DALIO: Degeneracy-Aware, Efficient LiDAR-Inertial Odometry with Adaptive Multi-Resolution Mapping

Haoda Zhu*, Fangcheng Zhu*, Yuhao Fang, Jiale Han, Yunfan Ren,
Siqi Liang, Longji Yin, Jie Mei, and Fu Zhang†

***Abstract*—This paper presents DALIO, a novel LiDAR-inertial odometry (LIO) framework that achieves robust localization across diverse environments while significantly reducing computational complexity. Embodying a "Computation on Demand" philosophy, DALIO introduces a degeneracy-aware batch selection module that dynamically adjusts the number of points used for scan registration based on real-time assessments of environmental constraints. Additionally, DALIO incorporates a degeneracy suppression method that filters out the influence of measurements along poorly constrained directions, thereby preventing noise in degenerate modes from corrupting the state update and improving overall estimation stability. Furthermore, DALIO employs MultiMap, an adaptive multi-resolution mapping strategy that preserves essential raw point data while selectively refining map resolution, effectively reducing memory usage and enabling accurate localization in both large-scale and small-scale settings. Extensive evaluations conducted on public datasets, multiple UAV platforms, and resource-constrained embedded systems demonstrate that DALIO achieves localization accuracy comparable to or better than state-of-the-art methods. At the same time, DALIO significantly reduces runtime by up to 65% while maintaining real-time performance across diverse hardware platforms.**

***Index Terms*—SLAM, LiDAR Perception, Localization, Sensor Fusion**

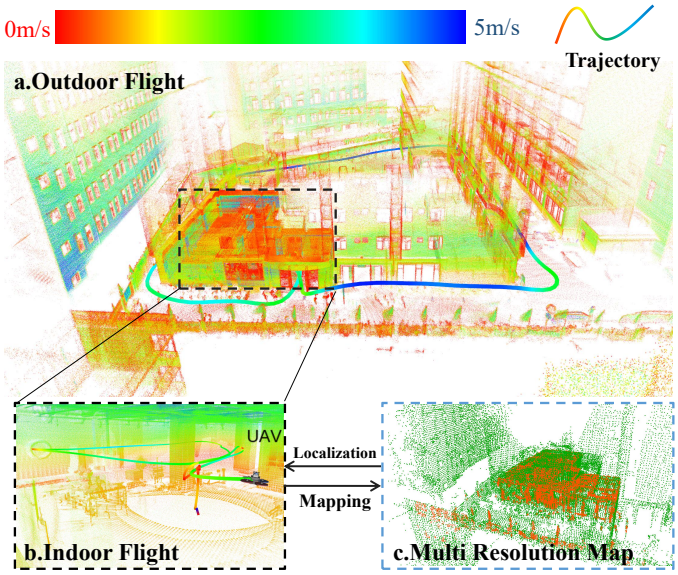

Fig. 1. a. Fast autonomous navigation using DALIO on a UAV, achieving stable mapping in both indoor and outdoor environments. The trajectory is color-coded by velocity. b. Indoor flight trajectory and mapping result. c. Multi-resolution map visualization: green points indicate low resolution, red points indicate high resolution.

I. Introduction

RECENT advancements in LiDAR technology and the decreasing cost of sensors [1] have significantly expanded the adoption of LiDAR-inertial Odometry beyond autonomous driving. Compact and affordable LiDAR sensors are now widely deployed on various mobile platforms, such as unmanned aerial vehicles (UAVs) [2] and handheld mapping equipment [3], enabling precise localization in GPS-denied environments. However, these platforms typically operate under stringent hardware constraints, including limited computational resources and memory capacity. At the same time, they must run multiple essential modules, such as perception [4], planning [5], and control [6] in parallel with localization. This makes it critical for LIO systems to minimize computational load while maintaining high localization accuracy to ensure system responsiveness and stability. Early methods like LOAM [7] and Loam-Livox [8] improved localization accuracy by employing feature extraction. However, their high computational demand limited their real-time applicability. To support real-time localization in complex and unstructured environments, recent approaches such as FAST-LIO2 [9] eliminate explicit feature extraction to reduce computational overhead. Moreover, FAST-LIO2 utilizes incremental KD-Tree [10] to enhance the efficiency of point cloud registration and map management. Additionally, FASTER-LIO [11] implements hash-based mapping structures to accelerate nearest-neighbor searches, further decreasing computational complexity. Nonetheless, several critical challenges remain unresolved:

(1) Current LIO methods still consume considerable computational resources, limiting onboard capacity for other essential modules. Reducing runtime and memory usage without compromising localization accuracy remains crucial for small robotic platforms.
(2) The limited field-of-view (FOV) and passive installation of LiDAR sensors often cause degeneracy in point cloud registration, leading to degraded localization accuracy.
(3) Existing LIO methods typically require meticulous parameter tuning due to variations between indoor and outdoor environments, limiting their general applicability and robustness. As shown in Fig. 1, such complex transitions remain a major challenge.

To address these challenges, we propose DALIO, a lightweight LIO framework optimized for low-cost embedded

*Equal contribution.
†Corresponding author: Fu Zhang (e-mail: fuzhang@hku.hk; phone: +852 3917 7909; address: HW 7–10, The University of Hong Kong, Hong Kong SAR, China).
Haoda Zhu, Fangcheng Zhu, Jiale Han, Yunfan Ren, Siqi Liang, Longji Yin, and Fu Zhang are with the Department of Mechanical Engineering, The University of Hong Kong, Hong Kong.
Yuhao Fang and Jie Mei are with the School of Intelligence Science and Engineering, Harbin Institute of Technology, Shenzhen, China.

hardware and robust performance across diverse environments. Fig. 1 illustrates DALIO deployed on a UAV, achieving stable real-time localization. The key contributions of this work include:

1) Degeneracy-Aware Batch Selector: We propose a novel degeneracy-aware batch selector that segments the point cloud into multiple batches and adaptively identifies the minimal number of batches necessary for stable localization. By selectively utilizing only essential points for scan matching, this method remarkably reduces computational costs while maintaining robustness under degenerate conditions.

2) Degeneracy Suppression Method: We introduce a novel degeneracy suppression method within the error state kalman filter (ESKF) framework that effectively mitigates the impact of LiDAR ranging noise along degenerate directions. This improves short-term stability and robustness when LiDAR measurements lack sufficient structure.

3) MultiMap: We develop an adaptive multi-resolution voxel map that adjusts resolution based on environmental geometry and prevents redundant updates in well-explored areas. This design maintains low memory usage and consistent performance with minimal parameter tuning across diverse environments.

4) Comprehensive Evaluation on Public Datasets and Real-world Platforms: We validate DALIO using public datasets and multiple real-world platforms, including UAV flights up to $8\,\mathrm{m/s}$. Results show that DALIO achieves state-of-the-art accuracy with substantially lower computational cost, enabling real-time performance on low-power embedded hardware.

II. Related Work

A. Efficient LiDAR-Inertial Odometry

Recent LiDAR-inertial odometry (LIO) research prioritizes computational efficiency through optimized mapping and registration. FAST-LIO2 introduces an incremental KDTree (iKDTree) to bypass explicit feature extraction, yet necessitates periodic rebalancing that incurs computational overhead. Hash-based alternatives like FASTER-LIO [11] and IG-LIO [12] mitigate rebalancing costs but remain sensitive to downsampling parameters. While LoLa-SLAM [13] accelerates processing through concurrent segmentation, this approach risks degeneracy in poorly constrained environments. Crucially, existing fixed-quantity point selection strategies cause excessive computation in feature-rich scenes and reduced robustness in degenerate ones. Addressing this, DALIO employs a degeneracy-aware batch selector to dynamically minimize the point batches required for robust localization, optimizing computational effort on resource-constrained platforms.

B. Degeneracy Detection in LIO

Degeneracy arises in geometrically featureless environments, such as corridors or open fields, where sparse constraints compromise localization. Various strategies have been proposed to mitigate this: Zhang et al. [14] integrated degeneracy detection into the optimization framework to improve consistency; X-ICP [15] analyzes point cloud structure during registration to enhance robustness; and Lim et al. [16] adapt voxelization parameters to prevent divergence in confined spaces. In multi-robot contexts, Zhu et al. [17, 18] utilized collaborative observations to compensate for single-UAV measurement degeneracy. Unlike these methods, which primarily utilize degeneracy detection to prevent failure, our approach leverages it to enhance efficiency. We reduce computational cost in non-degenerate scenarios and activate suppression mechanisms within the ESKF only when degeneracy is detected, ensuring both efficiency and robustness.

C. Multi-Resolution Map Representations

Multi-resolution mapping balances detail preservation with memory efficiency. Hierarchical structures have been applied in reconstruction (e.g., OGN [19], HSP [20]) and path planning (e.g., Rog-map [21]). In the context of LIO, VoxelMap [22] combines hash tables with octrees for scalable storage. However, its reliance on continuous plane fitting within each voxel demands dense point accumulation, incurring high computational costs that limit responsiveness in dynamic settings. Our proposed MultiMap addresses this by storing raw points and triggering subdivision via a single initial plane check. This "stop-update" strategy preserves essential geometric features while significantly reducing memory usage and processing time compared to continuous refinement approaches.

