanical system (MEMS) inertial measurement unit (IMU) can be employed to correct the distortion and construct a LiDAR-inertial navigation system. LOAM utilizes the orientation and acceleration from an IMU to remove motion distortion, exhibiting improved accuracy [6]. In LIO-SAM, the IMU is applied in a LiDAR-inertial state estimator [11] within

Manuscript received: July 16, 2023; Revised: September 5, 2023; Accepted: October 26, 2023. This paper was recommended for publication by Editor Javier Civera upon evaluation of the Associate Editor and Reviewers' comments. This research is partly funded by the National Key Research and Development Program of China (No. 2020YFB0505803) and the National Natural Science Foundation of China (No. 41974024). (*Corresponding authors: Tisheng Zhang; Xiaoji Niu.*)

The authors are with the GNSS Research Center, Wuhan University, Wuhan 430079, China (e-mail: thl@whu.edu.cn; zts@whu.edu.cn; xjniu@whu.edu.cn; wlq@whu.edu.cn; weilf@whu.edu.cn; jnliu@whu.edu.cn). Tisheng Zhang, Xiaoji Niu, and Jingnan Liu are also with the Hubei Luojia Laboratory, Wuhan 430079, China.

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

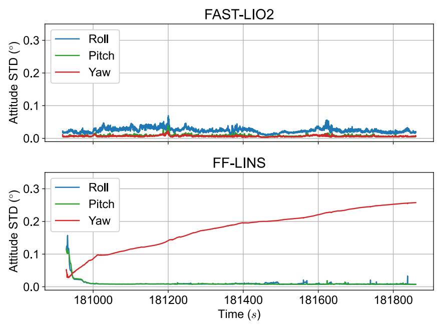

Fig. 2. Attitude standard deviation (STD) estimation comparison between FAST-LIO2 and FF-LINS on *Robot-campus* dataset. FAST-LIO2 indicates inconsistent estimation, as the yaw STD does not grow. In contrast, FF-LINS exhibits great consistency in state estimation, because the yaw STD grows over time, while the roll and pitch STDs converge due to their observability.

the framework of factor graph optimization (FGO) [18]. However, LIO-SAM [11] is a loosely-coupled system, as the LiDAR odometry is adopted in the state estimator rather than the raw LiDAR measurements. Besides, the LiDAR odometry in LIO-SAM is implemented by building a local point-cloud map, which is also inconsistent in state estimation, as depicted in Fig. 1. Hence, LIO-SAM has to employ a pose graph optimization [18] to fuse the LiDAR odometry and other absolute positioning sources, including the GNSS and the loop-closure constraint [11].

LINS [13] and FAST-LIO [14] are two similar tightly-coupled LiDAR-inertial odometry using the iterated extended Kalman filter (IEKF). The state estimation in [13], [14] is achieved by registering the extracted feature points in a LiDAR frame to the global feature-point map, which may result in inconsistency in the LiDAR-inertial state estimator. Specifically, the unobservable terms for a DR system, including the global yaw and the global position [8], can be wrongly observable by using the frame-to-map method, as depicted in Fig. 2. Moreover, the misalignment when registering the LiDAR frame to the global map may leading to inconsistent pose estimation relative to the IMU measurements. Thus, a wrong IMU biases estimation may occur and ruin the accuracy of the inertial navigation system (INS) [19]. In addition, the LiDAR-IMU extrinsic parameters cannot be effectively estimated with such a frame-to-map method, according to our experiments on FAST-LIO2.

Recently, the newly solid-state LiDAR with a non-repetitive scanning pattern has been widely used for navigation and mapping [12], [15], [16], [20]. LOAM-Livox employs the LOAM method for the solid-state LiDAR, Livox Mid-40, by adopting a new feature-extraction method [20]. For another solid-state LiDAR, Livox Horizon, with a different scanning pattern, LiLi-OM [12] proposes an applicable feature-extraction method. Besides, this method is integrated into a LiDAR-inertial scheme using a sliding-window optimization [12]. However, the frame-to-map method is employed in [12], and thus the inconsistent problem still exists. FAST-LIO2 [15] extends the work in FAST-LIO [14] by incorporating a direct registration method without feature extraction. Similarly, Faster-LIO [16] uses incremental voxels as the point-cloud spatial data structure rather than the incremental k-d tree in FAST-LIO2 [15]. Nevertheless, FAST-LIO2 and Faster-LIO adopt the same inconsistent state estimator.

TABLE I
THE DIFFERENCES BETWEEN THE FRAME-TO-MAP AND THE FRAME-TO-FRAME TIGHTLY-COUPLED LIDAR-INERTIAL SYSTEMS

| Property | The frame-to-map methods | The frame-to-frame methods |
|---|---|---|
| Constraint type | Absolute | Relative |
| Consistency | Inconsistent | Consistent |
| Observability | Wrongly observable for yaw angle and position | Unobservable for yaw angle and position |
| Extrinsic parameters | Cannot be estimated | Can be estimated |

As mentioned above, the LiDAR-inertial navigation system should be a DR system [10]. Hence, the LiDAR system should build up the frame-to-frame association to construct a relative constraint to achieve a consistent state estimation, as shown in Fig. 1. LIPS designs a LiDAR-Inertial 3D Plane simultaneous-localization-and-mapping (SLAM) system with a robust relative plane anchor factor in graph-based optimization for indoor applications [21]. However, the planes should be segmented offline using the Point Cloud Library (PCL) [22], which cannot run in real-time. LIC-Fusion 2.0 [23] proposes a sliding-window plane-feature tracking method, which is then integrated into a multi-state constraint Kalman filter (MSCKF) [8]. As the relative constraints with the frame-to-frame assocaition are constructed in LIPS and LIC-Fusion 2.0, the inconsistent problem in the state estimation should be solved. However, these methods are mainly designed for the 3D spinning LiDARs, and they are not applicable for solid-state LiDARs, such as Livox LiDARs with a non-repetitive scanning pattern. Besides, complex plane-extraction [21] or plane-association [23] algorithms should be employed to construct the frame-to-frame associations, and thus the computational complexity may significantly increase.

Table I exhibits the differences between the frame-to-map and the frame-to-frame tightly-coupled LiDAR-inertial systems. In this letter, we aim to construct a consistent solid-state-LiDAR-inertial navigation system (FF-LINS). We follow the INS-centric architecture in [17] to process the LiDAR data. A direct frame-to-frame data association algorithm is presented without explicitly extracting plane features. With the frame-to-frame association, a LiDAR frame-to-frame factor is proposed to construct a tightly-coupled LiDAR-inertial state estimator under the framework of FGO. The main contributions of our work are as follows:

- We propose a consistent solid-state-LiDAR-inertial state estimator that tightly integrates the LiDAR and IMU measurements within the FGO framework. The LiDAR-IMU extrinsic and time-delay parameters are all estimated and calibrated online to further improve the accuracy.
- A novel frame-to-frame data association algorithm is presented. We build a direct keyframe point-cloud map with accumulated LiDAR frames. The data association is achieved by finding the nearest points in the keyframe point-cloud maps within the sliding window.
- A frame-to-frame measurement model, which constructs a relative measurement, is proposed to achieve consistent state

estimation. The frame-to-frame measurement residuals, and the Jacobians for the IMU poses and the LiDAR-IMU extrinsic parameters, are all analytically expressed.

