Swarm-LIO2: Decentralized Efficient LiDAR-Inertial Odometry for Aerial Swarm Systems

Fangcheng Zhu, Yunfan Ren, Longji Yin, Fanze Kong, Qingbo Liu, Ruize Xue, Wenyi Liu, Yixi Cai, Guozheng Lu, Haotian Li, and Fu Zhang

***Abstract*—Aerial swarm systems possess immense potential in various aspects, such as cooperative exploration, target tracking, and search and rescue. Efficient accurate self- and mutual state estimation are the critical preconditions for completing these swarm tasks, which remain challenging research topics. This article proposes Swarm-LIO2, a fully decentralized, plug-and-play, computationally efficient, and bandwidth-efficient light detection and ranging (LiDAR)-inertial odometry for aerial swarm systems. Swarm-LIO2 uses a decentralized plug-and-play network as the communication infrastructure. Only bandwidth-efficient and low-dimensional information is exchanged, including identity, ego state, mutual observation measurements, and global extrinsic transformations. To support the plug and play of new teammate participants, Swarm-LIO2 detects potential teammate autonomous aerial vehicles (AAVs) and initializes the temporal offset and global extrinsic transformation all automatically. To enhance the initialization efficiency, novel reflectivity-based AAV detection, trajectory matching, and factor graph optimization methods are proposed. For state estimation, Swarm-LIO2 fuses LiDAR, inertial measurement units, and mutual observation measurements within an efficient error state iterated Kalman filter (ESIKF) framework, with careful compensation of temporal delay and modeling of measurements to enhance the accuracy and consistency. Moreover, the proposed ESIKF framework leverages the global extrinsic for ego state estimation in the case of LiDAR degeneration or refines the global extrinsic along with the ego state estimation otherwise. To enhance the scalability, Swarm-LIO2 introduces a novel marginalization method in the ESIKF, which prevents the growth of computational time with swarm size. Extensive simulation and real-world experiments demonstrate the broad adaptability to large-scale aerial swarm systems and complicated scenarios, including GPS-denied scenes and degenerated scenes for cameras or LiDARs. The experimental results showcase the centimeter-level localization accuracy, which outperforms other state-of-the-art LiDAR-inertial odometry for a single-AAV system. Furthermore, diverse applications demonstrate the potential of Swarm-LIO2 to serve as a reliable infrastructure for various aerial swarm missions.**

Received 22 August 2024; accepted 27 November 2024. Date of publication 25 December 2024; date of current version 17 January 2025. This work was supported in part by Hong Kong Research Grants Council (RGC) under the scheme General Research Fund (GRF) under Grant 17204523, in part by Dajiang Innovations, and in part by China Electronics Technology Group Corporation (CETC). This article was recommended for publication by Associate Editor H. Garcia de Marina and Editor J. Civera upon evaluation of the reviewers' comments. *(Fangcheng Zhu, Yunfan Ren, and Longji Yin contributed equally to this work.) (Corresponding author: Fu Zhang.)*

The authors are with the Mechatronics and Robotic Systems Laboratory, Department of Mechanical Engineering, The University of Hong Kong, Hong Kong (e-mail: zhufc@connect.hku.hk; renyf@connect.hku.hk; ljyin@connect.hku.hk; kongfz@connect.hku.hk; qingboliu0703@gmail.com; xrz836884211@gmail.com; liuwenyi@connect.hku.hk; yixicai@connect.hku.hk; gzlu@connect.hku.hk; haotianl@connect.hku.hk; fuzhang@hku.hk).

Source code is available online at https://github.com/hku-mars/Swarm-LIO2.
Digital Object Identifier 10.1109/TRO.2024.3522155

