SW-LIO: A Sliding Window Based Tightly Coupled LiDAR-Inertial Odometry

Zelin Wang, Xu Liu, Limin Yang, and Feng Gao

***Abstract*—This letter presents SW-LIO, a tightly coupled LiDAR-inertial odometry based on the sliding window approach. The proposed methodology encompasses rapid ground segmentation and the design of an iterative error-state Kalman filter (ESKF) to effectively fuse LiDAR point clouds and IMU measurements. By establishing a coupling relationship between the current state and the previous frames through the sliding window, the point clouds from the previous frames serve as a constraint for the current pose, resulting in more accurate state estimation. Furthermore, ground residual, bias residual and gravity residual are proposed, enabling more precise estimation of state variables beyond pose. These enhancements enable the system to deliver superior initial values for the filter in fast-moving or unstable environments, thereby improving the system's robustness. To evaluate the proposed framework, comprehensive testing has been conducted on public datasets as well as challenging real-world scenarios. The experimental results demonstrate that SW-LIO outperforms other state-of-the-art methods in terms of robustness and precision while maintaining similar time consumption.**

***Index Terms*—SLAM, localization, mapping.**

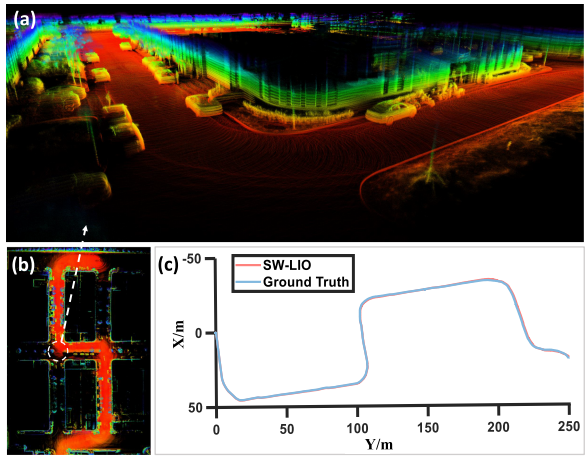

Fig. 1. Mapping results built by SW-LIO. The data collection is conducted within an outdoor industrial park, spanning a total path length of approximately 450 meters. (a) Mapping details. (b) Mapping results. (c) Comparison with ground truth.

I. Introduction

STATE estimation is a fundamental requirement for enabling autonomous capabilities in the majority of mobile robots, serving as a pivotal concern in the field of robotics [1]. To address the challenge of estimating the motion of a mobile robot in six degrees of freedom (DoF), researchers have proposed simultaneous Localization and Mapping (SLAM) technology [2]. Essentially, SLAM allows a mobile robot to simultaneously perform real-time localization and mapping when operating within an unknown environment. LiDAR SLAM has emerged as a prominent choice in SLAM technology, owing to its notable benefits such as a wide horizontal field of view (FOV) and robustness against variations in environmental lighting conditions [3], [4], [5].

However, LiDAR SLAM may suffer from motion distortion due to the sensor's self-motion. Furthermore, LiDAR SLAM is susceptible to degradation under challenging conditions due to the limited resolution between scan lines. Recent studies have demonstrated that the integration of inertial measurement unit (IMU) sensors can address the limitations of standalone LiDAR systems [6]. IMU sensors offer precise motion measurements at high frequencies within a short period and exhibit insensitivity to external environmental factors. By fusing IMU data, the LiDAR-inertial odometry (LIO) system can effectively mitigate the adverse effects of dynamic motion distortion, leading to enhanced accuracy and robustness.

LiDAR-inertial odometry system can be categorized into two main approaches: optimization-based methods and filter-based methods. Optimization-based methods, which tightly integrate thousands of LiDAR feature points and abundant IMU data in every computation cycle, demand substantial computational resources that may surpass the capacity of the robot onboard computers [7]. Compared with optimized optimization-based methods, filter-based LIO systems demonstrate superior efficiency. However, they may encounter limitations such as inadequate linearization approximation and suboptimal initial values, which will potentially reduce the overall performance.

In this letter, we propose SW-LIO, a lightweight tightly coupled LiDAR-inertial odometry based on the sliding window method. The mapping results built by SW-LIO is shown in Fig. 1. More specifically, our main contributions are as follows:

Manuscript received 21 June 2023; accepted 17 August 2023. Date of publication 4 September 2023; date of current version 11 September 2023. This letter was recommended for publication by Associate Editor J. Zhang and Editor J. Civera upon evaluation of the reviewers' comments. This work was supported by the National Natural Science Foundation of China under Grant 92248303. *(Corresponding author: Feng Gao.)*

The authors are with the Department of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: zlwang_97@sjtu.edu.cn; xu.liu@sjtu.edu.cn; ylm20159@sjtu.edu.cn; fengg@sjtu.edu.cn).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2023.3311371, provided by the authors.

Digital Object Identifier 10.1109/LRA.2023.3311371

1) A novel approach for rapid ground segmentation is proposed, considering both the height information of the point cloud and the distance from the LiDAR center. This method enables a more precise extraction of ground points and facilitates subsequent processing steps.
2) An iterative error-state Kalman filter based on the sliding window method is proposed. This approach effectively couples the point clouds from previous frames, allowing them to constrain the pose estimation of the current frame. As a result, the accuracy of the system is improved.
3) Novel residuals are designed, including ground residual, IMU bias residual, and gravity residual. These residuals contribute to obtaining more accurate estimates of bias and gravity vector, enabling the system to provide improved initial values for the filter in fast-moving or unstable environments. This enhancement contributes to the overall accuracy and robustness of the LIO system.

This letter is organized as follows: Section II reviews the related work of LiDAR-inertial odometry approaches. The overview of complete systems and the details of each key component are introduced in Section III. Section IV shows the experiment results, and Section V is the conclusion.

II. Related Work

LiDAR Simultaneous Localization and Mapping has emerged as a prominent research within the field of robotics, attracting significant attention in recent years. While a substantial body of literature exists on LiDAR odometry, this letter aims to narrow its focus to the comprehensive exploration of two specific approaches: Loosely-coupled and Tightly-coupled.

A. Loosely-Coupled LiDAR-Inertial Odometry

The Loosely-coupled LiDAR-inertial odometry involves individual processing of IMU and LiDAR measurements to calculate separate state estimation outcomes, which are subsequently fused. A well-known method called LOAM [5] presents a framework centered on LiDAR-based odometry and mapping. Incorporating IMU measurements aids in compensating for point cloud tilt and providing an initial estimation of the system's state. Hening et al. [8] present a novel framework that employs the adaptive extended Kalman filter for the estimation of the velocity and position of a UAV. The proposed approach integrates GPS corrections, while also incorporating an Inertial Navigation System as an additional input to enhance the accuracy of pose estimation. M. Demir et al. [9] adopts a probabilistic scan matching confidence estimation method to establish reliability estimation by fusing the LiDAR localization with the dead reckoning from sensors like IMU. Zhen et al. [10] employs the Kalman filter for sensor fusion and utilizes the Gaussian particle filter for measurement updates. The experimental results demonstrate the robustness of the system, including scenarios such as robot kidnapping.