● The proposed FF-LINS is comprehensively evaluated on both public and private datasets. The experiment results demonstrate that FF-LINS with the proposed consistent state estimator yields improved accuracy and robustness.

The remainder of this paper is organized as follows. We give an overview of the system pipeline in section II. The proposed FF-LINS is presented in section III. The experiments and results are discussed in section IV for quantitative evaluation. Finally, we conclude the proposed FF-LINS.

## II. System Overview

The system overview of the proposed FF-LINS is depicted in Fig. 3. The system pipeline is in an INS-centric architecture, and the proposed FGO is a sliding-window optimizer [17]. Once the INS is initialized, the INS mechanization is conducted to provide prior poses for LiDAR frame processing. With the prior INS poses, the LiDAR frame is processed, and the LiDAR keyframe is selected. Then, we build the keyframe point-cloud map with accumulated LiDAR frames. Hence, the frame-to-frame data association can be conducted by finding the nearest points in all keyframe point-cloud maps within the sliding window. Finally, the LiDAR frame-to-frame measurements can be constructed between the LiDAR keyframes. The LiDAR and IMU measurements are tightly coupled within the FGO framework to perform the maximum-a-posterior estimation.

## III. Methodology

The proposed consistent frame-to-frame solid-state-LiDAR-inertial navigation system is presented in this section. We will first introduce the INS-centric LiDAR frame processing. Then, the direct frame-to-frame data association algorithm is proposed. Finally, we present the consistent state estimator with the analytical form of the frame-to-frame measurement residuals and Jacobians.

### *A. INS-Centric LiDAR Frame Processing*

We follow the INS-centric processing architecture [17] to process the LiDAR frame. The high-frequency INS poses will be employed to assist the LiDAR frame processing, including the motion-distortion compensation, the keyframe selection, and the keyframe point-cloud map building.

#### *1) INS Mechanization*

The INS is initialized firstly with zero position and zero yaw angle, while the roll and pitch angles are determined from the accelerometer measurements [19]. We can also obtain a rough gyroscope biases estimation if zero-velocity conditions are detected [17]. The INS mechanization is formulated by adopting the INS kinematic model as

$$\dot{\boldsymbol{p}}^{\mathrm{w}}_{\mathrm{wb}}=\boldsymbol{v}^{\mathrm{w}}_{\mathrm{wb}},\dot{\boldsymbol{v}}^{\mathrm{w}}_{\mathrm{wb}}=\mathbf{R}^{\mathrm{w}}_{\mathrm{b}}\boldsymbol{f}^{\mathrm{b}}+\boldsymbol{g}^{\mathrm{w}},\dot{\mathbf{q}}^{\mathrm{w}}_{\mathrm{b}}=\frac{1}{2}\mathbf{q}^{\mathrm{w}}_{\mathrm{b}}\otimes\begin{bmatrix}0\\ \boldsymbol{w}^{\mathrm{b}}\end{bmatrix}, \tag{1}$$

where $\boldsymbol{p}^{\mathrm{w}}_{\mathrm{wb}}$ and $\boldsymbol{v}^{\mathrm{w}}_{\mathrm{wb}}$ are the position and velocity of the IMU frame (b-frame) in the world frame (w-frame), respectively; the

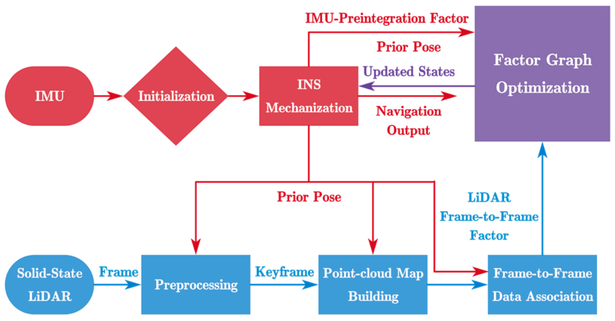

Fig. 3. System overview of the proposed FF-LINS.

quaternion $\mathbf{q}^{\mathrm{w}}_{\mathrm{b}}$ and the rotation matrix $\mathbf{R}^{\mathrm{w}}_{\mathrm{b}}$ denote the rotation of the b-frame with respect to the w-frame; $\boldsymbol{g}^{\mathrm{w}}$ is the gravity vector in the w-frame; $\boldsymbol{w}^{\mathrm{b}}$ and $\boldsymbol{f}^{\mathrm{b}}$ are the compensated angular velocity and acceleration from the IMU, respectively; $\otimes$ denotes the quaternion product. The INS mechanization can be formulated by adopting the kinematic model in (1) to obtain high-frequency INS poses.

#### *2) LiDAR Frame Preprocessing*

A keyframe-based LiDAR frame processing is employed in the proposed FF-LINS. When a new LiDAR frame is valid, we preprocess the LiDAR frame with the prior INS pose. Specifically, the interpolated INS poses are adopted to retrieve undistorted point clouds. The direct-based method is employed without explicitly extracting plane features. The undistorted point clouds of a LiDAR frame are directly downsampled using a voxel grid filter [22], and the leaf size is set to 0.5 m [15].

In the proposed INS-centric architecture, the state-estimation update will only be conducted when a new LiDAR keyframe is selected, and the INS can output continuous poses during the period [17], as depicted in Fig. 3. In other words, only the LiDAR keyframe will be adopted to perform state estimation. The proposed INS-centric processing can significantly save computational costs and thus improve real-time performance without decreasing accuracy.

To fully use the short-time accuracy of the INS, the LiDAR keyframe should be selected within a short interval. Besides, we should also consider LiDAR's motions to build up valid frame-to-frame LiDAR data association. If the translation or the rotation change exceeds thresholds [11], *e.g.* 0.4 m and 10 °, a new LiDAR keyframe will be selected. Here, the translation and the rotation are derived from the prior INS poses. If the motion of the LiDAR is small for a long interval, *e.g.* 0.5 s, we will also pick up a keyframe. When a new LiDAR keyframe is selected, the frame-to-frame data association can be conducted to perform the consistent state estimation. Nevertheless, the LiDAR non-keyframes will be reserved to build the point-cloud map corresponding to the new LiDAR keyframe.