***Index Terms*—Aerial swarms, light detection and ranging (LiDAR) perception, localization, sensor fusion.**

NOMENCLATURE

$\boxplus/\boxminus$	Encapsulated "boxplus" and "boxminus" operations on the state manifold.
$t_{i,k}$	Time stamp of the kth LiDAR measurement of AAV i.
$\mathbf{x}_{i,k}, \hat{\mathbf{x}}_{i,k}, \bar{\mathbf{x}}_{i,k}$	Ground-truth, predicted, and updated states of AAV i at time stamp $t_{i,k}$.
$\tilde{\mathbf{x}}_{i,k}$	Error between ground-truth state and its estimation at time stamp $t_{i,k}$.
$\breve{\mathbf{x}}_{j,k}$	Measurements utilized by AAV i.
${}^{G_i}\mathbf{T}_{b_i}, {}^{G_i}\mathbf{v}_{b_i}$	Pose (including attitude and position) and linear velocity of AAV i in its own global frame G_i.
$\mathbf{b}_{g_i}, \mathbf{b}_{a_i}, {}^{G_i}\mathbf{g}$	Gyroscope bias, the accelerometer bias, and the gravity vector of AAV i.
$\mathbf{P}_i$	Covariance of AAV i's state.
${}^{i}\tau_{j}$	Temporal offset between AAV j and AAV i.
${}^{G_i}\mathbf{T}_{G_j}$	Extrinsic transformation from the global reference frame of AAV j to that of AAV i, consisting of rotation ${}^{G_i}\mathbf{R}_{G_j}$ and translation ${}^{G_i}\mathbf{p}_{G_j}$.
${}^{G_i}\mathbf{x}_{b_j}$	State of AAV j estimated by AAV i.
${}^{b_i}\breve{\mathbf{p}}_{b_j}$	Relative position of AAV j with respect to AAV i measured in the latter and hence represented by the latter's body frame, also referred to as active mutual observation.
N	Total number of the AAVs in the swarm.

I. INTRODUCTION

IN RECENT years, multirobot systems, especially aerial swarm systems, have exhibited great potential in many fields, such as collaborative autonomous exploration [1], [2], [3], target tracking [4], [5], [6], [7], search and rescue [8], [9], [10], etc. Thanks to their great team cooperation capability, swarm systems can complete various missions in complex scenarios, even in degenerated environments for a single robot. For a single-robot system, well-developed state estimation techniques provide accurate ego state estimation [11], [12], [13], [14], [15], serving as a critical precondition for a wide variety of autonomous tasks such as trajectory planning [16], [17], [18] and motion control [19]. For robotic swarm systems, state estimation plays an equally significant role [20], [21], where each robot needs to estimate the state of the self autonomous aerial

1941-0468 © 2024 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

vehicle (AAV) (i.e., ego state estimation) as well as the other teammate AAVs (i.e., mutual state estimation). Accurate and robust estimation of ego and mutual states is crucial for the robot swarms to collaborate on a task.

Over the past few decades, multiple sensors and devices have been adopted to achieve reliable state estimation for robotic swarm systems. GPS and real time kinematic (RTK)-GPS are commonly used for self-localization in outdoor environments, as reported in previous studies [22], [23]. For GPS-denied environments, motion capture systems [24] and anchor-based ultra-wideband (UWB) systems [25], [26], [27] have been utilized for state estimation in multirobot systems. These methods [24], [25], [26], [27] often rely on the stationary ground station, resulting in a centralized system that is prone to single point of failure. In more recent research, cameras have become a popular choice in multirobot systems due to their lightweight design, low cost, and rich color information. These camera-based systems are often complemented by an inertial measurement unit (IMU) and an anchor-free UWB to provide more robust state estimation [21], [27], [28], [29], [30], [31]. However, cameras are vulnerable to inadequate illumination and lack direct depth measurements, leading to high computational complexity in computing 3-D measurements. Although the complementary anchor-free UWB can provide distance measurements, it is susceptible to multipath effects and obstacle occlusion in the environment, which decreases the overall system accuracy.

In recent years, 3-D light detection and ranging (LiDAR) sensors have gained popularity in state estimation due to their ability to provide direct accurate 3-D measurements over a long range and various illumination conditions. While traditional mechanical spinning LiDARs are often expensive and heavy, recent advancements in LiDAR technology have introduced cost-effective and lightweight LiDARs that are suitable for deployment on mobile robots, particularly AAVs. These LiDARs have not only enabled the development of autonomous navigation systems using LiDARs for AAVs [32], [33], [34], [35], but opened up new possibilities for state estimation in swarm systems.

Leveraging the aforementioned LiDAR advantages, this article aims to develop a fully decentralized, plug-and-play, computationally efficient, and bandwidth-friendly state estimation method for aerial swarm systems based on LiDAR-inertial measurements. Fully decentralized means that no master agent exists in any module of the whole system from communication hardware to algorithm software, which avoids single point of failure. Plug and play means that an agent can automatically join the swarm and easily collaborate with other teammates before or in the middle of a mission. The system must also be computationally efficient and bandwidth efficient, since the limited payload capacity of AAVs imposes significant constraints on the computational resources and network bandwidth.