III. Framework Overview

An overview of the DALIO system is shown in Fig. 2. For each LiDAR frame, IMU measurements are first forward-propagated to obtain a prior state estimate, which is then used to compensate for motion distortion in the point cloud. The undistorted points are processed by a degeneracy-aware batch selector, which incrementally selects the minimal number of batches required for robust state estimation based on whether the point cloud exhibits degeneracy. If degeneracy persists even when using the full point cloud, a degeneracy suppression method is activated to reduce the impact of LiDAR noise along ill-constrained directions.

In MultiMap, both selected and unselected batches contribute to map updates. Voxels are dynamically classified as *Candidate*, *Planar*, or *Subdivided*, enabling adaptive resolution and efficient mapping across diverse environments.

TABLE I
Some Important Notations

Notation	Explanation
$\mathbf{x}, \widehat{\mathbf{x}}, \bar{\mathbf{x}}, \tilde{\mathbf{x}}$	The ground-truth, predicted, updated and error state value.
${}^{G}\mathbf{R}_{I}$	Orientation of the body frame I in the global frame G.
${}^{G}\mathbf{p}_{I}$	Position of the body frame I in the global frame G.
${}^{G}\mathbf{v}_{I}$	Velocity of the body frame I in the global frame G.
${}^{I}\mathbf{R}_{L}, {}^{I}\mathbf{p}_{L}, {}^{I}\mathbf{T}_{L}$	The extrinsic parameters from LiDAR to IMU, where ${}^{I}\mathbf{T}_{L} \in SE(3)$ is the 4×4 transformation matrix composed of ${}^{I}\mathbf{R}_{L}$ and ${}^{I}\mathbf{p}_{L}$.
${}^{G}\mathbf{g}$	The gravity vector in the global frame.
$\boldsymbol{\omega}_m, \mathbf{a}_m$	IMU gyroscope and accelerometer measurements.
$\mathbf{n}_g, \mathbf{n}_a$	White noise of gyroscope and accelerometer measurements.
$\mathbf{b}_g, \mathbf{b}_a$	Gyroscope and accelerometer bias.
$\mathbf{n}_{\mathbf{b}_g}, \mathbf{n}_{\mathbf{b}_a}$	Gaussian noise driving the random walk of $\mathbf{b}_g$ and $\mathbf{b}_a$.

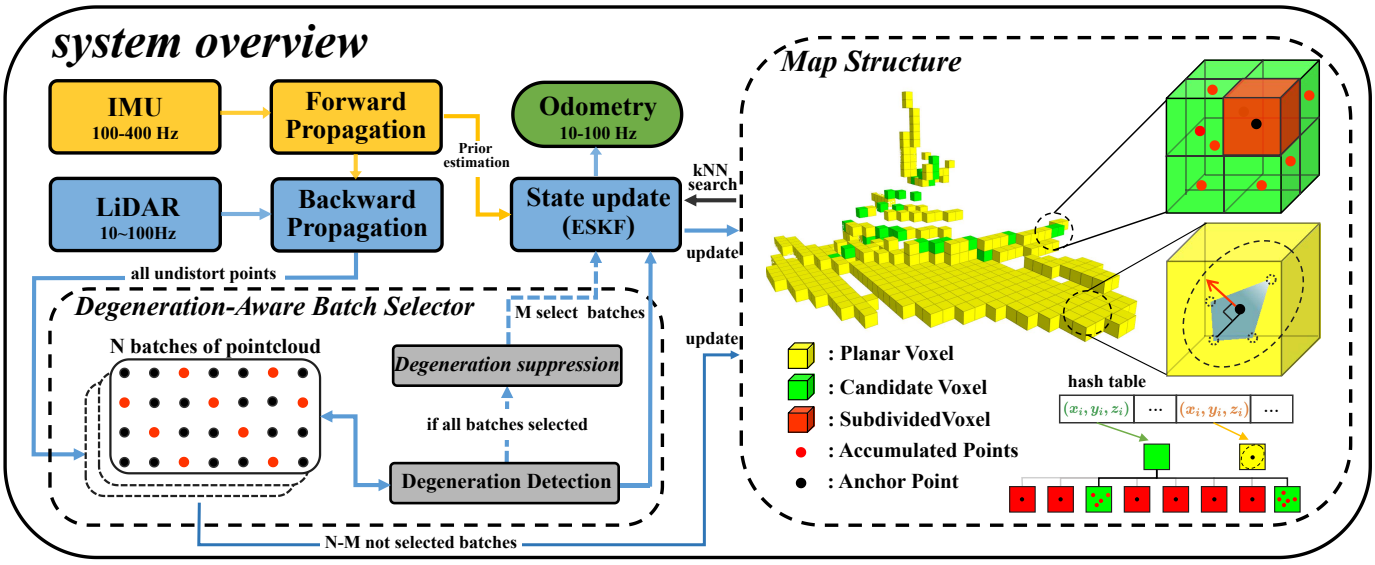

Fig. 2. Overview of the DALIO framework and MultiMap structure.

IV. State Estimation

DALIO employs a tightly-coupled Error State Kalman Filter (ESKF) to fuse LiDAR and IMU measurements. Utilizing the notations defined in Table I, this section briefly outlines the state formulation based on FAST-LIO [23].

A. State Representation and Propagation

We adopt the manifold notation $\boxplus/\boxminus$ from [24] for the state manifold $\mathcal{M} = SO(3) \times \mathbb{R}^{15}$. The state vector $\mathbf{x} \in \mathcal{M}$ is arranged as:

$$\mathbf{x} \triangleq [{}^{G}\mathbf{R}_I^T,\ {}^{G}\mathbf{p}_I^T,\ {}^{G}\mathbf{v}_I^T,\ \mathbf{b}_g^T,\ \mathbf{b}_a^T,\ {}^{G}\mathbf{g}^T]^T \tag{1}$$

Upon receiving an IMU measurement (with inputs $\boldsymbol{\omega}_m, \mathbf{a}_m$ and noise terms as listed in Table I), we perform forward propagation. The predicted state $\hat{\mathbf{x}}$ and covariance $\hat{\mathbf{P}}$ are updated via the standard discrete-time inertial kinematics:

$$\begin{aligned} \hat{\mathbf{x}}_{i+1} &= \hat{\mathbf{x}}_i \boxplus (\Delta t \cdot \mathbf{f}(\hat{\mathbf{x}}_i, \mathbf{u}_i, \mathbf{0})) \\ \hat{\mathbf{P}}_{i+1} &= \mathbf{F}_{\delta\hat{\mathbf{x}}} \hat{\mathbf{P}}_i \mathbf{F}_{\delta\hat{\mathbf{x}}}^T + \mathbf{F}_{\mathbf{w}} \mathbf{Q} \mathbf{F}_{\mathbf{w}}^T \end{aligned} \tag{2}$$

where Δt is the sampling interval. The specific forms of the continuous dynamics $\mathbf{f}(\cdot)$ and the Jacobians $\mathbf{F}_{\delta\hat{\mathbf{x}}}, \mathbf{F}_{\mathbf{w}}$ follow the standard derivation in [23].

B. State Update

In this section, we introduce the entire process of state update, including batch selection (Section IV-B1), degeneracy analysis (Section IV-B2), and modeling of LiDAR measurements (Section IV-B3).

1) Degeneracy-Aware Batch Selection: After motion compensation, we obtain an undistorted LiDAR scan. Since LIO may operate in both large-scale and small-scale environments, the number of points per frame can vary considerably. In large-scale scenes, even a relatively small subset of points may already provide sufficient constraints for localization due to the abundant LiDAR measurements. To realize "Computation on Demand", we propose the Degeneracy-Aware Batch Selector. Instead of processing the entire scan blindly, this module avoids utilizing unnecessary points by incrementally evaluating environmental constraints.

Algorithm 1: Degeneracy-Aware Batch Selector

Input: Pointcloud $\mathbf{P}$, batch number N, threshold ϵ_d
Output: Selected point set $\mathbf{S}$ for localization

1 Split $\mathbf{P}$ into N batches $\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_N$ of equal size;
2 Initialize $\mathbf{S} \leftarrow \emptyset$, $\mathbf{A} \leftarrow \mathbf{0}$, $\mathbf{A}_{\mathbf{obs}} \leftarrow \mathbf{0}$;
3 **for** $i = 1$ *to* N **do**
4 $\{\mathbf{H}_p\}_{p \in \mathbf{B}_i} = \texttt{FindNearestPlane}(\mathbf{B}_i)$;
5 $\mathbf{A}_{\mathbf{obs}} = \texttt{Compute}(\{\mathbf{H}_p\}_{p \in \mathbf{B}_i})$;
6 $\mathbf{A} \leftarrow \mathbf{A} + \mathbf{A}_{\mathbf{obs}}$;
7 $\lambda_d^{(i)} = \texttt{FindMinEigenvalue}(\texttt{SVD}(\mathbf{A}))$;
8 **if** $\lambda_d^{(i)} > \epsilon_d$ **then**
9 Set $\mathbf{S} \leftarrow \bigcup_{j=1}^{i} \mathbf{B}_j$;
10 **return** $\mathbf{S}$;
11 **end**
12 **end**
13 **return** $\bigcup_{i=1}^{N} \mathbf{B}_i$;

Alg. 1 performs point selection in an incremental manner. The undistorted point cloud is first downsampled and divided into N batches (line 1). For each batch i, nearest-neighbor searches are conducted to estimate local planes (line 4), and the corresponding observation matrix $\mathbf{A}_{\text{obs}}$ is constructed (line 5). This matrix is then added to the accumulated matrix $\mathbf{A}$ (line 6), and the smallest singular value of $\mathbf{A}$ is evaluated (line 7). If the singular value exceeds the threshold ϵ_d, the batches from 1 to i are selected (line 8). The process stops once sufficient observability is obtained or when all batches have been processed, ensuring that only the necessary points are used for reliable constraints.