#### *3) Point-Cloud Map Building*

The point clouds of a single solid-state-LiDAR are sparse [20], which is inconducive for the frame-to-frame data association. Relatively dense point clouds can be obtained by accumulating several LiDAR frames due to the non-repetitive scanning pattern of the solid-state LiDAR. Hence, it is convenient to construct the frame-to-frame data association

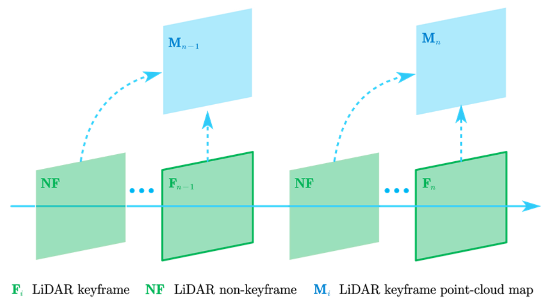

Fig. 4. An illustration of the LiDAR keyframe and keyframe point-cloud map.

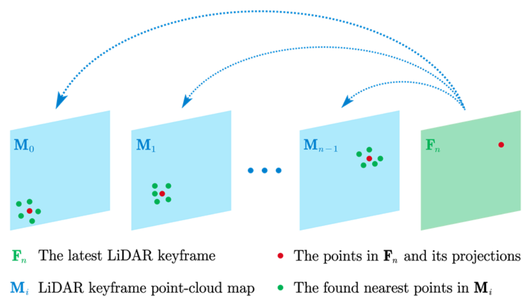
$\mathbf{M}_0$
$\mathbf{M}_1$
$\mathbf{M}_{n-1}$
$\mathbf{F}_n$
$\mathbf{F}_n$ The latest LiDAR keyframe
$\mathbf{M}_i$ LiDAR keyframe point-cloud map
● The points in $\mathbf{F}_n$ and its projections
● The found nearest points in $\mathbf{M}_i$

Fig. 5. An illustration of the frame-to-frame data association.

with such dense point clouds. As we can obtain high-accuracy poses from the INS in a short time, the LiDAR frames can be further accumulated with the prior INS poses. Specifically, all LiDAR frames since the previous keyframe, including the non-keyframes and the new keyframe, will be employed together to build the point-cloud map corresponding to the new keyframe. As depicted in Fig. 4, the point clouds in LiDAR non-keyframes will be projected to the corresponding time of the LiDAR keyframe with the prior poses from the INS. Finally, we obtain the LiDAR keyframe point-cloud map $\mathbf{M}$, which will be adopted for the frame-to-frame data association. The keyframe point-cloud map is also downsampled using the voxel grid filter in section III.A.2.

## *B. Frame-to-Frame Data Association*

As mentioned above, only the LiDAR keyframe will be adopted for state estimation. Specifically, only the point clouds in the LiDAR keyframe will be employed to construct the frame-to-frame data association. In other words, the keyframe point-cloud map $\mathbf{M}$ covers much more fields of view than the LiDAR keyframe point cloud $\mathbf{F}$. That is the reason that we can build up valid frame-to-frame data associations.

As depicted in Fig. 5, if we have $n+1$ LiDAR keyframes in the sliding window, we will associate the latest keyframe $\mathbf{F}_n$ with the keyframe point-cloud maps $\mathbf{M}_i, i \in [0, n-1]$. For a point $\boldsymbol{p}^{\mathrm{r}_n}$ in $\mathbf{F}_n$, where $\mathrm{r}$ denotes the LiDAR frame (r-frame), it can be projected to the keyframe point-cloud maps with the prior LiDAR pose $\{\boldsymbol{p}^{\mathrm{w}}_{\mathrm{wr}_n}, \mathbf{q}^{\mathrm{w}}_{\mathrm{r}_n}\}$ from the INS and the estimated LiDAR pose $\{\boldsymbol{p}^{\mathrm{w}}_{\mathrm{wr}_i}, \mathbf{q}^{\mathrm{w}}_{\mathrm{r}_i}\}$. As shown in Fig. 5, the projection of the point $\boldsymbol{p}^{\mathrm{r}_n}$ in $\mathbf{M}_i$ can be written as

$$\boldsymbol{p}^{\mathrm{r}_i} = \left(\mathbf{R}^{\mathrm{w}}_{\mathrm{r}_i}\right)^T \left(\mathbf{R}^{\mathrm{w}}_{\mathrm{r}_n}\boldsymbol{p}^{\mathrm{r}_n} + \boldsymbol{p}^{\mathrm{w}}_{\mathrm{wr}_n} - \boldsymbol{p}^{\mathrm{w}}_{\mathrm{wr}_i}\right). \tag{2}$$

In the proposed FF-LINS, the frame-to-frame association is equal to the direct plane-point registration [15], and we treat all point clouds as plane-point candidates. With the projected point $\boldsymbol{p}^{\mathrm{r}_i}$ (the red points in Fig. 5), we find its five nearest points $\boldsymbol{p}_e, e \in [1,5]$ (the green points in Fig. 5) in the keyframe point-cloud map $\mathbf{M}_i$. An overdetermined linear equation can be constructed using the five nearest points to solve the following plane equation as

$$\boldsymbol{n}^T\boldsymbol{p} + d = 0, \tag{3}$$

where $\boldsymbol{p}$ is a point on the plane; $\boldsymbol{n}$ is the normalized normal vector of the plane; $d$ is a distance that satisfies the equation (3). The fitted local plane is checked by calculating the point-to-plane distance as

$$\mathrm{dis}_{\boldsymbol{p}_e} = \left|\boldsymbol{n}^T\boldsymbol{p}_e + d\right|, e \in [1,5]. \tag{4}$$

If $\mathrm{dis}_{\boldsymbol{p}_e} < 0.1m$ for all the five points, the fitted plane will be used for the following processing. Otherwise, the frame-to-frame association for the point $\boldsymbol{p}^{\mathrm{r}_n}$ in $\mathbf{M}_i$ is failed. A similar method in [6], [15] is used to check the $\mathrm{dis}_{\boldsymbol{p}^{\mathrm{r}_i}} = \left|\boldsymbol{n}^T\boldsymbol{p}^{\mathrm{r}_i} + d\right|$ to validate the frame-to-frame association.

Finally, we obtain the frame-to-frame associations between the latest LiDAR keyframe and other keyframes in the sliding window. The fitted plane parameters $\{\boldsymbol{n}, d\}$ for each frame-to-frame association will be employed to construct a LiDAR frame-to-frame measurement in FGO. Hence, we can build relative measurements between the latest LiDAR keyframe and other keyframes to achieve a consistent state estimation.

## *C. Factor Graph Optimization*

The INS information is fully utilized in the INS-centric LiDAR frame processing, and thus we obtain the undistorted LiDAR keyframe and the keyframe point-cloud map. The frame-to-frame data association is achieved by constructing plane measurements between the latest LiDAR keyframe and the keyframe point-cloud maps. Hence, the consistent LiDAR-inertial state estimator is achieved by tightly integrating the LiDAR frame-to-frame and IMU preintegration measurements within the FGO framework.