We decompose the task of aerial swarm state estimation into two key online modules: initialization and state estimation. In the initialization module, each AAV needs to detect and identify possible new teammate AAVs and calibrate temporal offsets and global extrinsic transformations with the found teammates. To achieve simple but effective teammate detection, reflective tapes are attached to each AAV, making teammate AAVs easily detectable from LiDAR reflectivity measurements. This teammate detection is conducted in real time at each LiDAR scan measurement, enabling the detection of new teammate AAVs even in the middle of a mission. For each newly detected AAV, its identification and global extrinsic transformation are obtained through trajectory matching, while the global extrinsics of the other teammate AAVs found on the network are swiftly calibrated through factor graph optimization. Moreover, by exchanging low-dimensional data via a decentralized ad hoc network, teammate monitoring and temporal calibration can be fulfilled efficiently and in a fully decentralized manner.

In the state estimation module, each AAV in the swarm system performs real-time, robust, and precise ego state estimation as well as mutual state estimation. Estimating the full state of all teammates in each AAV is computationally demanding. Thus, we propose to estimate on each AAV only the ego state, meanwhile refining the global extrinsic transformations with respect to (w.r.t.) the teammates. The ego state and global extrinsic transformations are estimated efficiently within an error state iterated Kalman filter (ESIKF) framework, by tightly fusing LiDAR point cloud measurements, IMU measurements, and observed teammate locations (i.e., mutual observation measurements), which are enhanced by careful measurement modeling and temporal compensation. In each step of state estimation, we marginalize the extrinsic states of all teammate AAVs not observed in the LiDAR. This state marginalization, along with a degeneration evaluation method, prevents the state dimension (hence computational complexity) from growing with the swarm size, effectively enhancing the scalability of our system to larger swarms.

This article is extended from our previous work [36], which proposed the general framework of swarm LiDAR-inertial odometry. Compared to the previous work [36], this article proposes five crucial extensions:

1) factor graph optimization for efficient teammate identification and global extrinsic calibration, which largely decreases the complexity and energy consumption of the swarm initialization. Specifically, the number of flights required in the initialization of a swarm with N AAVs is reduced from $O(N)$ to $O(1)$;
2) a novel state marginalization strategy and a LiDAR degeneration evaluation method that alleviate the computational burden and to enhance the swarm scalability. The marginalization reduces the growth rate of the state estimation complexity from cubic to sublinear;
3) detailed measurement modeling and carefully designed temporal compensation of the mutual observation measurements to compensate for the temporal mismatch due to asynchronous sensor measurements among different AAVs;
4) comprehensive simulation and real-world experiments verifying the effectiveness of Swarm-LIO2, i.e., support of large swarm scales (tested five AAVs in the real world, as shown in Fig. 1, and 40 AAVs in simulation), robust to degenerated scenes, and allows the dynamic change of swarm size with online joining or dropping out of any teammate AAVs;

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

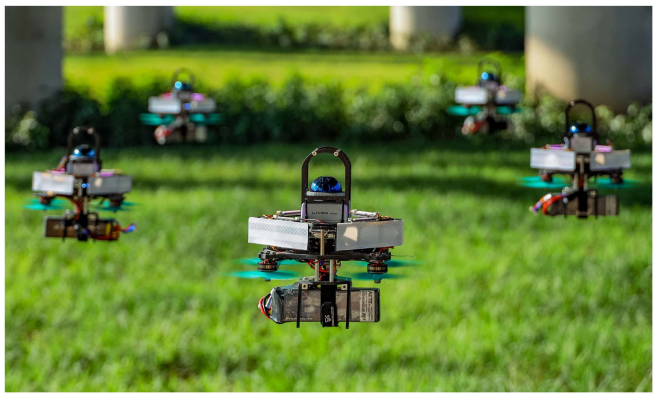

Fig. 1. Aerial swarm system constituted of five AAVs flying in the wild in which Swarm-LIO2 serves as the self and mutual state estimator. More details can be found in the attached video at https://youtu.be/Q7cJ9iRhlrY.

5) an open-source implementation of the proposed system, termed as *Swarm-LIO2*, including source codes for algorithms and hardware designs for our aerial platforms.

II. Related Works

In this section, we first review the mainstream frameworks of the state estimation for robotic swarm systems. Then, we discuss the existing swarm initialization approaches, which is the core module of swarm state estimation.