The results of the nearest-neighbor searches and plane-fitting can be directly reused in the state update stage of the Kalman filter, thus no extra computations are introduced.

2) Degeneracy Analysis: Here we detailedly demonstrate the process of degeneracy analysis. To evaluate potential degeneracy, we extract the pose-related part of each measurement Jacobian matrix related to point ${}^{L}\mathbf{p}_j, j = 1, ..., m$:

$$\mathbf{H}_j^{\text{pose}} \triangleq (\mathbf{H}_j)_{1:6} \in \mathbb{R}^{1\times 6}, \tag{3}$$

where m denotes the number of planar feature points in the selected batches. Then we construct the observability matrix by stacking these rows across all m measurements:

$$\mathbf{H}_{\text{obs}} = \begin{bmatrix} \mathbf{H}_1^{\text{pose}} \\ \vdots \\ \mathbf{H}_m^{\text{pose}} \end{bmatrix} \in \mathbb{R}^{m\times 6}. \tag{4}$$

We then compute the eigendecomposition of the symmetric matrix:

$$\mathbf{A}_{\text{obs}} = \mathbf{H}_{\text{obs}}^T \mathbf{H}_{\text{obs}} = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^T, \tag{5}$$

where $\mathbf{V}$ contains the singular vectors associated with the six pose parameters, $\mathbf{\Sigma} = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_6)$ contains the eigenvalues, ordered as $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_6$. We denote the smallest eigenvalue as $\lambda_{\min} = \lambda_1$ and compare it with a predefined threshold ϵ_d. A frame is regarded as non-degenerate if $\lambda_{\min} \ge \epsilon_d$, indicating that the point cloud provides sufficient constraints for stable localization. Otherwise, if $\lambda_{\min} < \epsilon_d$, the frame is considered degenerate, in which case the degeneracy-aware batch selector activates to incorporate additional points for localization (see Section IV-B1).

3) Measurement Model: Assuming that after several rounds of batch selection, sufficient points in the current scan are accumulated and LiDAR degeneracy is not detected, we can proceed to state update. Denote by k the current scan index and by $\left\{{}^{L}\mathbf{p}_j,\ j = 1, \ldots, m\right\}$ the planar feature points accumulated by the batch selection, each expressed in the local LiDAR frame L at the scan end time. Due to LiDAR measurement noise, each measured point ${}^{L}\mathbf{p}_j$ is contaminated by noise ${}^{L}\mathbf{n}_j$, comprising range and beam-direction errors. Removing this noise yields the true point location in the LiDAR frame:

$${}^{L}\mathbf{p}_j^{\text{gt}} = {}^{L}\mathbf{p}_j + {}^{L}\mathbf{n}_j. \tag{6}$$

This true point, when projected into the global frame using the extrinsic ${}^{I}\mathbf{T}_L$, should lie exactly on a small plane in the map. Thus, the measurement model is

$$0 = \mathbf{u}_j^T \left({}^{G}\mathbf{T}_I\, {}^{I}\mathbf{T}_L \left({}^{L}\mathbf{p}_j + {}^{L}\mathbf{n}_j\right) - {}^{G}\mathbf{q}_j\right) \tag{7}$$

where $\mathbf{u}_j$ is the normal vector of the corresponding plane and ${}^{G}\mathbf{q}_j$ is a point on that plane.

Approximating the implicit measurement equation (7) by its first-order Taylor expansion at $\hat{\mathbf{x}}_k$ yields:

$$\begin{aligned} 0 &= h_j\big(\mathbf{x}_k, {}^{L}\mathbf{n}_j\big) \approx h_j\big(\hat{\mathbf{x}}_k, \mathbf{0}\big) + \mathbf{H}_j\tilde{\mathbf{x}}_k + r_j \\ &= z_j + \mathbf{H}_j\tilde{\mathbf{x}}_k + r_j, \end{aligned} \tag{8}$$

where $\mathbf{H}_j$ is the measurement Jacobian, which characterizes the local sensitivity of the measurement with respect to the state and determines whether the optimization problem is sufficiently constrained. Here, z_j denotes the constant term obtained by evaluating the measurement function at the linearization point $\hat{\mathbf{x}}_k$ without noise, while r_j represents the residual measurement noise after linearization, with the associated covariance matrix given by $\mathbf{R} = \mathbf{E}\{r_j r_j^T\}$. The explicit form of $\mathbf{H}_j$'s row vector is:

$$\mathbf{H}_j = \left[-\mathbf{u}_j^T {}^{I}\mathbf{R}_I \left({}^{I}\mathbf{R}_L\, {}^{L}\mathbf{p}_j + {}^{I}\mathbf{p}_L\right)^{\wedge},\ \mathbf{u}_j^T,\ \mathbf{0}_{1\times 12}\right] \tag{9}$$

with $(\cdot)^{\wedge}$ denoting the skew-symmetric matrix operator.

Then, the error state $\tilde{\mathbf{x}}_k$ is estimated through the Maximum A Posteriori (MAP) formulation

$$\min_{\tilde{\mathbf{x}}_k} \left(\|\mathbf{x}_k \boxminus \hat{\mathbf{x}}_k\|^2_{\hat{\mathbf{P}}_k^{-1}} + \sum_{j=1}^{m} \left\| z_j^k + \mathbf{H}_j^k \tilde{\mathbf{x}}_k \right\|^2_{\mathbf{R}^{-1}} \right), \tag{10}$$

where $\hat{\mathbf{x}}_k$ and $\hat{\mathbf{P}}_k$ denote the propagated state and covariance at time k, respectively. Here, $\bar{\mathbf{H}}_j^k$ represents the j-th Jacobian used in the MAP update.

C. Degeneracy Suppression in State Update

When the system remains degenerate after using all point batches, we apply an additional degeneracy suppression step directly to the measurement Jacobian to ensure robust state updates. $\mathbf{V} \in \mathbb{R}^{6\times 6}$ in (5) is the orthonormal matrix of singular vectors. We partition $\mathbf{V}$ into two orthogonal subspaces according to their observability strength:

$$\mathbf{V} = \begin{bmatrix} \mathbf{V}_f & \mathbf{V}_u \end{bmatrix}, \quad \mathbf{V}_f \in \mathbb{R}^{6\times r}, \quad \mathbf{V}_u \in \mathbb{R}^{6\times(6-r)}. \tag{11}$$

Here, $\mathbf{V}_f$ contains the singular vectors whose singular values are larger than the threshold ϵ_d and thus span the well-observed subspace, while $\mathbf{V}_u$ contains those below ϵ_d, corresponding to noise-dominated directions.

Since $\mathbf{V}$ is orthonormal, the identity

$$\mathbf{V}\mathbf{V}^T = \begin{bmatrix} \mathbf{V}_f & \mathbf{V}_u \end{bmatrix} \begin{bmatrix} \mathbf{V}_f^T \\ \mathbf{V}_u^T \end{bmatrix} = \mathbf{V}_f\mathbf{V}_f^T + \mathbf{V}_u\mathbf{V}_u^T = \mathbf{I}_6 \tag{12}$$

confirms that $\mathbf{V}_f$ and $\mathbf{V}_u$ form two mutually orthogonal and complete subspaces. Therefore, any pose-related Jacobian row vector $\mathbf{H}_j^{\text{pose}} \in \mathbb{R}^{1\times 6}$ can be uniquely decomposed as

$$\mathbf{H}_j^{\text{pose}} = \mathbf{d}_f\mathbf{V}_f^T + \mathbf{d}_u\mathbf{V}_u^T, \tag{13}$$

where $\mathbf{d}_f \in \mathbb{R}^{1\times r}$ and $\mathbf{d}_u \in \mathbb{R}^{1\times(6-r)}$ are the projections onto the informative and degenerate subspaces, respectively.