### *1) Formulation*

The proposed consistent state estimator is a sliding-window optimizer. The state vector $\boldsymbol{X}$ in FF-LINS can be defined as follows

$$\begin{aligned} \boldsymbol{x}_k &= \left[\boldsymbol{p}^{\mathrm{w}}_{\mathrm{wb}_k}, \mathbf{q}^{\mathrm{w}}_{\mathrm{b}_k}, \boldsymbol{v}^{\mathrm{w}}_{\mathrm{wb}_k}, \boldsymbol{b}_{g_k}, \boldsymbol{b}_{a_k}\right], \boldsymbol{x}^{\mathrm{b}}_{\mathrm{r}} = \left[\boldsymbol{p}^{\mathrm{b}}_{\mathrm{br}}, \mathbf{q}^{\mathrm{b}}_{\mathrm{r}}\right], \\ \boldsymbol{X} &= \left[\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{x}^{\mathrm{b}}_{\mathrm{r}}, t_d\right], \end{aligned} \tag{5}$$

where $\boldsymbol{x}_k, k \in [0, n]$ is the IMU state at each time node, including the position, the attitude quaternion, and the velocity in the w-frame, and the gyroscope biases $\boldsymbol{b}_g$ and the accelerometer biases $\boldsymbol{b}_a$; $n$ is the size of the sliding window, *i.e.* the number of the IMU preintegration factors; $\boldsymbol{x}^{\mathrm{b}}_{\mathrm{r}}$ is the LiDAR-IMU extrinsic parameters; $t_d$ denotes the time delay between the LiDAR and the IMU data.

The state estimation is conducted by solving the following non-linear least squares problems of the form

$$\min_{\boldsymbol{X}} \left\{ \begin{array}{c} \sum_{j\in[0,m]} \left\| \mathbf{r}_R\left(\tilde{\boldsymbol{z}}_j^R, \boldsymbol{X}\right) \right\|_{\boldsymbol{\Sigma}_j^R}^2 + \\ \sum_{k\in[1,n]} \left\| \mathbf{r}_{Pre}\left(\tilde{\boldsymbol{z}}_{k-1,k}^{Pre}, \boldsymbol{X}\right) \right\|_{\boldsymbol{\Sigma}_{k-1,k}^{Pre}}^2 + \left\| \mathbf{r}_p - \mathbf{H}_p\boldsymbol{X} \right\|^2 \end{array} \right\}, \tag{6}$$

where $\mathbf{r}_R$ are the residuals for the LiDAR frame-to-frame measurements, which construct relative pose constraints between the latest LiDAR keyframe and other keyframes in the sliding window; $m$ denotes the total number of the LiDAR measurements; $\mathbf{r}_{Pre}$ are the residuals for the IMU preintegration measurements [24]; $\{\mathbf{r}_p, \mathbf{H}_p\}$ denote the prior information from the marginalization [25]. We adopt the Levenberg-Marquardt algorithm in Ceres solver [26] to solve the non-linear least squares problem in (6).

*2) Frame-to-Frame Measurement Residuals*

The LiDAR frame-to-frame measurement residual is equal to the point-to-plane distance [12]. Nevertheless, the proposed frame-to-frame measurement model is consistent, while the frame-to-map measurement model in the existing works is inconsistent, such as [11]–[13], [15], [16], as shown in Table I. The LiDAR-inertial extrinsic parameters $\left\{\boldsymbol{p}_{\mathrm{br}}^{\mathrm{b}}, \mathbf{q}_{\mathrm{r}}^{\mathrm{b}}\right\}$ and the time delay $t_d$ are all incorporated into the frame-to-frame measurement model for online estimation and calibration. For convenience, the time delay $t_d$ will be omitted in the following parts, and we can refer to [23] for further details.

Suppose the raw point $\boldsymbol{p}^{\mathrm{r}_n}$ in $\mathbf{F}_n$ is associated in the keyframe point-cloud map $\mathbf{M}_i$. Then, we have the associated plane parameters $\left\{\tilde{\boldsymbol{n}}, \tilde{d}\right\}$. The residual of the LiDAR frame-to-frame measurement is the function of the IMU poses $\left\{\boldsymbol{p}_{\mathrm{wb}_n}^{\mathrm{w}}, \mathbf{q}_{\mathrm{b}_n}^{\mathrm{w}}\right\}$ and $\left\{\boldsymbol{p}_{\mathrm{wb}_i}^{\mathrm{w}}, \mathbf{q}_{\mathrm{b}_i}^{\mathrm{w}}\right\}$, and the LiDAR-IMU extrinsic parameters $\left\{\boldsymbol{p}_{\mathrm{br}}^{\mathrm{b}}, \mathbf{q}_{\mathrm{r}}^{\mathrm{b}}\right\}$. The raw point $\boldsymbol{p}^{\mathrm{r}_n}$ in $\mathbf{F}_n$ can be projected to the $\mathbf{M}_i$ step-by-step as follows

$$\begin{aligned} \boldsymbol{p}^{\mathrm{b}_n} &= \mathbf{R}_{\mathrm{r}}^{\mathrm{b}}\boldsymbol{p}^{\mathrm{r}_n} + \boldsymbol{p}_{\mathrm{br}}^{\mathrm{b}}, \\ \boldsymbol{p}^{\mathrm{w}} &= \mathbf{R}_{\mathrm{b}_n}^{\mathrm{w}}\boldsymbol{p}^{\mathrm{b}_n} + \boldsymbol{p}_{\mathrm{wb}_n}^{\mathrm{w}}, \\ \boldsymbol{p}^{\mathrm{b}_i} &= \left(\mathbf{R}_{\mathrm{b}_i}^{\mathrm{w}}\right)^T\left(\boldsymbol{p}^{\mathrm{w}} - \boldsymbol{p}_{\mathrm{wb}_i}^{\mathrm{w}}\right), \\ \boldsymbol{p}^{\mathrm{r}_i} &= \left(\mathbf{R}_{\mathrm{r}}^{\mathrm{b}}\right)^T\left(\boldsymbol{p}^{\mathrm{b}_i} - \boldsymbol{p}_{\mathrm{br}}^{\mathrm{b}}\right), \end{aligned} \tag{7}$$

where $\boldsymbol{p}^{\mathrm{b}_n}$ and $\boldsymbol{p}^{\mathrm{b}_i}$ denote the projections of the raw point $\boldsymbol{p}^{\mathrm{r}_n}$ in the b-frame of the LiDAR keyframe $\mathbf{F}_i$ and $\mathbf{F}_n$; $\boldsymbol{p}^{\mathrm{w}}$ denotes the projection in the w-frame; $\boldsymbol{p}^{\mathrm{r}_i}$ is the projection in the r-frame of the keyframe $\mathbf{F}_i$. Hence, the LiDAR frame-to-frame measurement residual can be written as

$$\mathbf{r}_R\left(\tilde{\boldsymbol{z}}^R, \boldsymbol{X}\right) = \left(\tilde{\boldsymbol{n}}\right)^T\boldsymbol{p}^{\mathrm{r}_i} + \tilde{d}, \tag{8}$$

The LiDAR frame-to-frame measurement model in (7) and (8) is similar to the visual reprojection model in visual navigation [17], because they are all relative measurement models. In other words, the proposed LiDAR frame-to-frame measurement model is consistent in terms of state estimation.