A. State Estimation for Robotic Swarm

In the past few years, multirobot systems have flourished and some great collaborative simultaneous localization and mapping (SLAM) methods for ground robotic swarm systems have been proposed to achieve robust ego state estimation and consistent mapping [37], [38], [39], [40]. Chang et al. [37] propose a multirobot SLAM system in which each robot sends its single-robot odometry result and constructed submap to a centralized base station to perform loop closure detection and joint pose graph optimization. The wheeled robot platform used in [37] is equipped with abundant sensors, including three Velodyne LiDARs and a complete mobile LiDAR scanner system. Similarly, in [39], a centralized computer that receives map information from all robots is needed to implement loop closure correction and global optimization, and the platform for dataset collecting in [39] contains five cameras and an Ouster LiDAR. The aforementioned centralized systems are fragile to a single point of failure, promoting the development of decentralized methods. Lajoie and Beltrame [40] propose a decentralized multirobot SLAM method in which each robot performs the same computation with only onboard computing resources. In [38] and [40], compact descriptors are exchanged, which could partly decrease the communication load compared to exchanging raw map information, but still, the environmental information size will rapidly grow as the travel distance and swarm scale increases. All the aforementioned methods can be categorized into environment-feature-based methods, of which the obvious weakness is that they are limited to feature-rich environments and the communication bandwidth is relatively high.

Compared to ground robots, the restricted payload capacity and endurance of aerial vehicles greatly limit the weight of sensors and the quality of computation units, which further necessitates a swarm with lightweight sensor configurations, efficient algorithms, and low-bandwidth communication. To satisfy these requirements, some aerial swarm systems [28], [30], [41], [42] directly utilize independent visual-inertial odometry (VIO) to achieve ego state estimation, which gives up the inter-AAV data fusion. Although the independent VIO is easy to adopt, the state estimation results suffer from inevitable drift, especially after long-distance running. To mitigate the VIO drift during the online running of swarm systems, in [43], the inter-AAV place recognition results are exchanged among the AAVs. Likewise, in [20], [27], and [31], the UWB module is incorporated to provide distance constraints as a complementary sensor of the camera. Apart from map and descriptor exchange mentioned earlier, mutual observation is another important unique feature of swarm systems, which is a lightweight type of information, avoiding large communication and computation load. In [21], [44], and [45], learning-based methods, such as YOLOv3-tiny, are utilized to detect other robots to provide mutual observation measurements for VIO drift compensation. These vision-based methods usually struggle in low-visibility environments and may fail to provide accurate 3-D observation results due to imprecise distance estimation, which narrows down their applications in practical cases, e.g., in large-scale outdoor environments.

By contrast, LiDAR can provide accurate and long-range depth measurements, bringing many new opportunities for swarm state estimation. In [26], [46], [47], and [48], 3-D LiDAR-based place recognition (loop closure) is widely utilized to improve state estimation accuracy. However, the large communication bandwidth greatly limits the scalability of the swarm systems. Wasik et al. [49] propose a laser-based multirobot system; laser range finders are used for each robot to estimate the distances and angles to other robots. However, the adopted 2-D LiDARs do not apply to AAVs considered in this article, which fly in 3-D spaces. Pritzl et al. [50] utilize the LiDAR observation measurements to mitigate the VIO drift under the framework of nonlinear least squares optimization.

Compared to the centralized methods [37], [39], [46], [47], our system is fully decentralized, which would suffer no single-point-of-failure issue. Different from the environment-feature-based methods [37], [38], [40], [43], [46], [48], our approach fuses the mutual observation measurements under the ESIKF framework, leading to quite low communication bandwidth and efficient computation. Compared to the camera-based [21], [27], [28], [45] or 2-D LiDAR-based methods [49], our method utilizes the 3-D LiDAR sensor due to its capability of providing accurate and long-range depth measurements with a large field of view (FoV).

B. Swarm Initialization

The critical parts of the initialization of a robotic swarm system typically contain robot detection, robot identification, and global extrinsic calibration.

1) Robot Detection: Learning-based robot detection methods are widely employed for visual-based swarm systems. For

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

instance, Nguyen et al. [45] propose a visual–inertial multi-AAV localization system, in which a neural network for semantic segmentation on micro aerial vehicles (MAVNet) is used to detect other teammate robots. Xu et al. [20] propose a visual–inertial–UWB mutual state estimation system utilizing YOLOv3-tiny for teammate robot detection. These learning-based detection approaches usually need preliminary network training, resulting in extra time and computation consumption. In [51], each robot is equipped with a circle marker for easy detection. In contrast, robot detection is more difficult in LiDAR-based swarm systems because LiDAR lacks the ability to provide texture and color information. In [50] and [52], a local occupancy map is constructed and ray casting is utilized to detect the dynamic obstacles, which is memory intensive and time consuming. In a similar way to [51], in our previous work [36], reflective tapes are attached to each robot (AAV) and leverage the reflectivity measured by LiDAR sensors to detect teammate robots. This detection method is simple but effective, avoiding cumbersome network training.