In general, the Loosely-coupled methods exhibit notable computational efficiency and offer flexible implementation approaches. Nevertheless, its susceptibility to noise interference poses a significant accuracy reduction risk, primarily attributed to the system's lack of internal state correction for the IMU [11].

B. Tightly-Coupled LiDAR-Inertial Odometry

In contrast to the loosely-coupled methods, the tightly-coupled methods fuse the lidar point cloud and the original IMU measurement directly, aiming to optimize the estimation of state variables for all sensors. For optimization-based methods, Shan et al. [12] introduces a tightly coupled LiDAR-inertial odometry known as LIO-SAM. LIO-SAM employs factor graph optimization to fuse data from multiple sensors and obtain the optimal position and attitude of keyframes. Koide et al. [13] presents a LiDAR-inertial odometry that minimizes the global matching cost by tightly coupling the IMU preintegration factor and GICP matching cost factor. Nguyen et al. [14] propose a tightly coupled multi-input LiDAR-inertial odometry called MILIOM. It constructs a Factor graph and generates the estimation of the sliding window trajectory by fusing the feature-map matching coefficients and the IMU preintegration observation.

For filter-based methods, LINS [15] presents an iterative error-state Kalman filter that effectively updates the state by generating new feature correspondences in each iteration. FAST-LIO [16] introduces a new formulation for computing the Kalman gain, leading to significant enhancements in computational efficiency. Point-LIO [17] proposed a robust and high-bandwidth LiDAR-inertial odometry, which can estimate extremely aggressive robot movements. Taking inspiration from these works, we adopt the iterative error-state Kalman filter as the backbone of our framework, while improving accuracy through the utilization of a sliding window method and novel residuals.

III. Methodology

A. System Overview

Fig. 2 presents an overview of the system we proposed. The system starts by segmenting the ground points from the LiDAR points using the ground segmentation module. This helps with subsequent ground residual calculation and reduces computational consumption. The resulting points, along with IMU measurements, are then utilized as input for the iterative Kalman filter. Specifically, the sliding window stores LiDAR point clouds and the states of keyframes for computing residuals. Meanwhile, the IMU data is used for state propagation. The propagated states and covariance information contribute to residual computation and state update. Once the convergence condition is met, the optimal pose generates the output of the odometry and registers the corresponding point cloud in the global map. Furthermore, the global map provides search for kNN nearest neighbor points during residual calculation. The following sections will provide more detailed descriptions of each of these modules.

B. Ground Segmentation

The ground segmentation module is a critical component in our system, as it allows for ground points to be excluded from residual calculations, resulting in significant computational

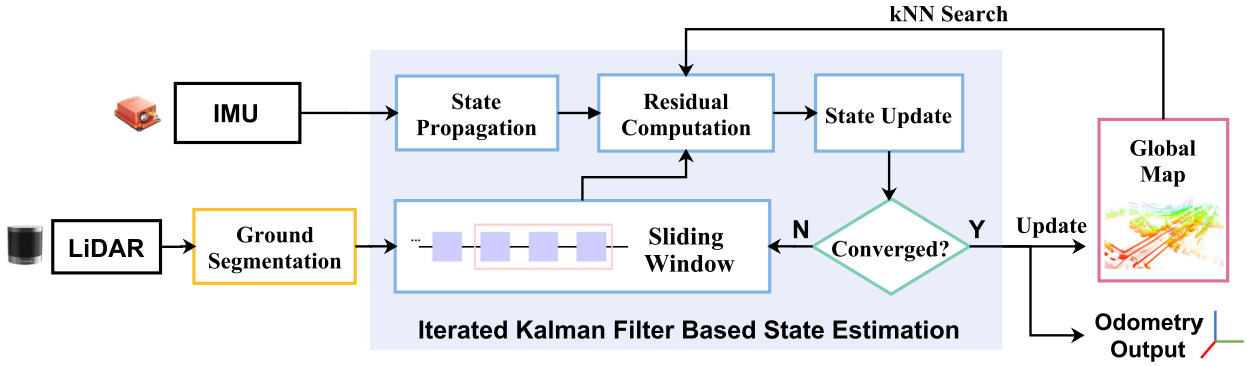

Fig. 2. System overview of SW-LIO. The fusion of IMU and LIDAR is accomplished through the utilization of an iterative Kalman filter based on sliding window method. Once the system attains convergence, it proceeds to generate the optimal pose and updates the map accordingly.

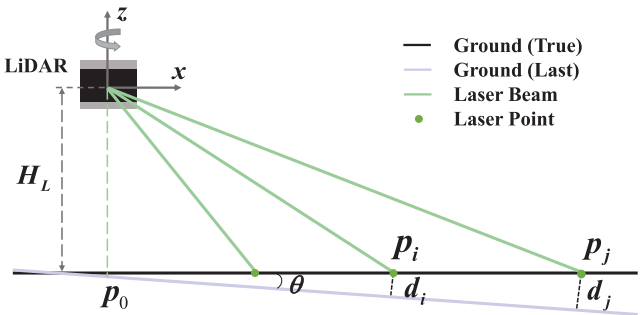

Fig. 3. Ground segmentation. The segmentation of ground points is conducted through a dual criterion approach, which takes into account both the point's elevation and the angle between the point and the ground in the last frame.

savings. Furthermore, ground points aid in obtaining superior pose estimation, which will be elaborated upon in the subsequent sections.

We adopt two criteria to jointly determine the ground points in a LiDAR point cloud. Firstly, the relative pose between the LiDAR and the mobile robot or autonomous vehicle remains fixed, which allows us to roughly filter out ground points based on their distance from the LiDAR in the vertical direction, as shown in Fig. 3 for the LiDAR installed horizontally. Then, we set the ground equation of the previous frame $k-1$ to $\mathbf{A}_{k-1}x + \mathbf{B}_{k-1}y + \mathbf{C}_{k-1}z + \mathbf{D}_{k-1} = 0$, and the LiDAR point cloud of frame k to $\mathcal{P}_k = \{p_1, p_2, \ldots, p_n\}$. As there exists a slight angle between the actual ground of the current frame and the ground of the previous frame, the distance from the point to the plane cannot be relied upon to determine whether point p is a ground point, as the distance between the ground points at different positions and the plane may differ considerably, as demonstrated in d_i and d_j in Fig. 3. Thus, based on the first criterion, we employ the following conditions to determine whether point p_i is a ground point:

$$\begin{cases} |H_L + z_{p_i}| \leqslant H_{threshold} \\ \frac{d_i}{\|p_i - p_0\|_2} \leqslant |\sin(\boldsymbol{\theta}_{threshold})| \end{cases} \tag{1}$$

Where $p_0 = (0, 0, -\mathbf{D}_{k-1}/\mathbf{C}_{k-1})$, $\boldsymbol{\theta}_{threshold}$ and $H_{threshold}$ are the set thresholds, H_L is the distance measured beforehand from LiDAR to the ground, and $\mathbf{d}_i$ is the distance from point p_i to the plane. The first criterion refers to whether the Z value of a point is close to the real ground, while the second criterion refers to whether the angle between the point and the previous frame plane is less than the threshold. As long as the above equation is satisfied, point p_i can be regarded as a ground point. Note that in the event the LiDAR is not installed horizontally, the point cloud can be initially transformed into a horizontal coordinate system. Finally, the RANSAC [18] is used to estimate the plane equation for the subsequent ground segmentation.