As the direct method without explicitly extracting the plane feature is adopted, the covariance $\boldsymbol{\Sigma}^R$ may be difficult to determine. Thanks to the frame-to-frame association method, the covariance $\boldsymbol{\Sigma}^R$ can be obtained offline by quantitative error statistics. Specifically, we can first generate the keyframe point clouds $\mathbf{F}$ and keyframe point-cloud maps $\mathbf{M}$ with FF-LINS. Then, the ground-truth poses can be employed to build the frame-to-frame associations and calculate the frame-to-frame measurement errors. Finally, we can analyze the distributions of all the frame-to-frame measurement errors to obtain the covariance. According to our experiments, the quantitative result of the STD is about 0.088 m. In practice, the covariance is set as $\boldsymbol{\Sigma}^R = \sigma^2\mathbf{I}$ and $\sigma = 0.1m$, which will be evaluated in section IV.D.

*3) Jacobians of the Frame-to-Frame Measurement Residual*

Using the error-perturbation method in [24], we can obtain the analytical Jacobians of $\boldsymbol{r}_R$ in (8) w.r.t the pose errors $\left\{\delta\boldsymbol{p}_{\mathrm{wb}_n}^{\mathrm{w}}, \delta\boldsymbol{\phi}_{\mathrm{wb}_n}^{\mathrm{w}}\right\}$ and $\left\{\delta\boldsymbol{p}_{\mathrm{wb}_i}^{\mathrm{w}}, \delta\boldsymbol{\phi}_{\mathrm{wb}_i}^{\mathrm{w}}\right\}$, and the LiDAR-IMU extrinsic errors $\left\{\delta\boldsymbol{p}_{\mathrm{br}}^{\mathrm{b}}, \delta\boldsymbol{\phi}_{\mathrm{r}}^{\mathrm{b}}\right\}$. Here, $\boldsymbol{\phi}$ denotes the rotation vector of a quaternion $\mathbf{q}$, and $\delta\boldsymbol{\phi}$ represents the attitude errors. Specifically, the Jacobians w.r.t the pose errors $\left\{\delta\boldsymbol{p}_{\mathrm{wb}_n}^{\mathrm{w}}, \delta\boldsymbol{\phi}_{\mathrm{wb}_n}^{\mathrm{w}}\right\}$ can be formulated as

$$\left\{ \begin{aligned} \frac{\partial\mathbf{r}_R}{\partial\delta\boldsymbol{p}_{\mathrm{wb}_n}^{\mathrm{w}}} &= \left(\tilde{\boldsymbol{n}}\right)^T\left(\mathbf{R}_{\mathrm{r}}^{\mathrm{b}}\right)^T\left(\mathbf{R}_{\mathrm{b}_i}^{\mathrm{w}}\right)^T \\ \frac{\partial\mathbf{r}_R}{\partial\delta\boldsymbol{\phi}_{\mathrm{wb}_n}^{\mathrm{w}}} &= -\left(\tilde{\boldsymbol{n}}\right)^T\left(\mathbf{R}_{\mathrm{r}}^{\mathrm{b}}\right)^T\left(\mathbf{R}_{\mathrm{b}_i}^{\mathrm{w}}\right)^T\mathbf{R}_{\mathrm{b}_n}^{\mathrm{w}}\left[\boldsymbol{p}^{\mathrm{b}_n}\right]_\times \end{aligned} \right., \tag{9}$$

where $[\bullet]_\times$ denotes the skew-symmetric matrix of a vector [19]; $\boldsymbol{p}^{\mathrm{b}_n}$ is the point projection in (7). Similarly, the Jacobians w.r.t the pose errors $\left\{\delta\boldsymbol{p}_{\mathrm{wb}_i}^{\mathrm{w}}, \delta\boldsymbol{\phi}_{\mathrm{wb}_i}^{\mathrm{w}}\right\}$ can be written as

$$\left\{ \begin{aligned} \frac{\partial\mathbf{r}_R}{\partial\delta\boldsymbol{p}_{\mathrm{wb}_i}^{\mathrm{w}}} &= -\left(\tilde{\boldsymbol{n}}\right)^T\left(\mathbf{R}_{\mathrm{r}}^{\mathrm{b}}\right)^T\left(\mathbf{R}_{\mathrm{b}_i}^{\mathrm{w}}\right)^T \\ \frac{\partial\mathbf{r}_R}{\partial\delta\boldsymbol{\phi}_{\mathrm{wb}_i}^{\mathrm{w}}} &= \left(\tilde{\boldsymbol{n}}\right)^T\left(\mathbf{R}_{\mathrm{r}}^{\mathrm{b}}\right)^T\left[\left(\mathbf{R}_{\mathrm{b}_i}^{\mathrm{w}}\right)^T\left(\boldsymbol{p}^{\mathrm{w}} - \boldsymbol{p}_{\mathrm{wb}_i}^{\mathrm{w}}\right)\right]_\times \end{aligned} \right., \tag{10}$$

where $\boldsymbol{p}^{\mathrm{w}}$ is the point projection in (7). We can also obtain the Jacobians w.r.t the LiDAR-IMU extrinsic errors $\left\{\delta\boldsymbol{p}_{\mathrm{br}}^{\mathrm{b}}, \delta\boldsymbol{\phi}_{\mathrm{r}}^{\mathrm{b}}\right\}$ as

$$\left\{ \begin{aligned} \frac{\partial\mathbf{r}_R}{\partial\delta\boldsymbol{p}_{\mathrm{br}}^{\mathrm{b}}} &= \left(\tilde{\boldsymbol{n}}\right)^T\left(\mathbf{R}_{\mathrm{b}_n}^{\mathrm{r}_i} - \left(\mathbf{R}_{\mathrm{r}}^{\mathrm{b}}\right)^T\right) \\ \frac{\partial\mathbf{r}_R}{\partial\delta\boldsymbol{\phi}_{\mathrm{r}}^{\mathrm{b}}} &= \left(\tilde{\boldsymbol{n}}\right)^T\left(\left[\left(\mathbf{R}_{\mathrm{r}}^{\mathrm{b}}\right)^T\left(\boldsymbol{p}^{\mathrm{b}_i} - \boldsymbol{p}_{\mathrm{br}}^{\mathrm{b}}\right)\right]_\times - \mathbf{R}_{\mathrm{b}_n}^{\mathrm{r}_i}\mathbf{R}_{\mathrm{r}}^{\mathrm{b}}\left[\boldsymbol{p}^{\mathrm{r}_n}\right]_\times\right) \end{aligned} \right., \tag{11}$$

where $\boldsymbol{p}^{\mathrm{b}_i}$ is the point projection in (7), and $\boldsymbol{p}^{\mathrm{r}_n}$ is the raw point in the keyframe $\mathbf{F}_n$. The rotation matrix $\mathbf{R}_{\mathrm{b}_n}^{\mathrm{r}_i}$ can be written as

$$\mathbf{R}_{\mathrm{b}_n}^{\mathrm{r}_i} = \left(\mathbf{R}_{\mathrm{r}}^{\mathrm{b}}\right)^T\left(\mathbf{R}_{\mathrm{b}_i}^{\mathrm{w}}\right)^T\mathbf{R}_{\mathrm{b}_n}^{\mathrm{w}}. \tag{12}$$

Finally, we obtain the Jacobians of $\boldsymbol{r}_R$ w.r.t pose errors and the LiDAR-IMU extrinsic errors in (9), (10), and (11), which are all analytically expressed.