2) Identification and Global Extrinsic Calibration: In a fully decentralized system, under general circumstances, each AAV estimates the states in its own global frame. Thus, after detecting objects that might be teammate robots, each robot needs to identify other teammates and calibrate the corresponding global extrinsic transformations. In the existing literature, the global extrinsic is usually calibrated offline such as by measuring the distance between the AAVs [27], [28], resulting in quite coarse global extrinsic values. As the flight distance increases, even small deviations in the global extrinsic parameters can lead to a significant drift. In [20], [21], and [45], the global extrinsic parameters are estimated online, but high-quality initial estimations are necessary. Tian et al. [38] calibrate the global extrinsic transformations by constructing a truncated least squares problem that employs the inter-robot loop closure results and the odometric estimates. This method requires environmental information exchange, leading to a large communication burden. Chang et al. [37] place three reflective tapes (fiducial markers) with known 3-D coordinates in the takeoff environment. Then, the LiDAR sensor on each robot would segment the three markers and thereby compute the global extrinsic by minimizing the distance between all triplets of marker positions. Compared to [37], we attach reflective tapes to the AAVs instead of the environment, which supports initialization outside the takeoff area. Different from the one-off calibration [27], [28] or the online estimation relying on good initial values [20], [21], [45], we utilize the trajectory matching proposed in our previous work [36] and factor graph optimization to calibrate the global extrinsic parameters and constantly refine them in the subsequent swarm state estimation. The whole process is fully autonomous, and no initial value is required.

III. System Overview

In this section, we outline the structure of Swarm-LIO2 and give a brief overview of its modules. Aiming to assist understanding of the proposed system, we define some important notations in the Nomenclature. We use $\circ$ to compactly represent the rigid transformation of a point $\mathbf{p} \in \mathbb{R}^{3\times 1}$ with pose $\mathbf{T} = (\mathbf{R}, \mathbf{t}) \in SE(3)$ as $\mathbf{T} \circ \mathbf{p} \triangleq \mathbf{R}\mathbf{p} + \mathbf{t}$.

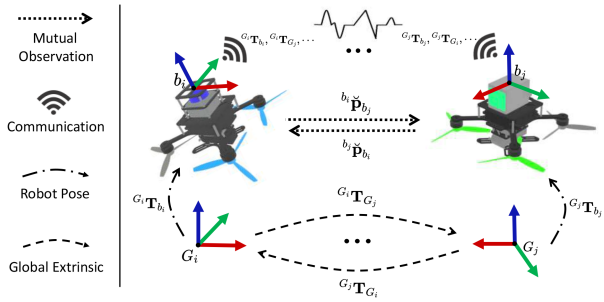

Fig. 2. Illustration of the swarm state estimation problem.

Consider an aerial swarm system consisting of N AAVs, and each one is equipped with a LiDAR and an IMU. To achieve decentralized swarm state estimation, each AAV is required to detect, automatically, all the teammate AAVs in the system and estimate, in real time, the state of itself (e.g., ego state estimation) as well as of all other teammates (e.g., mutual state estimation). Performing ego state and mutual state estimation altogether in one individual AAV is challenging due to the limited onboard computation resources and the high system dimension. Therefore, Swarm-LIO2 estimates the ego state on each AAV and broadcasts the ego state among the teammates. Since the ego state is performed in each AAV's own global reference frame (i.e., the first IMU frame), the extrinsic transformations among all AAV pairs' global frames also need to be calibrated. With the calibrated global extrinsic, the received teammates' ego state can be projected to the self global reference frame, hence achieving the mutual state estimation (see Fig. 2).

Summarizing the aforementioned analysis, Swarm-LIO2 has two key modules. The first module is online initialization, in which each AAV detects all the teammates and performs temporal and spatial calibration with the detected teammate. Let i denote the self-AAV and j denote the detected teammate candidate; since the computer clocks of different AAVs are usually asynchronous, the temporal offset ${}^{i}\tau_{j}$ between the clocks of any two AAVs i and j should be calibrated, which is essential for the inter-AAV data fusion. Then, the self-AAV needs to validate the teammate identity (AAV ID, which is a unique number assigned to each AAV once manufactured) and, if successful, calibrate the global extrinsic transformation ${}^{G_i}\mathbf{T}_{G_j} = ({}^{G_i}\mathbf{R}_{G_j}, {}^{G_i}\mathbf{p}_{G_j}) \in SE(3)$ w.r.t. it.