Since our objective is to retain only the informative component $\mathbf{d}_f\mathbf{V}_f$ and eliminate the noise-dominated part $\mathbf{d}_u\mathbf{V}_u$, this can be achieved by operating on $\mathbf{H}_j$ using the orthogonal projection operator $\mathbf{V}_f\mathbf{V}_f^T$ as

$$\begin{aligned} \bar{\mathbf{H}}_j^{\text{pose}} &= \mathbf{H}_j^{\text{pose}}\mathbf{V}_f\mathbf{V}_f^T = (\mathbf{d}_f\mathbf{V}_f^T + \mathbf{d}_u\mathbf{V}_u^T)\mathbf{V}_f\mathbf{V}_f^T \\ &= \mathbf{d}_f\mathbf{V}_f^T\mathbf{V}_f\mathbf{V}_f^T + \mathbf{d}_u\mathbf{V}_u^T\mathbf{V}_f\mathbf{V}_f^T = \mathbf{d}_f\mathbf{V}_f^T, \end{aligned} \tag{14}$$

and the complementary component is given by

$$\tilde{\mathbf{H}}_j^{\text{pose}} = \mathbf{H}_j^{\text{pose}} - \bar{\mathbf{H}}_j^{\text{pose}} = \mathbf{d}_u\mathbf{V}_u^T. \tag{15}$$

Applying this projection to all rows, we obtain the filtered observability matrix:

$$\bar{\mathbf{H}}_{\text{obs}} = \begin{bmatrix} \bar{\mathbf{H}}_1^{\text{pose}} \\ \vdots \\ \bar{\mathbf{H}}_m^{\text{pose}} \end{bmatrix} = \mathbf{H}_{\text{obs}}\, \mathbf{V}_f \mathbf{V}_f^T. \tag{16}$$

Let $\bar{\mathbf{H}}_j^{\text{pose}}$ replace the first six pose-related columns of $\mathbf{H}_j$, while the remaining non-pose columns remain unchanged. The complementary matrix corresponding to $\mathbf{H}_j$ is then given by

$$\tilde{\mathbf{H}}_j = \mathbf{H}_j - \bar{\mathbf{H}}_j = [\,\tilde{\mathbf{H}}_j^{\text{pose}},\, \mathbf{0}_{1\times 12}\,]. \tag{17}$$

To apply the filtered matrix $\bar{\mathbf{H}}_{\text{obs}}$ in the MAP formulation (10), the linearized residual equation (8) needs to be reformulated as

$$\begin{aligned} 0 &= z_j + \mathbf{H}_j \tilde{\mathbf{x}}_k + r_j \\ &= z_j + (\bar{\mathbf{H}}_j + \tilde{\mathbf{H}}_j)\tilde{\mathbf{x}}_k + r_j \\ &= z_j + \bar{\mathbf{H}}_j \tilde{\mathbf{x}}_k + \tilde{\mathbf{H}}_j \tilde{\mathbf{x}}_k + r_j \\ &= z_j + \bar{\mathbf{H}}_j \tilde{\mathbf{x}}_k + \tilde{r}_j, \end{aligned} \tag{18}$$

where $\tilde{r}_j$ denotes the newly modeled observation noise,

$$\tilde{r}_j = \tilde{\mathbf{H}}_j \tilde{\mathbf{x}}_k + r_j. \tag{19}$$

Since $\tilde{\mathbf{x}}_k$ and $\mathbf{r}_j$ are generally assumed to be uncorrelated, the covariance matrix of $\tilde{r}_j$ can be computed as

$$\begin{aligned} \tilde{\mathbf{R}}_j &= \mathbf{E}\{\tilde{r}_j \tilde{r}_j^T\} \\ &= \mathbf{E}\{(\tilde{\mathbf{H}}_j \tilde{\mathbf{x}}_k)(\tilde{\mathbf{H}}_j \tilde{\mathbf{x}}_k)^T\} + \mathbf{E}\{r_j r_j^T\} \\ &= \tilde{\mathbf{H}}_j \mathbf{E}\{\tilde{\mathbf{x}}_k \tilde{\mathbf{x}}_k^T\} \tilde{\mathbf{H}}_j^T + \mathbf{R} \\ &= \tilde{\mathbf{H}}_j \hat{\mathbf{P}}_k \tilde{\mathbf{H}}_j^T + \mathbf{R}. \end{aligned} \tag{20}$$

Then, the error state $\tilde{\mathbf{x}}_k$ can be estimated using the new MAP formulation

$$\min_{\tilde{\mathbf{x}}_k} \left(\|\mathbf{x}_k \boxminus \hat{\mathbf{x}}_k\|^2_{\hat{\mathbf{P}}_k^{-1}} + \sum_{j=1}^{m} \left\| z_j^k + \bar{\mathbf{H}}_j^k \tilde{\mathbf{x}}_k \right\|^2_{\tilde{\mathbf{R}}_j^{-1}} \right). \tag{21}$$

In practice, the optimization in (21) is nonlinear. It is solved using a Gauss–Newton method, which has been shown to be equivalent to an iterated Kalman filter [25]. To respect the manifold structure $\mathcal{M}$ of the state, each iteration parameterizes the update in the tangent space (i.e., the error state) around the current estimate. This inner loop continues until convergence. The converged state, denoted as $\bar{\mathbf{x}}_k$, and the corresponding covariance $\bar{\mathbf{P}}_k$ obtained from the Hessian of (21), are subsequently used for IMU propagation and for updating the global LiDAR map.

Crucially, using $\bar{\mathbf{H}}_{\text{obs}} = \mathbf{H}_{\text{obs}} \mathbf{V}_f \mathbf{V}_f^T$ restricts the measurement sensitivity to the informative subspace spanned by $\mathbf{V}_f$. As a result, the measurement update injects information only along well-observed directions and contributes no information in the degenerate subspace spanned by $\mathbf{V}_u$. Therefore, the posterior covariance contracts only in observable modes while remaining unchanged along degenerate directions—preventing spurious reduction of uncertainty and ensuring numerical stability of the filter.

D. Map Design

Due to the degeneracy-aware batch selector, the number of point clouds processed remains relatively constant regardless of whether the environment is small-scale or large-scale. To maintain stable performance and consistent computational efficiency across various scenarios, we propose a multi-resolution map structure called MultiMap.

1) Data Structure of MultiMap: The proposed map utilizes a hierarchical, hash-based storage approach. Specifically, point clouds are initially stored in sparse voxels, whose indices are hashed into a unordered map. Each voxel contains an octree structure with a predefined maximum depth, facilitating efficient spatial subdivision.

To effectively control memory usage, we introduce a Least Recently Used (LRU) caching algorithm. This algorithm maintains a fixed-length doubly-linked list to record the usage order of voxel nodes. Whenever a voxel node is accessed or updated, it is moved to the front of the list, indicating recent usage. If the number of voxel nodes exceeds the predefined capacity, the node at the tail of the list—the least recently used node—is removed from both the linked list and the hash table, thereby releasing its memory.

2) Initialization and Update of MultiMap: When a new frame of point cloud data arrives, each point is first mapped to the coarsest-resolution voxel. If the voxel does not exist, it is created and initialized. In either case, the point is inserted into the *Accumulated Points* set $\mathcal{P}_{acc}$.

Each voxel maintains an internal octree whose nodes are classified into three states: *Candidate Voxel*, *Planar Voxel*, or *Subdivided Voxel*, as shown in Fig. 2. At each octree level, a voxel is subdivided into eight child voxels. When at least four of these sub-voxels have accumulated points, a PCA is performed over all points in $\mathcal{P}_{acc}$ to assess planarity:

$$\mathbf{C} = \frac{1}{|\mathcal{P}_{acc}|} \sum_{p \in \mathcal{P}_{acc}} (p - \bar{p})(p - \bar{p})^\top,$$

where $\bar{p}$ denotes the centroid of $\mathcal{P}_{acc}$ and $\mathbf{C}$ is its sample covariance matrix. Let $\lambda_1 \geq \lambda_2 \geq \lambda_3$ be the eigenvalues of $\mathbf{C}$. If the smallest eigenvalue ratio $\lambda_3/(\lambda_1+\lambda_2+\lambda_3)$ is below a threshold τ_{pln}, the voxel is marked as a *Planar Voxel*, and only the point closest to the voxel center—the *Anchor Point* p_{anc}—is retained. Otherwise, the voxel is subdivided, and each child becomes a new *Candidate Voxel*.

This recursive process continues until reaching the maximum octree depth. At the final level, PCA is no longer applied. Once enough points are accumulated, the voxel is marked as a *Subdivided Voxel*, and only p_{anc} is kept.

This structure is robust to dynamic obstacles: transient objects usually fail to provide sufficient consistent points for subdivision and are eventually evicted by the LRU policy.

If a new point falls into a voxel already labeled as *Planar Voxel* or *Subdivided Voxel*, it is discarded immediately. This stop-update strategy preserves only one anchor point per voxel, thereby avoiding redundant computation and reducing runtime cost.

3) kNN Search in MultiMap: For k-nearest neighbor (kNN) search, each query point p is mapped to a voxel in a hash-indexed grid. We then gather points from that voxel and its

TABLE II
TIME EVALUATION (MS) AND RMSE (M) FOR NCLT AND NCD SEQUENCES

Seq.	DALIO			DALIO*		Fast-LIO2		Faster-LIO		VoxelMap		Dis. (km)
	Time	RMSE[1]	Batch Num[2]	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	
nclt 1	**10.58**	1.32	2.12	24.15	1.29	31.22	1.48	20.90	**1.28**	22.54	1.31	3.17
nclt 2	**10.13**	1.56	2.46	26.18	1.59	32.02	2.01	19.02	1.60	23.69	**1.46**	6.12
nclt 3	**11.40**	0.96	2.91	24.66	0.90	30.17	**0.87**	17.15	1.01	22.70	0.96	4.09
ncd 1	**16.20**	0.33	1.95	30.85	**0.32**	35.49	0.34	28.10	0.33	26.01	0.32	1.61
ncd 2	**17.34**	0.36	2.17	30.65	0.36	36.75	0.36	30.84	0.39	29.55	**0.32**	3.06
ncd 3	**16.08**	0.12	2.09	26.73	0.12	32.26	**0.11**	27.09	0.14	28.76	0.13	0.48

[1] RMSE represents the root mean square error between the estimated and ground-truth trajectories, considering both position and orientation.
[2] Batch Num refers to the average number of selected batches per frame in DALIO.