C. State Definition

The fusion output of IMU and LIDAR data will be obtained through the ESKF. In this process, the IMU coordinate system is denoted as I and the global coordinate system as G, with the first IMU frame serving as the origin of the global coordinate system. Our framework is tightly coupled and optimizes the IMU's bias and gravity vectors while estimating the state. To this end, we define the state as $\mathbf{x}$:

$$\begin{aligned} \mathcal{M} &= SO(3) \times \mathbb{R}^{15} \times [SO(3) \times \mathbb{R}^3]^{\mathbf{n}_w - 1} \\ \mathbf{x} &= \begin{bmatrix} \mathbf{x}_{curr}^T & \mathbf{x}_{prev}^T \end{bmatrix}^T \\ &= \begin{bmatrix} {}^G\mathbf{R}_I^T & {}^G\mathbf{p}_I^T & {}^G\mathbf{v}_I^T & \mathbf{b}_{\boldsymbol{\omega}}^T & \mathbf{b}_{\mathbf{a}}^T & {}^G\mathbf{g}^T & \mathbf{x}_{prev}^T \end{bmatrix}^T \in \mathcal{M} \end{aligned} \tag{2}$$

Where ${}^G\mathbf{R}_I$ and ${}^G\mathbf{p}_I$ represent the attitude and position of the IMU, ${}^G\mathbf{v}_I$ represents velocity, $\mathbf{b}_{\boldsymbol{\omega}}$ and $\mathbf{b}_{\mathbf{a}}$ represent the bias of the IMU, and ${}^G\mathbf{g}$ is the gravity vector. $\mathbf{n}_w$ denotes the number of keyframes in the sliding window, while $\mathbf{x}_{prev}$ corresponds to the state that will be optimized for all keyframes except for the current frame. Specifically, for the previous frames, only the attitude and position will be optimized. The sliding window module will be discussed further in the subsequent section. Furthermore, $\mathcal{M}$ represents a manifold that is locally homeomorphic to $\mathbb{R}^n$. To simplify derivation, two encapsulation operations, $\boxplus$ and $\boxminus$, are introduced to more efficiently represent the increment between

manifold elements in tangent vector space [19]:

$$\begin{bmatrix}\mathbf{R}\\ \mathbf{c}_1\end{bmatrix}\boxplus\begin{bmatrix}\boldsymbol{\xi}\\ \mathbf{c}_2\end{bmatrix}=\begin{bmatrix}\mathbf{R}\cdot\mathrm{Exp}(\boldsymbol{\xi})\\ \mathbf{c}_1+\mathbf{c}_2\end{bmatrix}$$
$$\begin{bmatrix}\mathbf{R}\\ \mathbf{c}_1\end{bmatrix}\boxminus\begin{bmatrix}\mathbf{R}_1\\ \mathbf{c}_2\end{bmatrix}=\begin{bmatrix}\mathrm{Log}(\mathbf{R}_1^{-1}\mathbf{R})\\ \mathbf{c}_1-\mathbf{c}_2\end{bmatrix} \tag{3}$$

where $\mathbf{c}_1,\mathbf{c}_2,\boldsymbol{\xi}\in\mathbb{R}^3$, $\mathbf{R},\mathbf{R}_1\in SO(3)$. $\mathrm{Exp}(\cdot)$ is an exponential mapping that maps vector $\mathbb{R}^3$ to $SO(3)$, and $\mathrm{Log}(\cdot)$ is its inverse map. We define the error state vector:

$$\begin{aligned}\delta\mathbf{x}&=\mathbf{x}\boxminus\hat{\mathbf{x}}\\&=\begin{bmatrix}\delta\boldsymbol{\theta}^T & \delta\boldsymbol{p}^T & \delta\mathbf{v}_I^T & \delta\mathbf{b}_{\boldsymbol{\omega}}^T & \delta\mathbf{b}_{\mathbf{a}}^T & \delta\mathbf{g}^T & \delta\mathbf{x}_{prev}^T\end{bmatrix}^T\end{aligned} \tag{4}$$

where $\hat{\mathbf{x}}$ represents the estimated state.

D. State Propagation

State propagation involves the estimation of the initial state and updating of the covariance matrix by utilizing IMU measurements between two successive LiDAR frames. The IMU's continuous motion model [16] is discretized to achieve this, specifically:

$$\mathbf{x}_{i+1}=\mathbf{x}_i\boxplus\left(\mathbf{f}\left(\mathbf{x}_i,\mathbf{w}_i\right)\Delta t_i\right) \tag{5}$$

where $\mathbf{w}=[\mathbf{n}_{\boldsymbol{\omega}}^T,\mathbf{n}_{\mathbf{a}}^T,\mathbf{n}_{\mathbf{b}\boldsymbol{\omega}}^T,\mathbf{n}_{\mathbf{ba}}^T]^T$ is the Gaussian noise vector, Δt_i is the IMU sampling period. $\mathbf{n}_{\boldsymbol{\omega}}^T$ and $\mathbf{n}_{\mathbf{a}}^T$ are the white noise of IMU angular velocity and acceleration, respectively, while $\mathbf{n}_{\mathbf{b}\boldsymbol{\omega}}^T$ and $\mathbf{n}_{\mathbf{ba}}^T$ are the random walk noise of angular velocity bias and acceleration bias. Function $\mathbf{f}$ is defined as below:

$$\mathbf{f}\left(\mathbf{x}_i,\mathbf{w}_i\right)=\begin{bmatrix}\boldsymbol{\omega}_i-\mathbf{b}_{\boldsymbol{\omega}_i}-\mathbf{n}_{\boldsymbol{\omega}_i}\\ {}^G\mathbf{v}_{I_i}\\ {}^G\mathbf{R}_{I_i}\left(\mathbf{a}_i-\mathbf{b}_{\mathbf{a}_i}-\mathbf{n}_{\mathbf{a}_i}\right)+{}^G\mathbf{g}\\ \mathbf{n}_{\mathbf{b}\boldsymbol{\omega}_i}\\ \mathbf{n}_{\mathbf{ba}_i}\\ \mathbf{0}_{(6\mathbf{n}_w-3)\times 1}\end{bmatrix} \tag{6}$$

where $\boldsymbol{\omega}_i$ and $\mathbf{a}_i$ are measurements of IMU.