The second module is state estimation, aiming to estimate in real time the ego state of each AAV (e.g., pose and velocity), by fusing the self-LiDAR and IMU data as well as mutual observation measurements. When the estimated ego state is projected to the teammates' global frames using the global extrinsic transformation, a small extrinsic error that occurs in the initialization stage could lead to a large mutual state estimation error if the AAV's travel distance is long. To mitigate this error, Swarm-LIO2 refines the global extrinsic transformations online along with the ego state.

The two modules of Swarm-LIO2 run in parallel on each AAV of the swarm system, as detailed in Fig. 3. For the initialization module (see Section IV), it further contains three submodules that run concurrently. The first submodule monitors new teammate AAVs on the network and calibrates the temporal

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

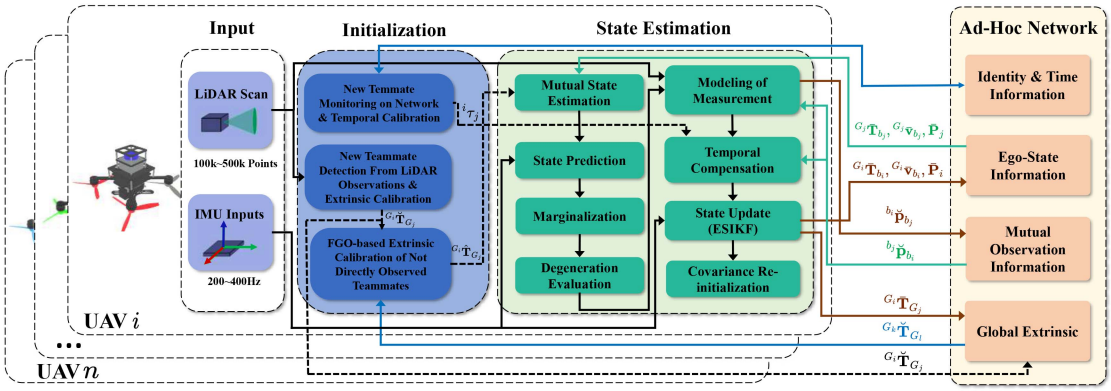

Fig. 3. Framework of the proposed state estimation system for aerial swarm systems. The dashed arrow lines indicate that the messages are sent only once, while the solid arrow lines indicate that the messages are sent constantly at the scan rate.

offsets ${}^{i}\tau_{j}$ w.r.t. it (see Section IV-A). The second submodule detects new teammates observed in LiDAR point cloud and calibrates the global extrinsic transformation ${}^{G_i}\mathbf{T}_{G_j}$ w.r.t. it (see Section IV-B). The calibrated global extrinsics are then sent to the third submodule of the self-AAV and teammate AAVs on the network. Then, the third submodule receives the global extrinsic from the second submodule of the self-AAV or teammate AAVs on the network, based on which the global extrinsic w.r.t. teammates not observed in LiDAR point cloud are calibrated via a factor graph optimization (see Section IV-C). Once the global extrinsic ${}^{G_i}\mathbf{T}_{G_j}$ w.r.t. AAV j is calibrated, AAV j is considered as a valid teammate whose state will be added to and estimated in the state estimation module. Meanwhile, the extrinsic ${}^{G_i}\mathbf{T}_{G_j}$ is sent to the state estimation module for further refinement.

For the state estimation module (see Section V), the swarm state is estimated, which consists of the ego state and the global extrinsic transformations w.r.t. all teammates. To reduce the state dimension, Swarm-LIO2 performs a marginalization step (see Section V-D), followed by a degeneration evaluation to evaluate the degeneration of current LiDAR measurements indicated by $\mathcal{I}_i$ and to perform further marginalization (see Section V-E). The marginalized state is then estimated in an ESIKF framework [53] by performing state prediction (see Section V-B) and iterative update (see Section V) after measurements modeling (see Section V-C1). The measurements contain LiDAR point cloud and mutual observation measurements (i.e., the teammate location observed by the self-AAV, denoted by ${}^{b_i}\breve{\mathbf{p}}_{b_j}$, and the self-location observed by the teammate, denoted by ${}^{b_j}\breve{\mathbf{p}}_{b_i}$, which is received from the teammate). Mutual observations have temporal mismatch, which are temporally compensated in Section V-C2. The state estimation results are finally transmitted to other teammate AAVs for their next round of state estimation (see Section V-F). In Swarm-LIO2, all information is exchanged via a fully decentralized ad hoc network infrastructure under the IEEE 802.11 architecture (i.e., IBSS), which is broadly supported by commonly used Wi-Fi modules and can be configured by programming the Wi-Fi driver [54].