TABLE III
TIME EVALUATION (MS) AND E2E (M) FOR HKU, HKUST, AND M3DGR SEQUENCES

Seq.	DALIO			DALIO*		Fast-LIO2		Faster-LIO		VoxelMap		Dis. (km)
	Time	E2E[1]	Batch Num	Time	E2E	Time	E2E	Time	E2E	Time	E2E	
hku 1	**7.01**	**<0.05**	3.75	11.78	**<0.05**	12.61	**<0.05**	10.34	0.11	16.32	**<0.05**	0.35
hku 2	**8.47**	**<0.05**	3.01	14.34	**<0.05**	11.94	0.08	12.75	**<0.05**	15.03	**<0.05**	0.18
hku 3	**7.33**	**<0.05**	5.11	11.47	**<0.05**	13.13	1.96	11.82	2.01	15.13	0.10	1.04
hkust 1	**9.29**	**0.67**	2.91	15.41	0.92	16.95	3.34	14.87	3.97	24.16	5.01	1.32
hkust 2	**8.91**	1.32	2.55	14.98	**0.98**	17.17	2.97	15.39	2.78	24.14	3.43	1.52
hkust 3	**8.11**	**<0.05**	3.02	13.20	**<0.05**	15.25	**<0.05**	12.15	**<0.05**	23.95	**<0.05**	0.50
m3dgr 1	**4.82**	0.59	8.47	5.71	**0.55**	6.11	1.02	5.04	0.98	8.18	3.70	0.27
m3dgr 2	**4.39**	**0.45**	7.20	5.66	0.47	5.70	3.33	5.33	5.51	7.25	–[2]	0.26
m3dgr 3	**7.84**	**<0.05**	3.16	15.68	**<0.05**	20.67	**<0.05**	14.36	**<0.05**	29.21	**<0.05**	0.58

[1] E2E (end-to-end error) denotes the absolute position error between the start and end points of the trajectory, using position only.
[2] "–" means the algorithm failed in this sequence due to large drift.

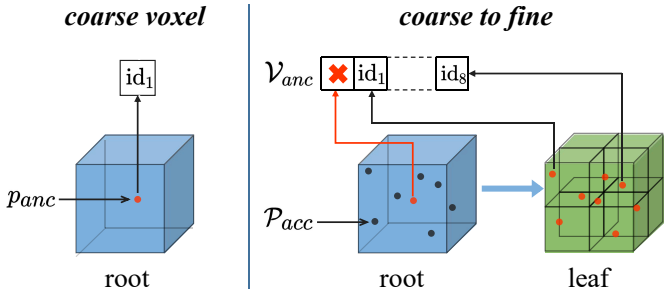

Fig. 3. Anchor Point vector $\mathcal{V}_{anc}$ stores direct references to anchor points within the voxel's octree, enabling fast kNN queries with minimal traversal.

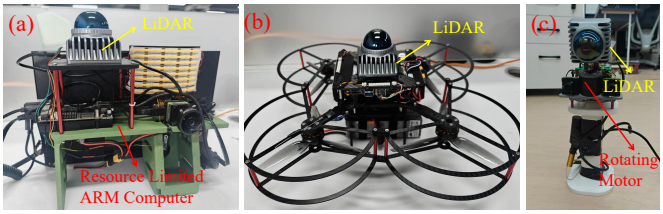

Fig. 4. Three platforms used in the experiment, handheld device with RK3588, UAV and rotating lidar device.

adjacent voxels, compute their distances to p, and return the K closest as the kNN result.

To reduce computation, each voxel maintains an auxiliary pointer vector $\mathcal{V}_{anc}$ at the root of the octree, shown in Fig. 3. This vector stores direct references to all Anchor Points within the voxel's octree nodes. By limiting neighbor search to these representative points, query-time complexity is reduced to approximately $O(1)$, shifting most computation to the map update stage.

V. EXPERIMENTS

In this section, we extensively evaluate the proposed DALIO framework on public datasets, as well as on handheld devices with limited computational resources, multiple UAV platforms and rotating lidar device, as illustrated in Fig. 4. The results demonstrate that DALIO is robust and adaptable across diverse scenarios and hardware platforms, while maintaining low computational overhead.

A. Evaluation on Public Datasets

In this section, we evaluate DALIO on five public datasets with diverse environment: HKU, HKUST (from R3LIVE [26]), NCD [27], NCLT [28], and M3DGR [29]. These datasets are selected to test the robustness of our method across a wide range of operating conditions. Specifically, HKU and HKUST involves indoor-outdoor transitions and multi-scale structures, making it suitable for evaluating multi-resolution mapping; NCLT and NCD includes long-range outdoor trajectories with dynamic urban scenes, serving as a benchmark for large-scale robustness and long-term consistency; and M3DGR covers a

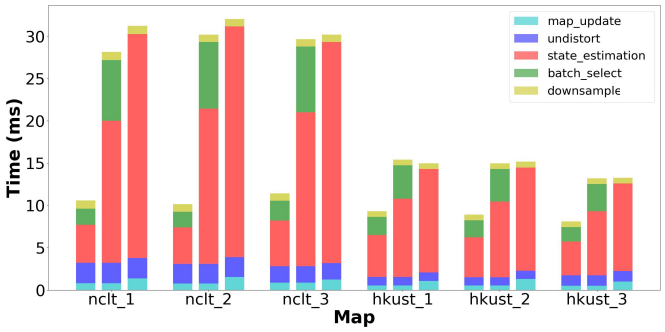

Fig. 5. Time consumption of each major module across different datasets. For each dataset, three columns represent DALIO, DALIO*, and FAST-LIO2, from left to right.

variety of degeneracy and GNSS-denied environments, such as corridors, making it ideal for testing degeneracy-aware localization.

We compare our method against several state-of-the-art LIO systems, including FAST-LIO2, FASTER-LIO, and VoxelMap. To highlight the contribution of our batch-based point cloud selection strategy, we also include a baseline variant named DALIO*, which disables the degeneracy-aware batch selector and uses all points for registration.

In our implementation, we use a multi-resolution map with resolutions of $0.5\,\mathrm{m}$, $0.25\,\mathrm{m}$, and $0.125\,\mathrm{m}$. These values were chosen so that the coarsest resolution ($0.5\,\mathrm{m}$) matches the map resolution used in other methods. To ensure a fair comparison, parameters with similar meanings across different methods were set to same values, while other settings followed each method's recommended configuration. All experiments were conducted on a system running Ubuntu 20.04 with an Intel i7-10875H CPU, with the same number of cores bound to each process and locked to the maximum frequency.

1) Efficiency Evaluation: As shown in Table II and Table III, DALIO consistently achieves the lowest computational cost among all compared methods. DALIO*, which disables the degeneracy-aware batch selector and uses all points for kNN search and plane fitting at every frame, results in significantly higher computation time. This highlights the effectiveness of the proposed batch selector in reducing processing cost.

Our MultiMap shares conceptual similarities with the voxel fitting strategy used in VoxelMap. However, MultiMap requires only a single plane-fitting step to determine whether a voxel should be subdivided, whereas VoxelMap repeatedly refines its map by performing plane fitting during subsequent updates. This repeated refinement leads to substantially higher computation time for VoxelMap.

Fig.5 further analyzes the runtime distribution across major LIO modules, including point cloud undistort, down sample, state estimation, and map update. Due to consistent parameter settings, most algorithms exhibit similar computational costs in point cloud undistort and down sample. Major differences appear in the state estimation and map update stages. In DALIO, the point cloud batching strategy reduces the number of points involved in nearest-neighbor search and plane fitting, which are typically the most computationally expensive operations.

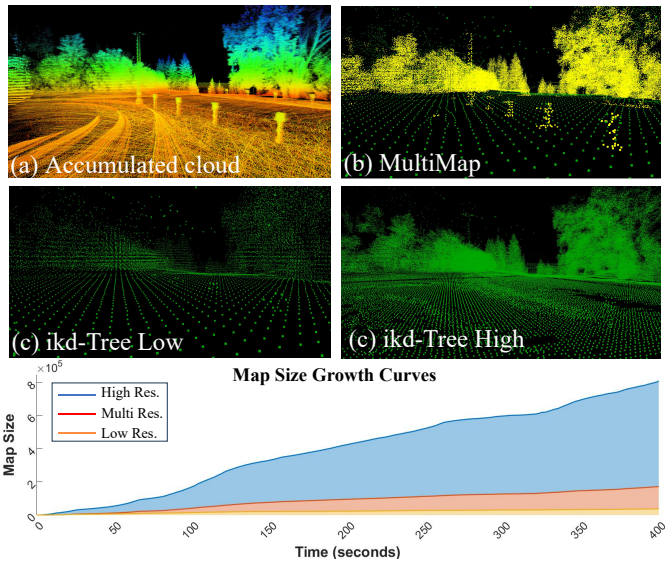

Fig. 6. Comparison of point cloud maps under different resolution settings and corresponding map size growth over time.

As a result, the runtime of the state estimation module is significantly reduced without sacrificing the accuracy of the estimation.