When the new IMU measurement arrives, the error state $\delta\mathbf{x}$ and the error state covariance matrix $\mathbf{P}_k$ are propagated until the new LiDAR frame arrives. The state is propagated according to the following equation, with the process noise $\mathbf{w}_i$ set to zero:

$$\widehat{\mathbf{x}}_{i+1}=\widehat{\mathbf{x}}_i\boxplus\left(\mathbf{f}\left(\widehat{\mathbf{x}}_i,\mathbf{0}\right)\Delta t_i\right) \tag{7}$$

The linearized error state dynamic model is:

$$\delta\mathbf{x}_{i+1}=\mathbf{F}_{\mathbf{x}_i}\delta\mathbf{x}_i+\mathbf{F}_{\mathbf{w}_i}\mathbf{w}_i \tag{8}$$

where $\mathbf{F}_{\mathbf{x}_i}$ is the error state transition matrix and $\mathbf{F}_{\mathbf{w}_i}$ is the noise Jacobian:

$$\mathbf{F}_{\mathbf{x}_i}=\begin{bmatrix}\mathbf{F}_{11} & 0 & 0 & -\mathbf{I}\Delta t_i & 0 & 0 & 0\\ 0 & \mathbf{I} & \mathbf{I}\Delta t_i & 0 & 0 & 0 & 0\\ \mathbf{F}_{31} & 0 & \mathbf{I} & 0 & \mathbf{F}_{35} & \mathbf{I}\Delta t_i & 0\\ 0 & 0 & 0 & \mathbf{I} & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & \mathbf{I} & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & \mathbf{I} & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{I}_{\mathbf{N}}\end{bmatrix} \tag{9}$$

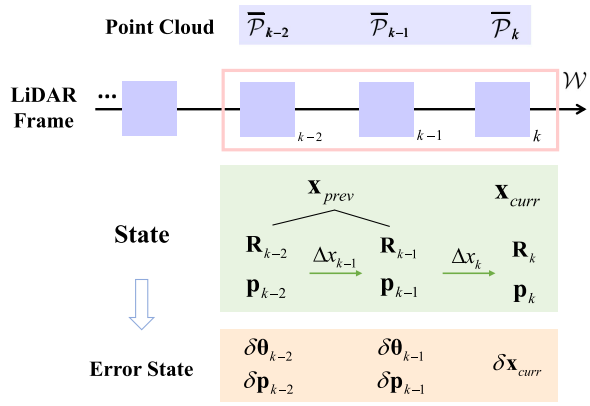

Fig. 4. Schematic representation of sliding window. It uses the point cloud of the current frame and the previous frame as the input of the filter, which also establishes a coupling relationship through the state change between adjacent frames.

$$\mathbf{F}_{\mathbf{w}_i}=\begin{bmatrix}-\mathbf{I}\Delta t_i & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & -{}^G\widehat{\mathbf{R}}_{I_i}\Delta t_i & 0 & 0\\ 0 & 0 & \mathbf{I}\Delta t_i & 0\\ 0 & 0 & 0 & \mathbf{I}\Delta t_i\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0_{\mathbf{M}}\end{bmatrix} \tag{10}$$

where $\mathbf{F}_{11}=\mathrm{Exp}((\boldsymbol{\omega}_i-\hat{\mathbf{b}}_{\boldsymbol{\omega}})\Delta t_i)$, $\mathbf{F}_{31}=-{}^G\widehat{\mathbf{R}}_{I_i}\lfloor\mathbf{a}_i-\hat{\mathbf{b}}_{\mathbf{a}}\rfloor_\times\Delta t_i$, $\mathbf{F}_{35}=-{}^G\widehat{\mathbf{R}}_{I_i}\Delta t_i$. $\mathbf{N}=(6\mathbf{n}_w-6)\times(6\mathbf{n}_w-6)$, $\mathbf{M}=(6\mathbf{n}_w-6)\times 3$. $\lfloor\cdot\rfloor_\times$ transfers a vector to its skewsymmetric matrix.

The covariance $\widehat{\mathbf{P}}$ is propagated as:

$$\widehat{\mathbf{P}}_{i+1}=\mathbf{F}_{\mathbf{x}_i}\widehat{\mathbf{P}}_i\mathbf{F}_{\mathbf{x}_i}^T+\mathbf{F}_{\mathbf{w}_i}\mathbf{Q}\mathbf{F}_{\mathbf{w}_i}^T \tag{11}$$

where $\mathbf{Q}$ is the covariance matrix of noise $\mathbf{w}$, which is computed off-line during sensor calibration.

E. Sliding Window

The recent LIO systems (such as LINS [15], FAST-LIO [16]) commonly rely on the current frame point cloud as input to estimate the current state. In contrast, we propose a sliding window approach that uses both the current and previous frame's point clouds as inputs to the Error-State Kalman Filter (ESKF). This approach provides the advantage of incorporating the point cloud data from previous frames, akin to a map-to-map matching process, that achieves more precise state estimation and increases the robustness of state estimation.

As shown in Fig. 4, the purple box represents a LiDAR frame that contains point cloud $\mathcal{P}$. In the context of the current frame, indexed as k-th, the residual computation excludes ground points. Accordingly, the set of all non-ground points in frame k is symbolized as $\overline{\mathcal{P}}_k$. The sliding window $\mathcal{W}$ contains all the state variables $\mathbf{x}=[\mathbf{x}_{curr}^T\ \mathbf{x}_{prev}^T]^T$ to be optimized. The state in the current frame and the previous frame is respectively

represented by $\mathbf{x}_{curr}$ and $\mathbf{x}_{prev}$:

$$\begin{aligned}\mathbf{x} &= \begin{bmatrix}\mathbf{x}_{curr}^T & \mathbf{x}_{prev}^T\end{bmatrix}^T \\ \mathbf{x}_{curr} &= \begin{bmatrix}\mathbf{R}_k^T & \mathbf{p}_k^T & \mathbf{v}_k^T & \mathbf{b}_{\boldsymbol{\omega}_k}^T & \mathbf{b}_{\mathbf{a}_k}^T & \mathbf{g}_k^T\end{bmatrix}^T \\ \mathbf{x}_{prev} &= \begin{bmatrix}\mathbf{R}_{k-1}^T & \mathbf{p}_{k-1}^T & \mathbf{R}_{k-2}^T & \mathbf{p}_{k-2}^T \cdots\end{bmatrix}^T \\ &\in [SO(3) \times \mathbb{R}^3]^{\mathbf{n}_w - 1}\end{aligned} \tag{12}$$

Where $\mathbf{n}_w$ denotes the number of keyframes in the sliding window. Note that for simplicity, the coordinate systems I and G are omitted. The optimization of the previous frame only includes the position and rotation, as optimizing other state variables is redundant due to state propagation's requirement for only the current frame's state variables. This approach presents an additional benefit, where the dimension of the sliding window's state remains acceptable, thus not overly consuming computational resources, compared to optimizing only the current frame. The error state can be calculated using (4).