*4) Outlier Culling*

We adopt a two-step optimization when solving the non-linear least squares problems in (6). As wrong frame-to-frame associations may occur, especially in complex environments, we employ the Huber robust cost function [26] to reduce the impacts of the outliers. After the first optimization, a chi-square test is employed to determine and remove the LiDAR frame-to-frame factor outliers from the optimizer. The estimated states will be further optimized in the second optimization.

## IV. Experiments and Results

In this section, we conduct exhaustive experiments to evaluate the proposed FF-LINS. The public and private datasets are all employed to examine the accuracy and robustness of FF-LINS. The running-time analysis is also conducted to evaluate the real-time performance of FF-LINS.

### *A. Datasets and Implementation*

The employed public datasets are the *Lili-OM* [12] and *R3LIVE* datasets [27]. The *LiLi-OM* dataset includes the Livox Horizon with a frame rate 10Hz and the built-in IMU, and the three longest sequences are adopted, including the sequences *Schloss-1, Schloss-2*, and *East*. The *R3LIVE* dataset includes the Livox AVIA with a frame rate 10Hz and the built-in IMU, and the three longest sequences with end-to-end trajectories are adopted, including the sequences *hku_main_building*, *hkust_campus_00*, and *hkust_campus_01*.

The private datasets are collected with a low-speed wheeled robot with an average speed of around 1.5 m/s. The sensors include a solid-state LiDAR (Livox Mid-70 with a frame rate of 10 Hz) and an industrial-grade MEMS IMU (ADI ADIS16465 with a gyroscope bias instability of 2 °/hr and a frame rate of 200 Hz). The solid-state LiDAR and the IMU are well-synchronized through hardware triggers. The ground-truth system is a high-accuracy GNSS/INS integrated navigation system using the GNSS-RTK and a navigation-grade IMU [19]. Besides, the ground truth (0.02 m for position and 0.01 deg for attitude) is generated by a post-processing software. As depicted in Fig. 6, there are four sequences in the *Robot* dataset, including *campus* (1.33 km and 934 s), *building* (2.56 km and 1825 s), *playground* (1.33 km and 969 s), and *park* (1.46 km and 1326 s). The testing scenes contain various structured and unstructured environments. Besides, many moving objects, such as pedestrians, bicycles, and vehicles, make it very challenging to achieve robust navigation.

The proposed FF-LINS is implemented using C++ and the robot operating system (ROS). The sliding-window size $n$ is set to 10 to reduce the computational complexity. The LiDAR-IMU extrinsic and time-delay parameters are all uncalibrated on these datasets. The state-of-the-art (SOTA) LiLi-OM (without loop closure) [12], LIO-SAM (without loop closure) [11], and FAST-LIO2 [15] are employed for comparison. We adopt FF-LINS-WO (without the online calibration) to evaluate the impact of the online calibration of the LiDAR-IMU extrinsic and time-delay parameters. All the systems are run in real-time on a desktop PC (AMD R7-3700X) under the framework of ROS.

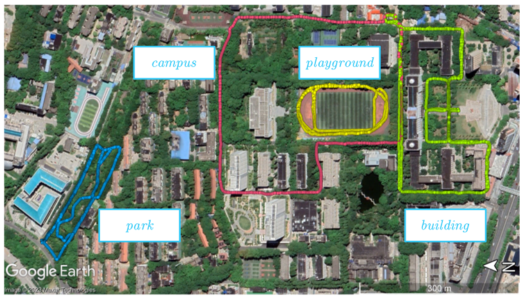

Fig. 6. Testing scenes in the *Robot* dataset. Different colors denote different sequences.

### *B. Evaluation of the Accuracy*

#### *1) Public LiLi-OM Dataset*

There is no ground truth in the *LiLi-OM* dataset, and we do not have the end-to-end reference. Hence, the GPS positioning results (with meter-level accuracy) at the starting and ending points are adopted to calculate the starting-ending distance. We also calculate the starting-ending distances of the employed LiDAR-inertial navigation systems. Finally, the distance errors on the *LiLi-OM* dataset are shown in Table II. According to the results in Table II, the proposed FF-LINS yields comparable accuracy to the SOTA methods. FAST-LIO2 exhibits degraded accuracy on *Schloss-2*, as its distance error is far larger than other systems. The distance errors are all meter-level on *Schloss-1* and *Schloss-2*, except for FAST-LIO2, and thus LiLi-OM, LIO-SAM, and FF-LINS achieve the same accuracy. Moreover, FF-LINS yields the best accuracy on *East*, which is the longest sequence.

#### *2) Public R3LIVE Dataset*

In the *R3LIVE* dataset, we have the end-to-end reference for quantitative evaluation. We fail to run LiLi-OM on the *R3LIVE* dataset, as LiLi-OM is designed for Livox Horizon rather than Livox AVIA in the *R3LIVE* dataset. We obtain the end-to-end results, as shown in Table III. LIO-SAM fails on *hku_main_building*, because of the few feature points in narrow indoor passages. Nevertheless, direct-based methods FAST-LIO2 and FF-LINS succeed in such environments. FF-LINS achieves superior accuracy than FAS-LIO2 on *hku_main_building* and *hkust_campus_00.* FAST-LIO2 yields the best end-to-end result on *hkust_campus_01.* The reason is that the frame-to-map methods, including FAST-LIO2 and LIO-SAM, may match with their built map when a loop occurs, as depicted in Fig. 7. However, they may also drift, and thus the frame-to-map matching will result in a large jump in the

TABLE II
Distance Errors on the *LiLi-OM* Dataset

| Error (m) | LiLi-OM | LIO-SAM | FAST_LIO2 | FF-LINS |
|---|---|---|---|---|
| *Schloss-1* | 1.36 | 0.47 | 1.10 | **0.23** |
| *Schloss-2* | 1.27 | **0.36** | 6.59 | 1.14 |
| *East* | 15.43 | 25.16 | 8.30 | **2.81** |
| Average | 6.02 | 8.66 | 5.33 | **1.39** |

TABLE III
End-to-End Errors on the *R3LIVE* Dataset

| Error (m) | LIO-SAM | FAST-LIO2 | FF-LINS-WO | FF-LINS |
|---|---|---|---|---|
| *hku_main_building* | Failed | 2.50 | 12.18 | **1.20** |
| *hkust_campus_00* | 3.29 | 3.69 | 14.16 | **2.41** |
| *hkust_campus_01* | 20.82 | **0.14** | 17.14 | 2.51 |
| Average | Invalid | 2.11 | 14.49 | **2.04** |

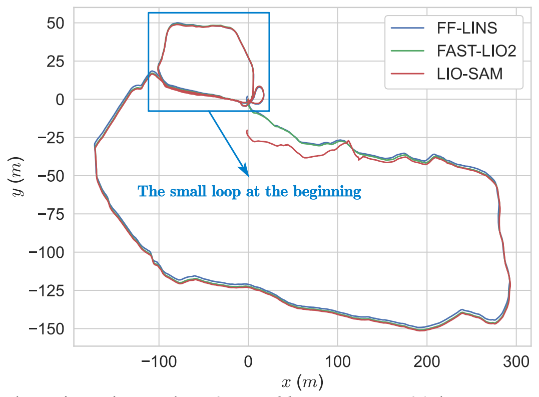

Fig. 7. The trajectories on the *R3LIVE-hkust_campus_01* dataset.