2) Accuracy and Robustness Evaluation: Table II reports results on the NCLT and NCD datasets, where ground truth is available and accuracy is evaluated using RMSE. Table III presents results on the HKU, HKUST, and M3DGR datasets, which lack ground truth but provide trajectories returning to the starting point, enabling end-to-end error evaluation. In outdoor scenes such as NCLT, DALIO shows no significant accuracy advantage over other methods, as the benefits of multi-resolution mapping are less evident in open environments. In contrast, DALIO achieves higher accuracy on the HKU, HKUST, and M3DGR datasets. This improvement is most evident in narrow indoor areas and in degenerate environments, such as the M3DGR sequence. In these cases, other methods may experience severe drift. Removing the batch-based point selection module yields similar accuracy but notably higher runtime, confirming the efficiency of the proposed strategy.

TABLE IV
COMPARISON RESULTS ON THE HKU AND HKUST DATASETS

End-to-End error (m) / Avg. Comp. Time (ms)			
Seq./Alg.	MultiMap	ikd-Tree High	ikd-Tree Low
hku 1	<**0.05** / **11.78**	<**0.05** / 15.17	<**0.05** / 12.61
hku 2	<**0.05** / 14.34	<**0.05** / 15.66	0.08 / **11.94**
hku 3	<**0.05** / **11.47**	1.23 / 17.84	1.96 / 13.13
hkust 1	**0.92** / **15.41**	1.85 / 18.09	3.34 / 16.95
hkust 2	**0.98** / **14.98**	4.67 / 20.83	2.97 / 17.17
hkust 3	<**0.05** / **13.20**	0.11 / 18.58	<**0.05** / 15.25

3) Ablation Study on Multi-Resolution Maps: To evaluate the impact of our multi-resolution mapping strategy, we conduct ablation experiments on three map structures while disabling the Degeneracy-Aware Batch Selector to isolate the mapping effect: (1) **MultiMap** ($0.5\,\mathrm{m}$–$0.25\,\mathrm{m}$–$0.125\,\mathrm{m}$), our proposed adaptive multi-resolution map; (2) **ikd-Tree**

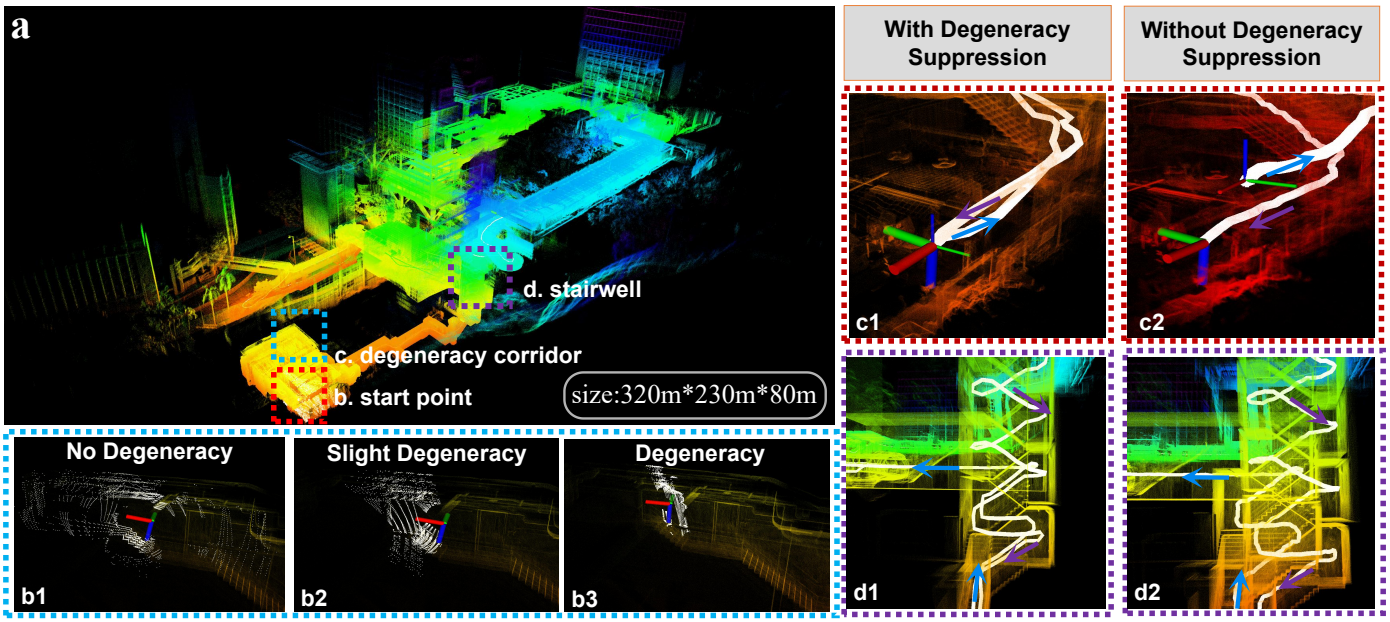

Fig. 7. Fig. a shows the mapping results of the rotating LiDAR, while Figs. b1, b2, and b3 display its point clouds in narrow channels—point cloud degeneracy occurs as the LiDAR rotates. Figs. c and d compare mapping results with and without degeneracy suppression: without suppression, the point clouds show layering at the stairs, and the odom does not return to the origin.

Low (0.5 m), an incremental KD-tree structure with a fixed resolution of 0.5 m; and (3) **ikd-Tree High** (0.125 m), the same structure with a higher resolution of 0.125 m. All other parameters are kept constant across these configurations.

As illustrated in Fig. 6, ikd-Tree High captures finer geometric details but incurs significantly higher memory usage and computational cost. In contrast, ikd-Tree Low reduces memory consumption but struggles to preserve small-scale structures, potentially degrading localization accuracy. MultiMap effectively balances these trade-offs by maintaining low resolution in structured regions while selectively refining resolution in unstructured regions, preserving critical features while minimizing memory overhead.

To accurately assess memory consumption, we count only the downsampled map points in ikd-Tree, whereas in MultiMap, both Anchor Points and temporarily stored Accumulated Points are included. The latter reflects actual memory usage, as Accumulated Points remain in the map until reaching the subdivision threshold. As shown by the map size growth curves in Fig. 6, ikd-Tree High exhibits the most rapid memory growth, ultimately consuming more than five times the storage of MultiMap.

To validate MultiMap's adaptability, we evaluate its performance on the HKU and HKUST datasets. As shown in Table IV, MultiMap achieves lower end-to-end error than both ikd-Tree Low and ikd-Tree High. This indicates that increasing map resolution alone does not necessarily reduce localization error. Instead, limiting resolution in large planar regions helps suppress sensor noise while maintaining higher resolution in complex areas, ultimately leading to improved localization robustness.

B. Evaluation on Handheld Devices

We evaluate DALIO on two distinct handheld platforms: (1) an affordable ARM-based device with limited computational power, and (2) a custom rotating LiDAR system designed to expand the field of view (FOV) for improved mapping.

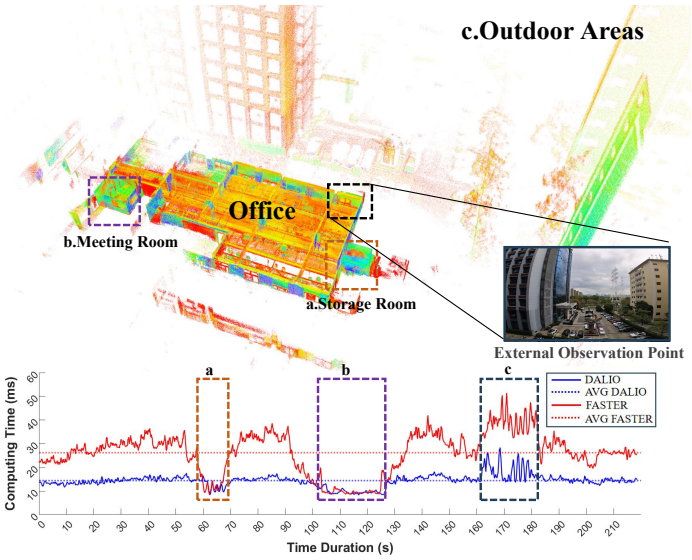

Fig. 8. Running results of the algorithm on the handheld device, including mapping outputs (top) and runtime over time (bottom). Regions (a), (b), and (c) indicate different segments of the runtime curve corresponding to the device operating in various environments.