IV. Swarm Initialization

In this section, we introduce the three submodules of the swarm initialization process running on each AAV of the swarm system. The first submodule (see Section IV-A) monitors new teammate AAVs on the network and calibrates the temporal offsets w.r.t. them. The second submodule (see Section IV-B) monitors new teammate AAVs from LiDAR measurements, validates their identities, and calibrates the global extrinsic transformations w.r.t. them via trajectory matching. The third submodule (see Section IV-C) monitors the extrinsic updates from the second module or the network and calibrates the global extrinsic transformations w.r.t. other teammates that are not directly observed by its LiDAR, via a novel factor graph optimization.

A. New Teammate Monitoring on Network and Temporal Calibration

The first submodule detects teammate AAVs on the network, maintains the connection with the found teammates, and performs temporal offset calibration w.r.t. them. To achieve this, each AAV would continuously broadcast its identity information in the ad hoc network, including its AAV ID and Internet protocol address, at a fixed frequency of 1 Hz. This identity information is commonly called the "heartbeat" packet, used for teammate monitoring and communication status maintenance. The "heartbeat" packet could also be encrypted if necessary to prevent AAV information leakage or cyberattacks. Upon receiving identity information from a teammate, the self-AAV adds the teammate to its teammate list and maintains the connection status continuously. For each teammate in the teammate list, the self-AAV assigns one of two states: connected or disconnected. After a teammate is added to the teammate list, its corresponding status is initialized as "connected." If the AAV fails to receive identity information from a teammate for 2 s, the teammate's state is set to "disconnected." Upon receiving the identity information from the disconnected teammate again, the state is switched back to "connected."

After discovering a new teammate on the network, the crucial temporal offset w.r.t. the teammate AAV is calibrated. For each

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

teammate AAV, a decentralized temporal calibration method based on the peer-delay mechanism in precision time protocol (PTP) [55] is utilized to acquire the temporal offset corresponding to it. The self-AAV would send request messages to each teammate AAV and receive response messages from teammates. By leveraging the time stamps of these messages, the self-AAV can calculate the temporal offset w.r.t. each teammate AAV following the principle of PTP [55]. To suppress random errors or fluctuations, this process is repeated 30 times, and the average value of ${}^{i}\tau_j$ is adopted. Since the clock drift among different AAVs is negligible within the typical AAV flight time (e.g., less than an hour), estimating the temporal offset one time is sufficient for actual swarm tasks, i.e., for any AAV i, once its temporal offset w.r.t. AAV j, ${}^{i}\tau_j$, is obtained, the corresponding temporal calibration is considered as complete and no request–response messages will be communicated with AAV j. If the clock drift is significant, the temporal offset can be estimated constantly at a fixed frequency, e.g., 1 Hz. This mechanism is performed for both AAVs in each pair of the swarm system and is robust to single point of failure due to the absence of a designated master clock. Each AAV performs temporal offset calibration with every teammate AAV newly found in the network. The calibrated temporal offset ${}^{i}t_j$ is stored in a Hash table where the key is AAV ID and the value is temporal offset. When a AAV receives any data from a teammate, the data are stamped with the teammate's clock. To use the data for the self-AAV, the received data will have its time stamp modified according to the temporal offset obtained by the fast and efficient lookup of the Hash table.

B. New Teammate Detection From LiDAR Observations and Extrinsic Calibration

For any teammate in the teammate list, apart from calibrating the temporal offset, each AAV also needs to calibrate the spatial offset, i.e., the extrinsic transformations between the two AAVs' global reference frames. In this section, we calibrate the global extrinsic w.r.t. those teammates observed by the LiDAR on the self-AAV.

We propose a novel reflectivity filtering and cluster extraction-based teammate detection method to easily detect the observed teammate AAVs from LiDAR point cloud measurements. After accumulating the trajectory of the directly observed objects over a certain time, a trajectory-matching-based identification and global extrinsic calibration method is used for fast swarm initialization.

To implement the aforementioned method, for each AAV, several reflective tapes are attached to its body, so that it can be easily detected and tracked by other teammates based on the reflectivity information measured by the LiDAR sensor. The detailed implementation of this submodule is summarized in Algorithm 1. After receiving a LiDAR scan, we first undistort the raw points following [11] and filter out the points on teammate AAVs with whom the initialization has completed (which is explained later in Section V-C1), to obtain the LiDAR points ${}^{b_i}\mathcal{P}$, which is represented in the current body frame. Then, points with high reflectivity values exceeding a predefined threshold, which can be calibrated beforehand on the reflective tapes attached to each

Algorithm 1: New Teammate Detection, Identification, and Global Extrinsic Calibration.