Furthermore, it should be noted that the current frame's pose is not observable by the point cloud of the previous frame, indicating that the current pose is independent of the previous frame's point cloud. The incorporation of sliding windows aims to optimize the current pose, however, such optimization would be rendered futile if the aforementioned independence persists. Hence, to establish a coupling between the state of the current frame and the previous frame's point cloud, we introduce the change $\Delta\mathbf{x}$ of the state between adjacent frames each time a sliding window is established:

$$\begin{aligned}\Delta\mathbf{x}_k &= \begin{bmatrix}\hat{\mathbf{R}}_k \\ \hat{\mathbf{p}}_k\end{bmatrix} \boxminus \begin{bmatrix}\hat{\mathbf{R}}_{k-1} \\ \hat{\mathbf{p}}_{k-1}\end{bmatrix} \\ &= \begin{bmatrix}\Delta\hat{\boldsymbol{\theta}}_k^T & \Delta\hat{\boldsymbol{p}}_k^T\end{bmatrix}^T \in \mathbb{R}^6\end{aligned} \tag{13}$$

F. Residual Computation

1) LiDAR Residual: Firstly, for LiDAR point cloud, we use the point-to-plane distance in LOAM [5] as the residual. During the observation process, points exhibit ranging noise and beam-directing noise, represented as ${}^L\mathbf{n}_j$. The elimination of these noises results in ground truth points, which lie on their closest neighboring plane. Consequently, the LiDAR residual $\mathbf{z}_j^L$ can be determined by applying the following equation:

$$\mathbf{z}_j^L = {}^G\mathbf{l}_j^T \left(\mathbf{T}_k {}^I\mathbf{T}_L \left({}^L\mathbf{p}_j - {}^L\mathbf{n}_j\right) - {}^G\mathbf{r}_j\right) \tag{14}$$

Where ${}^G\mathbf{l}_j^T$ is the unit normal vector of the nearest neighbor plane, $\mathbf{T}_k = (\mathbf{R}_k, \mathbf{p}_k)$ represents the pose, and ${}^I\mathbf{T}_L = ({}^I\mathbf{R}_L, {}^I\mathbf{p}_L)$ represents the extrinsic between IMU and LIDAR, which can be obtained through prior offline calibration. ${}^G\mathbf{r}_j$ represents a point on the plane. The meaning of this equation is that a point in global coordinate system and a point ${}^G\mathbf{r}_j$ on the plane form a vector, and the product of this vector and the normal vector ${}^G\mathbf{l}_j^T$ of the plane is the distance from the point to the plane.

In addition, for point clouds of previous frames (taking $k-1$ frame as an example), it is essential to establish the current state and its coupling relationship. Based on (13), the following equation can be derived:

$$\begin{cases}\mathbf{R}_{k-1} = \mathbf{R}_k(\mathrm{Exp}(\Delta\hat{\boldsymbol{\theta}}_k))^T \mathrm{Exp}(\delta\boldsymbol{\theta}_{k-1}) \\ \mathbf{p}_{k-1} = \mathbf{p}_k - \Delta\hat{\boldsymbol{p}}_k + \delta\boldsymbol{p}_{k-1}\end{cases} \tag{15}$$

By substituting this equation into the residual computation, it is observed that the derivative of the residual with respect to the current pose is non-zero. This implies that the point cloud from the previous frame can also contribute to enhancing the optimization of the current pose.

2) Ground Residual: In addition, for ground points, we define the ground residual $\mathbf{z}_j^G$:

$$\mathbf{z}^G = (\mathbf{R}_{k-1} {}^I\mathbf{R}_L \vec{\mathbf{n}}_{k-1}) \cdot (\mathbf{R}_k {}^I\mathbf{R}_L \vec{\mathbf{n}}_k) - 1 \tag{16}$$

Where $\vec{\mathbf{n}}_k = (\mathbf{A}_k, \mathbf{B}_k, \mathbf{C}_k)^T$ represents the unit normal vector of the ground. The equation above demonstrates that the angle between ground normal vectors of consecutive frames should be close to zero. In practice, there is an option to incorporate ground residuals into the system design, contingent upon the specific ground conditions.

3) Bias and Gravity Residual: Moreover, it is noteworthy that the point-to-plane residual is solely dependent on the pose in the state. As a result, the derivative of residual with respect to other state variables equals zero. Consequently, the remaining state variables are minimally adjusted through their covariance with the pose in the subsequent update procedure, like FAST-LIO [16]. To improve the accuracy of estimating other state variables, we propose the following new residuals.

Regarding the angular velocity bias $\mathbf{b}_{\boldsymbol{\omega}}$, the rotational component in (5) can be expressed in the following equation:

$$\begin{aligned}\mathbf{R}_k &= \mathbf{R}_{k-1} \sum \mathbf{f}_R(\mathbf{x}_i, \mathbf{w}_i)\Delta t_i \\ &= \mathbf{R}_{k-1} \sum \mathrm{Exp}(\boldsymbol{\omega}_i - \mathbf{b}_{\boldsymbol{\omega}_i} - \mathbf{n}_{\boldsymbol{\omega}_i})\Delta t_i\end{aligned} \tag{17}$$

Where $\sum$ represents the sum of $\boxplus$, which is the same as the following. We define rotational residual:

$$\mathbf{z}^R = \left\| \mathbf{R}_k \boxminus \mathbf{R}_{k-1} - \mathrm{Log}\left(\sum \mathbf{f}_R(\mathbf{x}_i, \mathbf{w}_i)\Delta t_i\right)\right\|_2 \tag{18}$$

The noise $\mathbf{w}_i$ includes modeling error and measurement error. This residual is related to $\mathbf{b}_{\boldsymbol{\omega}}$, and its derivative with respect to $\mathbf{b}_{\boldsymbol{\omega}}$ is non-zero. Regarding the acceleration bias $\mathbf{b}_{\mathbf{a}}$ and the gravity vector ${}^G\mathbf{g}$, note the velocity component in (5):

$$\begin{aligned}\mathbf{v}_k &= \mathbf{v}_{k-1} \sum \mathbf{f}_v(\mathbf{x}_i, \mathbf{w}_i)\Delta t_i \\ &= \mathbf{v}_{k-1} \sum \left[{}^G\mathbf{R}_{I_i}\left(\mathbf{a}_i - \mathbf{b}_{\mathbf{a}_i} - \mathbf{n}_{\mathbf{a}_i}\right) + {}^G\mathbf{g}\right]\Delta t_i\end{aligned} \tag{19}$$

Thus, we define velocity residual:

$$\mathbf{z}^v = \left\| \mathbf{v}_k \boxminus \mathbf{v}_{k-1} - \sum \mathbf{f}_v(\mathbf{x}_i, \mathbf{w}_i)\Delta t_i \right\|_2 \tag{20}$$

This residual is related to $\mathbf{b}_{\mathbf{a}}$ and ${}^G\mathbf{g}$, and its derivative with respect to these two state variables is non-zero. The utilization of these two residuals can improve the accuracy of the estimation of other state variables, in addition to position and rotation.