1) ARM-Based Handheld Device: We validated DALIO on an ARM-based handheld device equipped with an RK3588 computing unit (approximately $100). A Livox Mid-360 LiDAR was used for sensing, while the onboard camera was used solely for recording. The system was tested across various real-world environments, including an office (medium scale), a small meeting room and a storage room (small scale), and large outdoor areas visible through windows. As shown in Fig. 8, DALIO maintained consistently low computational cost, with

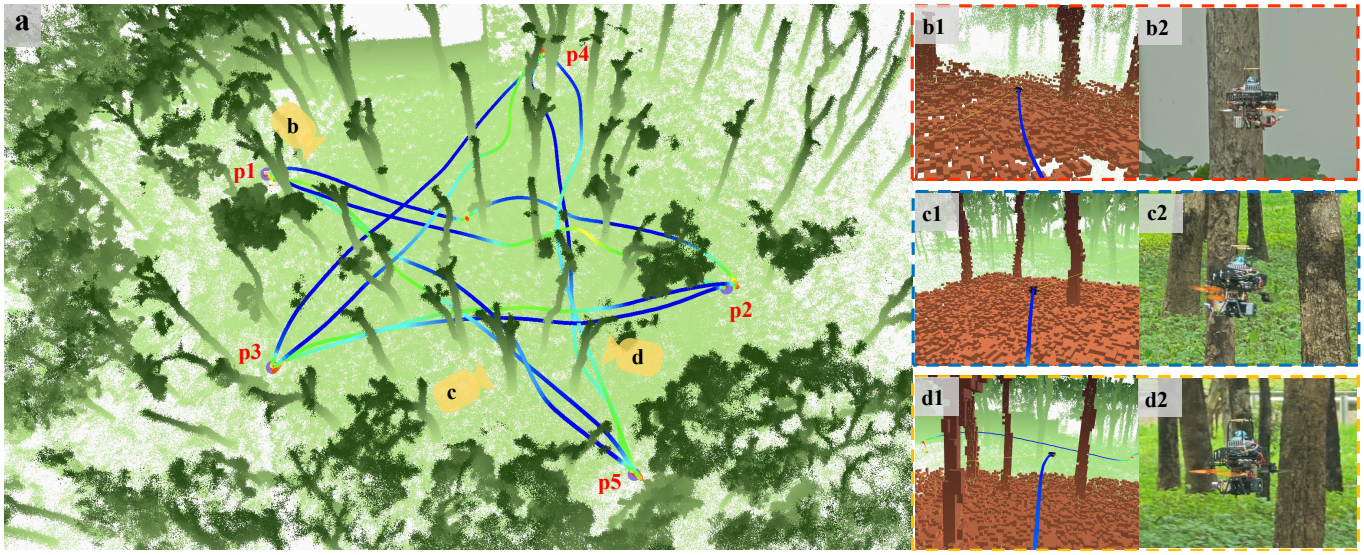

Fig. 9. Fig. a shows the complete point cloud map, with the trajectory color-coded by velocity. b, c, and d present multiple flight perspectives. b1, c1, and d1 show the occupancy grid maps used for real-time obstacle avoidance, while b2, c2, and d2 display the real world UAV flight trajectories captured by a camera.

minimal runtime variation when transitioning from small to large-scale environments. The generated maps were clear and globally consistent, and the end-to-end localization error upon returning to the starting point was consistently below $0.05\,\mathrm{m}$.

We also evaluated FASTER-LIO on the same device using identical parameters. As shown in Fig. 8, DALIO consistently outperformed FASTER-LIO in computational efficiency while achieving better localization accuracy. In the small meeting room and storage room (Figs. 8a and 8b), both methods had similar runtime due to the limited number of downsampled points, with DALIO using nearly all available points to maintain robustness. In contrast, in the office and large outdoor areas, DALIO benefited from its point cloud batching strategy, requiring only a subset of points for stable localization, thus significantly reducing runtime. On average, DALIO achieved a runtime of $14.26\,\mathrm{ms}$ per frame, compared to $27.34\,\mathrm{ms}$ for FASTER-LIO.

2) Rotating LiDAR Device: To further evaluate DALIO under large-scale and challenging environments, we collected a dataset using a rotating LiDAR device that continuously spins at $1\,\mathrm{Hz}$. This setup, based on a Livox Mid-360 LiDAR and an Intel N305 computing unit, produces full-surround point clouds but also introduces additional challenges: the LiDAR orientation is uncontrollable, and scans may temporarily align with degenerate directions, especially in narrow spaces (Fig. 7a1–a3). Fig. 4c illustrates the device configuration. The dataset covers a $1021\,\mathrm{m}$ trajectory across diverse environments, including degenerate corridors, staircases, outdoor building areas, and forested scenes. In these scenarios, DALIO's degeneracy suppression module effectively rejects unreliable LiDAR constraints, maintaining stable localization even during short-term degeneracy.

Beyond local degeneracy, this dataset also features drastic environmental variation. Fixed-resolution maps fail to provide stable localization across all conditions, whereas our MultiMap adapts its resolution online, enabling a single parameter set to handle large changes in scene scale and geometry.

Table V reports the quantitative results on this dataset. DALIO achieves the lowest end-to-end error and is the only method that nearly returns to the starting point. Without degeneracy suppression (DALIO$^\wedge$), localization accuracy degrades noticeably, highlighting the importance of this module. Compared to FAST-LIO, FASTER-LIO, and VoxelMap, DALIO not only improves robustness in degenerate cases but also maintains competitive runtime.

TABLE V
COMPARISON RESULTS ON THE ROTATING LIDAR DATASET.

End-to-End error (m) / Avg. Comp. Time (ms)					
Alg.	DALIO	DALIO$^\wedge$	FAST-LIO2	FASTER-LIO	VoxelMap
Rotating	**0.12** / 12.15	2.08 / **12.02**	5.16 / 24.71	4.91 / 22.45	– / 28.67

[1] "–" means the algorithm failed in this sequence due to large drift.

C. Evaluation on UAV Navigation

In the UAV experiments, we present two application scenarios: (1) trajectory tracking in a mixed indoor–outdoor environment, and (2) high-speed maneuvering in a forest.

1) Trajectory Tracking in Mixed Indoor–Outdoor Environments: To verify DALIO's stability in high-speed UAV navigation through both indoor and outdoor areas, we deploy it onboard a UAV and integrate it with Super [30] as the online planning module.

We conduct three trajectory tracking experiments with progressively increasing flight speeds. Fig. 1 illustrates the highest-speed case, Test 3, where the UAV reaches up to $5\,\mathrm{m/s}$. The trajectory is color-coded by velocity, with a total flight distance of $181\,\mathrm{m}$. A set of predefined waypoints guides the UAV while enabling real-time obstacle avoidance. The trajectory begins indoors, transitions to a large outdoor loop, and returns indoors, demonstrating DALIO's stable and real-time localization performance across diverse flight conditions. As shown in Fig. 1, the generated map remains consistent

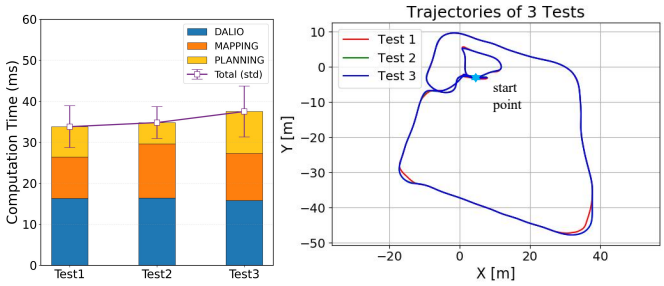

Fig. 10. Left: Computation time breakdown of DALIO, mapping, and planning for Test 1–3. The purple markers indicate the mean total time, with error bars showing the standard deviation. Right: Flight trajectories of Test 1–3, starting from the same takeoff point.

throughout the flight. In particular, Fig. 1c highlights the MultiMap strategy, where high-resolution red points capture detailed indoor structures, and low-resolution green points represent large-scale outdoor mapping.

Fig. 10 further summarizes the three experiments from both efficiency and repeatability perspectives. The left plot reports the computation time of DALIO, mapping(utilizing raycasting to remove dynamic obstacles), and planning, all measured onboard the UAV during real flight experiments, reflecting the actual runtime under realistic sensing, communication, and onboard computing constraints. The results demonstrate that the integrated system consistently maintains stable real-time performance in a real-world UAV operating environment. Experiments were conducted at different times of the day, covering both low-traffic conditions and cases where pedestrians were present. In all flights, the UAV took off from the same start point and eventually returned to it, yielding consistently low end-to-end errors. Because the obstacle configuration varied across tests, the trajectories in the right plot of Fig. 10 show slight differences, demonstrating DALIO's robustness in the presence of dynamic obstacles.

2) High-Speed Maneuvering in a Forest: To evaluate DALIO's localization performance during aggressive maneuvers, we conduct a high-speed flight in a forest. The UAV takes off within a $20\,\mathrm{m} \times 20\,\mathrm{m}$ area and flies between five waypoints arranged in a pentagon-shaped pattern. The UAV departs from p_1, passes sequentially through p_2 to p_5, repeats the loop once, and finally lands back at p_1. The maximum flight speed reaches $8\,\mathrm{m/s}$, the peak acceleration reaches $5g$, and the maximum angular velocity reaches $10\,\mathrm{rad/s}$. DALIO provides accurate, real-time odometry throughout the maneuvers. To assess loop-closure consistency, the UAV completes two full circuits of the trajectory, as shown in Fig. 9a. The figure also shows that UAV speed varies significantly, as it must avoid obstacles that suddenly appear in the FOV during high-speed flight without a prior map. DALIO maintains stable localization even under these highly dynamic conditions. Moreover, in such geometrically sparse environments, DALIO continues to achieve accurate localization with low computational cost, leaving sufficient resources for perception and planning modules.

VI. Conclusion and Future Work

This paper presented DALIO, a LiDAR–inertial odometry framework with a degeneracy-aware batch selector, a degeneracy suppression strategy, and a multi-resolution map. Experiments on diverse platforms demonstrate that DALIO achieves significantly lower runtime than existing methods while maintaining robust accuracy. Future work will focus on handling dynamic environments, such as filtering moving objects during mapping, and extending the framework with semantic and global optimization modules for improved long-term consistency.