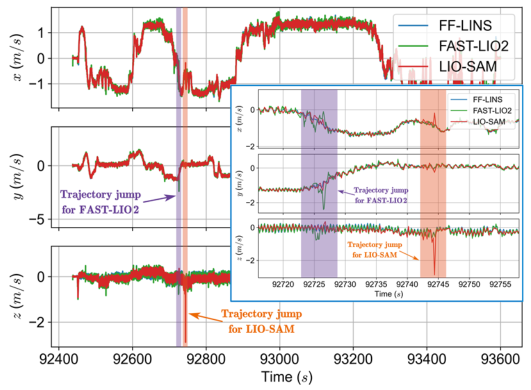

Fig. 8. The velocity curves on the *R3LIVE-hkust_campus_01* dataset. The velocities are obtained from the position changes.

trajectory, as shown in Fig. 8. Here, the velocity curve is employed because the trajectory jump can be more notable to be examined. Besides, *hkust_campus_00* and *hkust_campus_01* are collected in the same testing scenes. FAST-LIO2 exhibits different results on the two sequences, while the proposed FF-LINS yields a similar accuracy. Hence, the result for FF-LINS on *hkust_campus_01* is not so-called bad. In addition, FF-LINS yields the best accuracy on the *R3LIVE* dataset in terms of the average error.

### 3) *Private Robot Dataset*

We fail to run LiLi-OM and LIO-SAM on the *Robot* dataset. As Livox Mid-70 only contains only one scanning line, few feature points can be extracted, which is terrible for feature-based systems like LiLi-OM and LIO-SAM. The absolute rotation error (ARE) and absolute translation error (ATE) are adopted for quantitative evaluation. Table IV indicates that FF-LINS exhibits superior accuracy than FAST-LIO2 on all four sequences. The results may be the sparse single LiDAR frame of Livox Mid-70, resulting in fewer frame-to-map associations than for FAST-LIO2. Hence, FAST-LIO2 degrades accuracy in the *Robot* dataset, especially when large motions occur, which

TABLE IV
ARE AND ATE ON THE *ROBOT* DATASET

| ARE/ATE (deg/m) | FAST-LIO2 | FF-LINS-WO | FF-LINS (0.03 m) | FF-LINS (0.05 m) | FF-LINS (0.08 m) | FF-LINS (0.1 m) |
|---|---|---|---|---|---|---|
| *campus* | 3.55/4.42 | 2.45/2.17 | 0.68/2.42 | 0.58/2.12 | 0.46/1.72 | **0.41/1.51** |
| *building* | 3.13/3.12 | 2.23/2.24 | 0.91/2.47 | 0.73/2.25 | **0.64**/1.96 | 0.65/**1.90** |
| *playground* | 2.84/1.59 | 2.55/1.79 | 1.45/1.83 | 1.09/1.46 | 0.84/1.30 | **0.77/1.27** |
| *park* | 3.24/4.00 | 2.40/2.08 | **0.19/0.94** | 0.60/1.13 | 0.84/1.34 | 0.90/1.44 |
| Average | 3.19/3.28 | 2.41/2.07 | 0.81/1.92 | 0.75/1.74 | 0.70/1.58 | **0.68/1.53** |

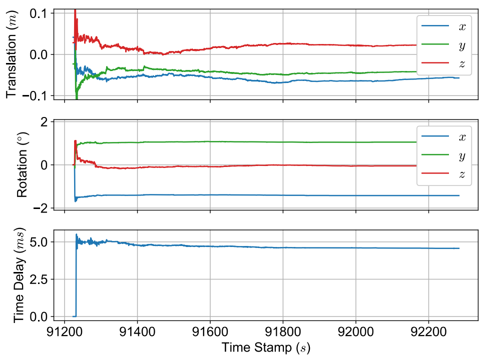

Fig. 9. Estimated LiDAR-IMU extrinsic and time-delay parameters on the *R3LIVE-hkust_campus_00*.

may result in few LiDAR frame-to-map measurements. In contrast, with the INS-centric architecture, the keyframe point-cloud maps are built with several LiDAR frames to construct the frame-to-frame association in FF-LINS. In other words, the proposed frame-to-frame association is more robust. Besides, the INS information is fully utilized in FF-LINS, and the LiDAR-IMU extrinsic and the time-delay parameters are all estimated and calibrated online. Hence, FF-LINS can achieve more consistent state estimation and thus can perform higher navigation accuracy.

## C. *The Impact of the Online Calibration*

FF-LINS-WO is adopted to evaluate the impact of the online calibration of the LiDAR-IMU extrinsic and time-delay parameters. Table III shows that FF-LINS-WO indicates significant accuracy degradation without the online calibration on the *R3LIVE* dataset. The reason is that the rotation parts of the LiDAR-IMU extrinsic parameters are relatively large. Specifically, the angles w.r.t the $x$ and $y$ axes (the horizontal attitude angles) are larger than 1.0 degrees, as depicted in Fig. 9. The horizontal attitude angles, *i.e.* the roll and pitch angles, are observable terms due to the gravity [8]. Hence, their impacts are much more significant in FF-LINS.

Moreover, all LiDAR-IMU extrinsic parameters converge, and even the tiny time-delay parameter converges. The results in Fig. 9 demonstrate that the proposed FF-LINS is consistent in state estimation; thus, these parameters can be effectively estimated. It should be noted the estimated parameters are almost the same on different sequences within a dataset.

The results in Table IV also demonstrate that the online calibration can notably improve the system accuracy on the *Robot* dataset, especially the rotation accuracy. The reason is that the angle w.r.t the $z$ axis (the yaw angle) of the LiDAR-IMU extrinsic parameters is larger than 2.0 degrees on the *Robot* dataset, according to our analyses. The yaw angle is an unobservable term [8], and thus its impact should be limited, as the translation accuracy only degrades a little.