G. State Update

In state update, all the residuals jointly determine the observation equation, and when all noise is taken into account, the residual should be zero. Furthermore, a first-order approximation at $\delta\mathbf{x}_k = \mathbf{0}$ yields:

$$\begin{aligned}\mathbf{0} &= \mathbf{h}\left(\mathbf{x}_k, \mathbf{w}_i, {}^L\mathbf{n}_j\right) = \mathbf{h}\left(\widehat{\mathbf{x}}_k \boxplus \delta\mathbf{x}_k, \mathbf{w}_i, {}^L\mathbf{n}_j\right) \\ &= \mathbf{h}_k\left(\widehat{\mathbf{x}}_k, \mathbf{0}, \mathbf{0}\right) + \mathbf{H}_k\delta\mathbf{x}_k + \mathbf{e}_j \\ &= \mathbf{z}_k + \mathbf{H}_k\delta\mathbf{x}_k + \mathbf{e}_j \end{aligned} \tag{21}$$

Where $\mathbf{H}_k$ represents the Jacobian matrix of $\mathbf{h}$ with respect to $\delta\mathbf{x}_k$, and $\mathbf{e}_j \in \mathcal{N}(\mathbf{0}, \mathcal{Q}_k)$ represents Gaussian noise. $\mathbf{z}_k = [\mathbf{z}_1^L, \ldots, \mathbf{z}_n^L, \mathbf{z}^G, \mathbf{z}^R, \mathbf{z}^v]^T$. Note that $\mathbf{P}_k$ denotes the covariance of the error state $\delta\mathbf{x}_k$, which is obtained through state propagation. In the subsequent iterations, the state will be updated. Hence, the covariance can be updated by:

$$\delta\mathbf{x}_k = (\widehat{\mathbf{x}}_k^\tau \boxplus \delta\mathbf{x}_k^\tau) \boxminus \widehat{\mathbf{x}}_k = \widehat{\mathbf{x}}_k^\tau \boxminus \widehat{\mathbf{x}}_k + \mathbf{J}^\tau\delta\mathbf{x}_k^\tau \tag{22}$$

Where τ is the current number of iterations, and $\mathbf{J}$ is the Jacobian matrix, thus $\widehat{\mathbf{P}}_k^\tau = (\mathbf{J}^\tau)^{-1}\widehat{\mathbf{P}}_k(\mathbf{J}^\tau)^{-T}$. Combine the prior distribution in (8) with the linearized measurement residual in (21) to obtain the maximum a-posterior estimation:

$$\min_{\delta\mathbf{x}_k^\tau}\left(\|\delta\mathbf{x}_k\|^2_{\widehat{\mathbf{P}}_k^{-1}} + \sum\|\mathbf{z}_k^\tau + \mathbf{H}_k^\tau\delta\mathbf{x}_k^\tau\|^2_{\mathcal{Q}_k^{-1}}\right) \tag{23}$$

Where $\mathcal{Q}_k = \mathrm{diag}(\mathcal{Q}_1, \cdots \mathcal{Q}_{n+3})$. Then the process of updating state estimation is accomplished using iterative Kalman filtering:

$$\begin{aligned}\mathbf{K}^\tau &= \left(\mathbf{H}_k^{\tau T}\mathcal{Q}_k^{-1}\mathbf{H}_k^\tau + \widehat{\mathbf{P}}_k^{\tau -1}\right)^{-1}\mathbf{H}_k^{\tau T}\mathcal{Q}_k^{-1}, \\ \widehat{\mathbf{x}}_k^{\tau+1} &= \widehat{\mathbf{x}}_k^\tau \boxplus \left[-\mathbf{K}\mathbf{z}_k^\tau - (\mathbf{I} - \mathbf{K}^\tau\mathbf{H}_k^\tau)(\mathbf{J}^\tau)^{-1}\left(\widehat{\mathbf{x}}_k^\tau \boxminus \widehat{\mathbf{x}}_k\right)\right]\end{aligned} \tag{24}$$

Repeat the above iterative steps until convergence ($\|\widehat{\mathbf{x}}_k^{\tau+1} \boxminus \widehat{\mathbf{x}}_k^\tau\|_2 < \varepsilon$), and the final covariance is updated to:

$$\widehat{\mathbf{P}}_k = (\mathbf{I} - \mathbf{K}^\tau\mathbf{H}_k^\tau)\widehat{\mathbf{P}}_k^\tau \tag{25}$$

Finally, we register the LiDAR point cloud into the global map based on the updated state.

IV. Experiments

In this section, we verify the performance of our proposed method through a series of experiments conducted on both public datasets and real-world environments. Among them, we take $\mathbf{n}_w = 3$. A comprehensive comparative analysis are performed against prominent state-of-the-art algorithms, including LOAM [5], LIO-SAM [12], LINS [15], and FAST-LIO2 [20]. All algorithms are implemented in C++ and executed using the robot operating system (ROS) [21] in Ubuntu Linux.

A. Public Dataset Experiments

In this experiment, we evaluate the accuracy of our method in large-scale indoor and outdoor environments on the M2DGR [22] dataset to verify the generalization of our proposed system. M2DGR dataset is an extensive SLAM dataset collected by a ground robot. This dataset contains a vast amount of perceptual information and relies on a 32-beam Velodyne LiDAR for collecting 3D point clouds of the environment. The robot is equipped with a 9-axis IMU called Handsfree A9, and all sensors are meticulously calibrated and synchronized. The dataset comprises 36 sequences, amounting to approximately 1 TB of data, captured across diverse scenarios, including both indoor and outdoor environments. To obtain accurate ground truth trajectories, it employs a GNSS-RTK suite and a motion-capture system with twelve highspeed tracking cameras (50 Hz).

For our evaluation, we select a range of scenarios for testing purposes and employ the Absolute Trajectory Error (ATE) [23] as the primary evaluation metric. The experimental results are presented in Table I. All algorithms are tested on an Intel i7 2.3 GHz processor-based computer.

In all sequences, SW-LIO exhibits superior performance in terms of drift compared to other LIO methods. Notably, SW-LIO consistently maintains minimal drift even in long-term sequences, as observed in street01 and street07. Furthermore, SW-LIO demonstrates enhanced robustness owing to the incorporation of sliding windows and supplementary residuals. As a result, SW-LIO achieves remarkable accuracy in challenging sequences featuring sharp turns (street07) and transitions between indoor and outdoor environments (door02).