References

[1] Z. Liu, F. Zhang, and X. Hong, "Low-cost retina-like robotic lidars based on incommensurable scanning," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 1, pp. 58–68, 2022.

[2] N. Chen, F. Kong, W. Xu, Y. Cai, H. Li, D. He, Y. Qin, and F. Zhang, "A self-rotating, single-actuated uav with extended sensor field of view for autonomous navigation," *Science Robotics*, vol. 8, no. 76, p. eade4538, 2023. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.ade4538

[3] C. Zheng, W. Xu, Z. Zou, T. Hua, C. Yuan, D. He, B. Zhou, Z. Liu, J. Lin, F. Zhu, Y. Ren, R. Wang, F. Meng, and F. Zhang, "Fast-livo2: Fast, direct lidar–inertial–visual odometry," *IEEE Transactions on Robotics*, vol. 41, pp. 326–346, 2025.

[4] Y. Cai, F. Kong, Y. Ren, F. Zhu, J. Lin, and F. Zhang, "Occupancy grid mapping without ray-casting for high-resolution lidar sensors," *IEEE Transactions on Robotics*, vol. 40, pp. 172–192, 2024.

[5] Y. Ren, S. Liang, F. Zhu, G. Lu, and F. Zhang, "Online whole-body motion planning for quadrotor using multi-resolution search," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 1594–1600.

[6] W. Liu, Y. Ren, and F. Zhang, "Integrated planning and control for quadrotor navigation in presence of suddenly appearing objects and disturbances," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 899–906, 2024.

[7] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time," in *Robotics: Science and Systems*, vol. 2, no. 9, 2014.

[8] 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)*. IEEE, 2020, pp. 3126–3131.

[9] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, 2022.

[10] Y. Cai, W. Xu, and F. Zhang, "ikd-tree: An incremental kd tree for robotic applications," *arXiv preprint arXiv:2102.10808*, 2021.

[11] 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 Robotics and Automation Letters*, vol. 7, no. 2, pp. 4861–4868, 2022.

[12] Z. Chen, Y. Xu, S. Yuan, and L. Xie, "ig-lio: An incremental gicp-based tightly-coupled lidar-inertial odometry," *IEEE Robotics and Automation Letters*, vol. 9, no. 2, pp. 1883–1890, 2024.

[13] M. Karimi, M. Oelsch, O. Stengel, E. Babaians, and E. Steinbach, "Lola-slam: Low-latency lidar slam using continuous scan slicing," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2248–2255, 2021.

[14] J. Zhang, M. Kaess, and S. Singh, "On degeneracy of optimization-based state estimation problems," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 809–816.

[15] T. Tuna, J. Nubert, Y. Nava, S. Khattak, and M. Hutter, "X-icp: Localizability-aware lidar registration for robust localization in extreme environments," *IEEE Transactions on Robotics*, vol. 40, pp. 452–471, 2024.

[16] H. Lim, D. Kim, B. Kim, and H. Myung, "Adalio: Robust adaptive lidar-inertial odometry in degenerate indoor environments," in *2023 20th International Conference on Ubiquitous Robots (UR)*, 2023, pp. 48–53.

[17] F. Zhu, Y. Ren, F. Kong, H. Wu, S. Liang, N. Chen, W. Xu, and F. Zhang, "Swarm-lio: Decentralized swarm lidar-inertial odometry," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 3254–3260.

[18] F. Zhu, Y. Ren, L. Yin, F. Kong, Q. Liu, R. Xue, W. Liu, Y. Cai, G. Lu, H. Li, and F. Zhang, "Swarm-lio2: Decentralized efficient lidar-inertial odometry for aerial swarm systems," *IEEE Transactions on Robotics*, vol. 41, pp. 960–981, 2025.
[19] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2107–2115.
[20] C. Häne, S. Tulsiani, and J. Malik, "Hierarchical surface prediction for 3d object reconstruction," in *2017 International Conference on 3D Vision (3DV)*, 2017, pp. 412–420.
[21] Y. Ren, Y. Cai, F. Zhu, S. Liang, and F. Zhang, "Rog-map: An efficient robocentric occupancy grid map for large-scene and high-resolution lidar-based motion planning," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 8119–8125.
[22] C. Yuan, W. Xu, X. Liu, X. Hong, and F. Zhang, "Efficient and probabilistic adaptive voxel mapping for accurate online lidar odometry," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8518–8525, 2022.
[23] W. Xu and F. Zhang, "Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3317–3324, 2021.
[24] C. Hertzberg, R. Wagner, U. Frese, and L. Schröder, "Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds," *Information Fusion*, vol. 14, no. 1, pp. 57–77, 2013.
[25] B. Bell and F. Cathey, "The iterated kalman filter update as a gauss-newton method," *IEEE Transactions on Automatic Control*, vol. 38, no. 2, pp. 294–297, 1993.
[26] J. Lin and F. Zhang, "R^3live: 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)*. IEEE, 2022, pp. 10 672–10 678.
[27] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon, "The newer college dataset: Handheld lidar, inertial and vision with ground truth," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 4353–4360.
[28] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "University of Michigan North Campus long-term vision and lidar dataset," *International Journal of Robotics Research*, vol. 35, no. 9, pp. 1023–1035, 2015.
[29] D. Zhang, J. Zhang, Y. Sun, T. Li, H. Yin, H. Xie, and J. Yin, "Towards robust sensor-fusion ground slam: A comprehensive benchmark and a resilient framework," *arXiv preprint arXiv:2507.08364*, 2025.
[30] Y. Ren, F. Zhu, G. Lu, Y. Cai, L. Yin, F. Kong, J. Lin, N. Chen, and F. Zhang, "Safety-assured high-speed navigation for mavs," *Science Robotics*, vol. 10, no. 98, p. eado6187, 2025. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.ado6187

Haoda Zhu received the B.E. degree in automation in 2024 from the School of Mechanical Engineering and Automation, Harbin Institute of Technology, Shenzhen, China. He is currently working toward the Ph.D. degree in mechanical engineering at the University of Hong Kong (HKU), Hong Kong SAR, China.

His research interests include robotics, LiDAR-based simultaneous localization and mapping (SLAM), and sensor fusion.

Fangcheng Zhu received the B.E. degree in automation in 2021 from the School of Mechanical Engineering and Automation, Harbin Institute of Technology, Shenzhen, China. He is currently a Ph.D. candidate in robotics with the Department of Mechanical Engineering, University of Hong Kong (HKU), Hong Kong SAR, China.

His research interests include robotics and aerial swarm systems, with a focus on LiDAR-based simultaneous localization and mapping (SLAM) and sensor calibration.

Yuhao Fang received the B.E. degree from the Harbin Institute of Technology, Shenzhen, China. He is currently working toward the M.Sc. degree with the School of Intelligence Science and Engineering, Harbin Institute of Technology, Shenzhen, China.

His research interests include computer vision, robotics, and SLAM.

Jiale Han received the M.Sc. degree from Shanghai Jiao Tong University (SJTU), Shanghai, China. He is currently with the Department of Mechanical Engineering, The University of Hong Kong (HKU), Hong Kong SAR, China.

His research interests include autonomous navigation and robotic perception.

Yunfan Ren received the B.Eng. degree in Automation from Harbin Institute of Technology, China, in 2021. He is currently a Ph.D. Candidate with the Department of Mechanical Engineering at The University of Hong Kong, where he is a member of the MaRS Lab.

His research interests include autonomous navigation, UAV planning, optimal control, and swarm system autonomy. He was recognized as an Outstanding Navigation Paper Finalist at ICRA 2023 and as both the Best Overall and Best Student Paper Finalist at IROS 2023. He also serves as a reviewer for leading conferences and journals, including IEEE Robotics and Automation Letters (RAL), IROS, and ICRA.

Siqi Liang received the B.E. degree in automation in 2023 from the School of Mechanical Engineering and Automation, Harbin Institute of Technology, Shenzhen, China. He is currently working toward the Ph.D. degree in mechanical engineering at the University of Hong Kong (HKU), Hong Kong SAR, China.

His research interests include robotics, state estimation, and LiDAR SLAM.

Longji Yin received the B.Eng. degree in automation and the B.A. degree in English literature from Zhejiang University, Zhejiang, China, in 2019 and the M.Sc. degree in robotics from the Johns Hopkins University, Maryland, U.S. in 2021. He is currently working toward the Ph.D. degree in mechanical engineering at the University of Hong Kong, Hong Kong SAR, China.

His research interests include motion planning and control for autonomous aerial robots.

Jie Mei is currently a Professor with the School of Intelligence Science and Engineering, Harbin Institute of Technology, Shenzhen, China.

His research interests include nonlinear control, distributed control of multi-agent systems, and their applications in robotics.

Fu Zhang received the B.E. degree in automation from the University of Science and Technology of China (USTC), Hefei, China, in 2011 and the Ph.D. degree in controls from the University of California at Berkeley, Berkeley, CA, USA, in 2015.

He joined the Department of Mechanical Engineering, University of Hong Kong (HKU), Hong Kong, as an Assistant Professor in 2018. He was promoted to tenured associate professor at the University of Hong Kong (HKU) in 2024. His current research interests include robotics and controls, with a focus on UAV design, navigation, control, and LiDAR-based SLAM.