## D. *The Impact of the Measurement Covariance*

We also conduct experiments on the *Robot* dataset to evaluate the impact of the measurement covariance. As the quantitative result of the STD is about 0.088 m in section III.C.2, we set the STD of the frame-to-frame measurements as 0.03 m,

TABLE V
THE AVERAGE RUNNING TIMES OF FF-LINS ON THE *ROBOT* DATASET

| Time (ms) | *campus* | *building* | *playground* | *park* |
|---|---|---|---|---|
| Frame-to-frame association | 2.7 | 2.6 | 2.7 | 2.7 |
| Factor graph optimization | 38.5 | 37.5 | 45.6 | 32.6 |
| Equivalent FPS | 42 | 44 | 40 | 62 |

0.05 m, 0.08 m, and 0.1 m, and the results are shown in Table IV. If the STD is smaller, the averages of the ARE and ATE are larger. When the STD is set to 0.08 m and 0.1 m, the results show almost no change. Hence, the STD of the frame-to-frame measurements can be set to 0.1 m in the proposed FF-LINS.

### *E. Running time analysis*

The average running times of FF-LINS on the *Robot* dataset are shown in Table V. The LiDAR frame preprocessing costs about 0.6 ms per frame. The FGO times vary on different datasets, as the number of valid frame-to-frame associations may be notably different. The results in Table V indicate that FF-LINS can achieve an equivalent frames per second (FPS) of 40~62, exhibiting superior real-time performance.

## V. CONCLUSIONS

This letter proposes a frame-to-frame solid-state-LiDAR-inertial state estimator, which achieves robust and consistent navigation in challenging environments. The proposed LiDAR measurement model can provide a relative constraint by constructing the direct frame-to-frame data association. Hence, the inconsistency problem in LiDAR-inertial navigation systems due to the frame-to-map association has been solved. The LiDAR-IMU extrinsic and time-delay parameters can be effectively estimated and calibrated online with the consistent state estimator. Besides, we do not need to extract feature points or segment and track plane points, significantly improving the real-time performance.

The proposed LiDAR frame-to-frame measurement model can be seamlessly incorporated into a multi-sensor fusion navigation system with absolute-positioning sensors, such as the GNSS and the high-precision map. Besides, the proposed method provides an effective solution for offline LiDAR-IMU calibrations. In addition, the frame-to-frame association can also be utilized for large-scale and consistent mapping by incorporating loop closure.

## REFERENCES

[1] P. J. Besl and N. D. McKay, “A method for registration of 3-D shapes,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, Feb. 1992.

[2] G. Pandey, S. Savarese, J. R. McBride, and R. M. Eustice, “Visually bootstrapped generalized ICP,” in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 2660–2667.

[3] P. Biber and W. Strasser, “The normal distributions transform: a new approach to laser scan matching,” in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, Las Vegas, Nevada, USA: IEEE, 2003, pp. 2743–2748.

[4] M. Magnusson, A. Lilienthal, and T. Duckett, “Scan registration for autonomous mining vehicles using 3D-NDT,” *J. Field Robot.*, vol. 24, no. 10, pp. 803–827, Oct. 2007.

[5] J. Zhang, Y. Yao, and B. Deng, “Fast and Robust Iterative Closest Point,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3450–3466, Jul. 2022.

[6] J. Zhang and S. Singh, “Low-drift and real-time lidar odometry and mapping,” *Auton. Robots*, vol. 41, no. 2, pp. 401–416, Feb. 2017.

[7] T. Shan and B. Englot, “LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid: IEEE, Oct. 2018, pp. 4758–4765.

[8] G. Huang, “Visual-Inertial Navigation: A Concise Review,” in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 9572–9582.

[9] M. Li and A. I. Mourikis, “Improving the accuracy of EKF-based visual-inertial odometry,” in *2012 IEEE International Conference on Robotics and Automation*, IEEE, 2012, pp. 828–835.

[10] C. Cadena *et al.*, “Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age,” *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.

[11] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, “LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 5135–5142.

[12] K. Li, M. Li, and U. D. Hanebeck, “Towards High-Performance Solid-State-LiDAR-Inertial Odometry and Mapping,” *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5167–5174, Jul. 2021.

[13] C. Qin, H. Ye, C. E. Pranata, J. Han, S. Zhang, and M. Liu, “LINS: A Lidar-Inertial State Estimator for Robust and Efficient Navigation,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 8899–8906.

[14] W. Xu and F. Zhang, “FAST-LIO: A Fast, Robust LiDAR-Inertial Odometry Package by Tightly-Coupled Iterated Kalman Filter,” *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3317–3324, Apr. 2021.

[15] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, “FAST-LIO2: Fast Direct LiDAR-Inertial Odometry,” *IEEE Trans. Robot.*, pp. 1–21, 2022.

[16] 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. Autom. Lett.*, vol. 7, no. 2, pp. 4861–4868, Apr. 2022.

[17] X. Niu, H. Tang, T. Zhang, J. Fan, and J. Liu, “IC-GVINS: A Robust, Real-Time, INS-Centric GNSS-Visual-Inertial Navigation System,” *IEEE Robot. Autom. Lett.*, vol. 8, no. 1, pp. 216–223, Jan. 2023.

[18] F. Dellaert and M. Kaess, “Factor Graphs for Robot Perception,” *Found. Trends Robot.*, vol. 6, no. 1–2, pp. 1–139, 2017.

[19] P. D. Groves, *Principles of GNSS, inertial, and multisensor integrated navigation systems*. in GNSS technology and applications series. Boston: Artech House, 2008.

[20] J. Lin and F. Zhang, “Loam livox: A fast, robust, high-precision LiDAR odometry and mapping package for LiDARs of small FoV,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 3126–3131.

[21] P. Geneva, K. Eckenhoff, Y. Yang, and G. Huang, “LIPS: LiDAR-Inertial 3D Plane SLAM,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid: IEEE, Oct. 2018, pp. 123–130.

[22] R. B. Rusu and S. Cousins, “3D is here: Point Cloud Library (PCL),” in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 1–4.

[23] X. Zuo *et al.*, “LIC-Fusion 2.0: LiDAR-Inertial-Camera Odometry with Sliding-Window Plane-Feature Tracking,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 5112–5119.

[24] H. Tang, T. Zhang, X. Niu, J. Fan, and J. Liu, “Impact of the Earth Rotation Compensation on MEMS-IMU Preintegration of Factor Graph Optimization,” *IEEE Sens. J.*, vol. 22, no. 17, pp. 17194–17204, Sep. 2022.

[25] G. Sibley, L. Matthies, and G. Sukhatme, “Sliding window filter with application to planetary landing: Sibley et al.: Sliding Window Filter,” *J. Field Robot.*, vol. 27, no. 5, pp. 587–608, Sep. 2010.

[26] Agarwal, Sameer, Mierle, Keir, and The Ceres Solver Team, “Ceres Solver.” Mar. 2022. [Online]. Available: https://github.com/ceres-solver/ceres-solver

[27] J. Lin and F. Zhang, “R3LIVE: A Robust, Real-time, RGB-colored, LiDAR-Inertial-Visual tightly-coupled state Estimation and mapping package,” in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 10672–10678.