B. Indoor Experiments

In this experimental study, we conduct rigorous evaluations within a challenging indoor setting. We use a handheld device equipped with a 32-beam Robosense LiDAR RS-Helios and an Xsense IMU MTi-680 G for data collection in a long straight corridor. The LiDAR operates at a scanning frequency of 10 Hz, while the IMU provides data at an acquisition rate of approximately 300 Hz. Given the inherent difficulty in obtaining ground truth values in this environment, we conduct the experiment by commencing from the starting point, traversing the corridor while holding the devices, and subsequently returning to the initial position. The total distance covered during this process is approximately 180 meters. To evaluate the accuracy of state estimation, we employ different methodologies and quantify the drift from the initial position. The outcomes of our experimental analysis are presented in Table II, Fig. 5 and Fig. 6.

It is evident from the results that both LINS and LIO-SAM encounter difficulties in localizing within the present environment, primarily attribute to algorithmic degradation and the substantial vibrations experienced by handheld devices in the corridor. Moreover, the high angular velocity during turns exacerbates the challenges associated with state estimation. As shown in Fig. 5(c) and Table II, the trajectories generated by other methods exhibit substantial drift, particularly in the Z direction. In contrast, our proposed method demonstrates the least amount of drift upon returning to the starting position. Additionally, Fig. 6 illustrates the superior mapping accuracy achieved by our method compared to FAST-LIO2.

In addition, to study the importance of each module in SW-LIO, we conduct an ablation study on our method by removing different modules, and the results are shown inTable II. Evidently, our method fails to attain optimal results upon the removal of certain modules, thereby highlighting the critical significance of each individual module.

TABLE I
ABSOLUTE TRAJECTORY ERROR(ATE/M) OF DIFFERENT APPROACHES ON M2DGR DATASET

Method/Sequence	Street01	Street06	Street07	Gate01	Hall02	Door02	Room01	Roomdark05
LOAM	2.349	0.685	10.93	0.498	0.83	0.225	0.156	0.291
LINS	1.781	1.722	12.02	0.390	0.274	0.213	1.196	0.703
LIO-SAM	2.566	0.429	10.22	0.717	0.278	0.197	0.342	0.284
FAST-LIO2	0.442	0.571	3.836	0.380	0.284	0.218	0.162	0.285
SW-LIO	**0.283**	**0.407**	**2.820**	**0.368**	**0.103**	**0.119**	**0.108**	**0.191**

The best performing methods are presented in bold.

TABLE II
DRIFT COMPARISON WHEN RETURNING TO THE STARTING POSITION IN INDOOR EXPERIMENTS

Method	LOAM	LINS	LIO-SAM	FAST-LIO2
Drift/m	5.344	✕	✕	1.647
Method	Ours-NG	Ours-NSW	Ours-NN	SW-LIO
Drift/m	0.575	1.102	0.885	**0.102**

✕ denotes that the SLAM system has failed.
Ours-NG: Our method without ground residual.
Ours-NSW: Our method without sliding window.
Ours-NN: Our method without new residuals.
The best performing methods are presented in bold.

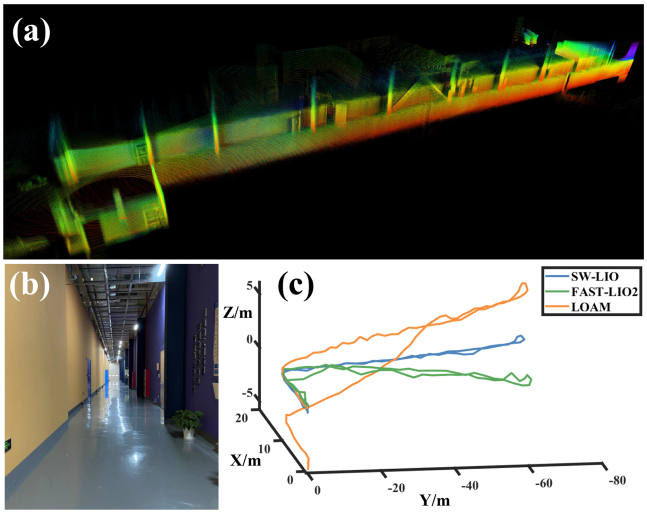

Fig. 5. Indoor experimental results in a long straight corridor. The data is collected through a handheld device. To quantify drift, the experiment starts and ends at the same place. (a) Mapping results of SW-LIO. (b) The corridor environment. (c) Comparison of trajectories with other methods.

C. Outdoor Experiments

To further test SW-LIO in practical scenarios, data collection is conducted using a hexapod robot named HexGuide, as shown in Fig. 7. HexGuide exhibits remarkable adaptability to different terrains and possesses the capability to autonomously guide visually impaired individuals. The robot is equipped with an RS-Helios 32-beam LiDAR and an Xsense IMU (inside the robot). Furthermore, a differential GPS real-time kinematics (RTK) system is installed on the robot to accurately obtain the ground-truth trajectory. The data collection is conducted within

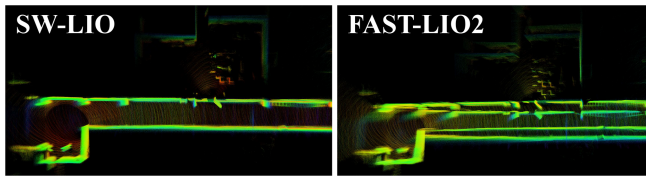

Fig. 6. Comparison of mapping results between SW-LIO and FAST-LIO2 in indoor experiments. It can be seen that our method has better mapping results.

TABLE III
ABSOLUTE TRAJECTORY ERROR(ATE/M) AND AVERAGE RUNNING TIME (MS) OF DIFFERENT APPROACHES IN OUTDOOR EXPERIMENTS

Method	LOAM	LINS	LIO-SAM	FAST-LIO2	SW-LIO
ATE/m	1.043	3.683	0.834	2.857	**0.451**
Time/ms	65.19	32.94	46.77	**18.84**	22.15

an outdoor industrial park, spanning a total path length of approximately 450 meters. The whole experiment is implemented on an Nvidia Jetson AGX computer equipped with this robot. The experimental results are shown in Figs. 7, 8, and Table III.

The performance of SW-LIO is apparent, as demonstrated by the mapping results (Fig. 7(c), (d)) with the highest accuracy (Table III). The moving of the hexapod robot introduces vibrations that result in significant fluctuations in IMU measurements. In addressing this challenge, our approach incorporates additional residuals to enhance the estimation of IMU bias and gravity vector, thereby improving the system's robustness. In addition, our method has a similar time consumption to FAST-LIO2, demonstrating the superiority of our method.

V. CONCLUSION

We propose SW-LIO, a tightly coupled LiDAR-inertial odometry based on the sliding window method. We present a rapid ground segmentation method for the separation of ground point clouds, followed by state estimation using an iterative error-state Kalman filter. To enhance the system's accuracy and robustness, a coupling relationship is established between the current state and previous frames using the sliding window method. Furthermore, several novel residuals are introduced to further improve the system's performance. Extensive testing in challenging scenarios, including public datasets, indoor corridors, and outdoor industrial parks, validates the efficacy of the proposed system. Comparative analysis against state-of-the-art methods

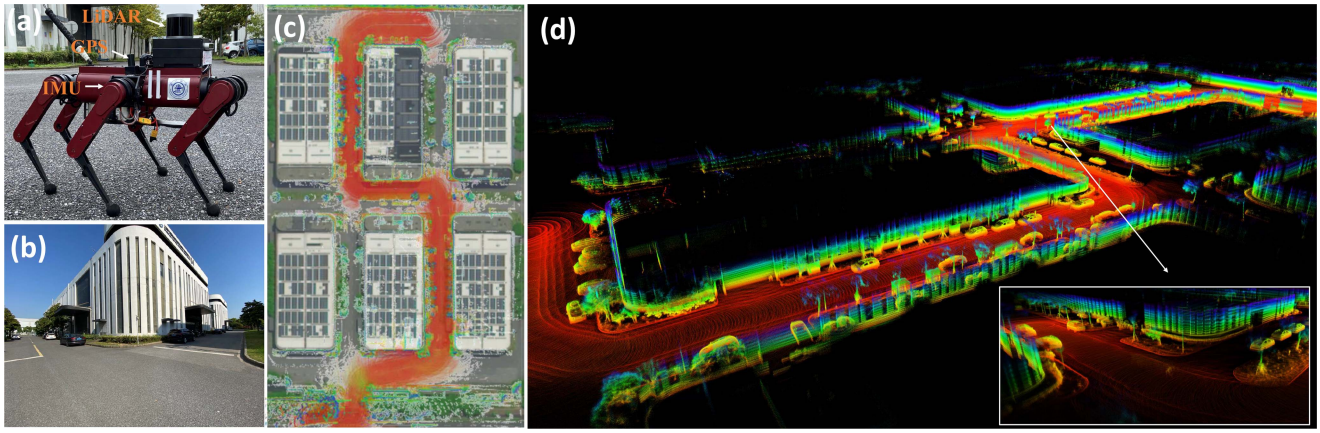

Fig. 7. Outdoor experimental results in an industrial park. Data collection is conducted using a hexapod robot equipped with an RS-Helios LiDAR and an Xsense IMU. (a) Hexapod robot used for experiment. (b) The industrial park. (c) 3D map aligning with satellite map. (d) Mapping results of SW-LIO.

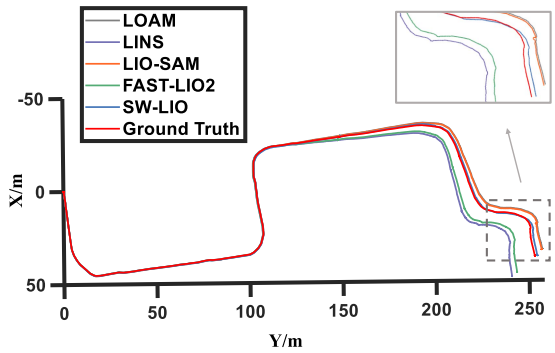

Fig. 8. Comparison of trajectories in outdoor experiments. Our method generates trajectories that are closer to ground truth.

demonstrates that SW-LIO achieves superior accuracy while maintaining similar time consumption.

REFERENCES

[1] H. Ye, Y. Chen, and M. Liu, "Tightly coupled 3D LiDAR inertial odometry and mapping," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 3144–3150.

[2] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized LiDAR odometry and mapping on variable terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 4758–4765.

[3] T. Li et al., "P 3-LOAM: PPP/LiDAR loosely coupled SLAM with accurate covariance estimation and robust RAIM in Urban canyon environment," *IEEE Sensors J.*, vol. 21, no. 5, pp. 6660–6671, Mar. 2021.

[4] H. Wang, C. Wang, C.-L. Chen, and L. Xie, "F-LOAM: Fast LiDAR odometry and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 4390–4396.

[5] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," *Robot.: Sci. Syst.*, vol. 2, no. 9, pp. 1–9, 2014.

[6] K. Li, M. Li, and U. D. Hanebeck, "Towards high-performance solid-state-LiDAR-inertial odometry and mapping," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 5167–5174, Jul. 2021.

[7] P. Shi, Z. Zhu, S. Sun, X. Zhao, and M. Tan, "Invariant extended Kalman filtering for tightly coupled LiDAR-inertial odometry and mapping," *IEEE/ASME Trans. Mechatron.*, vol. 28, no. 4, pp. 2213–2224, Aug. 2023.

[8] S. Hening, C. A. Ippolito, K. S. Krishnakumar, V. Stepanyan, and M. Teodorescu, "3D LiDAR SLAM integration with GPS/INS for UAVs in urban GPS-degraded environments," in *Proc. AIAA Inf. Syst.-AIAA Infotech, Aerosp.*, 2017, Art. no. 0448.

[9] M. Demir and K. Fujimura, "Robust localization with low-mounted multiple LiDARs in urban environments," in *Proc. IEEE Intell. Transp. Syst. Conf.*, 2019, pp. 3288–3293.

[10] W. Zhen, S. Zeng, and S. Soberer, "Robust localization and localizability estimation with a rotating laser scanner," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 6240–6245.

[11] H. Xiao, Y. Han, J. Zhao, J. Cui, L. Xiong, and Z. Yu, "LIO-vehicle: A tightly-coupled vehicle dynamics extension of LiDAR inertial odometry," *IEEE Robot. Automat. Lett.*, vol. 7, no. 1, pp. 446–453, Jan. 2022.

[12] 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 *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5135–5142.

[13] K. Koide, M. Yokozuka, S. Oishi, and A. Banno, "Globally consistent and tightly coupled 3D LiDAR inertial mapping," in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 5622–5628.

[14] 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.

[15] 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 *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 8899–8906.

[16] W. Xu and F. Zhang, "FAST-LIO: A fast, robust LiDAR-inertial odometry package by tightly-coupled iterated Kalman filter," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 3317–3324, Apr. 2021.

[17] D. He, W. Xu, N. Chen, F. Kong, C. Yuan, and F. Zhang, "Point-LIO: Robust high-bandwidth light detection and ranging inertial odometry," *Adv. Intell. Syst.*, vol. 5, no. 7, 2023, Art. no. 2200459.

[18] R. Schnabel, R. Wahl, and R. Klein, "Efficient RANSAC for point-cloud shape detection," *Comput. Graph. Forum*, vol. 26, no. 2, pp. 214–226, 2007.

[19] C. Hertzberg, R. Wagner, U. Frese, and L. Schröder, "Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds," *Inf. Fusion*, vol. 14, no. 1, pp. 57–77, 2013.

[20] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-LIO2: Fast direct LiDAR-inertial odometry," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2053–2073, Aug. 2022.

[21] M. Quigley et al., "ROS: An open-source robot operating system," in *Proc. ICRA Workshop Open Source Softw.*, 2009, Art. no. 5.

[22] J. Yin, A. Li, T. Li, W. Yu, and D. Zou, "M2DGR: A multi-sensor and multi-scenario SLAM dataset for ground robots," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 2266–2273, Apr. 2022.

[23] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 573–580.