Input: A scan of LiDAR raw points excluding points on known teammate UAVs ${}^{b_i}\mathcal{P}$, UAV i's odometry ${}^{G_i}\mathbf{T}_{b_i}$, time interval of LiDAR input Δt, UAV j's position trajectory ${}^{G_j}\mathcal{T}_j$, threshold of trajectory matching residual thr

Output: The number of clusters M, Global extrinsic transformation ${}^{G_i}\mathbf{T}_{G_j}$, tracked position of UAV j in UAV i's global frame ${}^{G_i}\mathbf{p}_{b_j}$

1 ${}^{b_i}\mathcal{P}_h$ = `ReflectivityFiltering`(${}^{b_i}\mathcal{P}$);
2 M, ${}^{b_i}\breve{\mathbf{p}}_m$ = `FastEuclideanClustering`(${}^{b_i}\mathcal{P}_h$);
3 **for** $m = 1 : M$ **do**
4 ${}^{G_i}\bar{\mathbf{p}}_m$ = `TemporaryTracking`(Δt, ${}^{G_i}\mathbf{T}_{b_i}$, ${}^{b_i}\mathcal{P}$, ${}^{b_i}\breve{\mathbf{p}}_m$);
5 ${}^{G_i}\mathcal{T}_m$.PushBack(${}^{G_i}\bar{\mathbf{p}}_m$);
6 **if** `TrajExcited`(${}^{G_i}\mathcal{T}_m$) **then**
7 **for** $j = 1 : N$; $j \neq i$ **do**
8 res, $\mathbf{T}$ = `TrajMatching`(${}^{G_j}\mathcal{T}_j$, ${}^{G_i}\mathcal{T}_m$);
9 **if** $res < thr$ **then**
10 ${}^{G_i}\mathbf{T}_{G_j} = \mathbf{T}$;
11 ${}^{G_i}\mathbf{p}_{b_j} = {}^{G_i}\mathbf{p}_m$;
12 break;
13 **end**
14 **end**
15 **end**
16 **end**

AAV, are extracted by `ReflectivityFiltering`(${}^{b_i}\mathcal{P}$) in line 1 of Algorithm 1. Then, in line 2, the high-reflectivity points ${}^{b_i}\mathcal{P}_h$ are efficiently clustered by `FastEuclideanClustering` [56], which aims to detect new potential teammate AAVs. Each detected object, with clustered centroid ${}^{b_i}\breve{\mathbf{p}}_m$, is then tracked by a Kalman-filter-based temporary tracker based on the assumption of constant velocity in line 4 [36].

The tracked trajectory of each potential new teammate is accumulated (line 5) for subsequent identification and global extrinsic calibration. To achieve this, all AAVs in the swarm system will exchange their estimated ego states (in their own global frames) with others. The ego state of each AAV is estimated by the state estimation module (see Section V), which runs in parallel to the initialization module, as explained in Section III, based on LiDAR points, IMU data, and mutual observations of teammate AAVs if available.

Let ${}^{G_i}\mathcal{T}_m = \{{}^{G_i}\bar{\mathbf{p}}_{m,\kappa}, \kappa = 1, \ldots, \mathcal{K}\}$ denote the trajectory of the mth temporary tracker and ${}^{G_j}\mathcal{T}_j = \{{}^{G_j}\breve{\mathbf{p}}_{b_j,\kappa}, \kappa = 1, \ldots, \mathcal{K}\}$ represent the trajectory received from AAV j; the mth tracked object is identified as AAV j if the following trajectory matching problem has a unique optimal solution with a residual below a certain threshold:

$$\arg\min_{{}^{G_i}\mathbf{T}_{G_j}} \sum_{\kappa=1}^{\mathcal{K}} \frac{1}{2} \| {}^{G_i}\bar{\mathbf{p}}_{m,\kappa} - {}^{G_i}\mathbf{T}_{G_j} \circ {}^{G_j}\breve{\mathbf{p}}_{b_j,\kappa} \|. \tag{1}$$

Considering possible short-term communication disconnection, some data of ${}^{G_j}\breve{\mathbf{p}}_{b_j}$ might be lost. Thus, we only pick ${}^{G_i}\bar{\mathbf{p}}_{m,\kappa}$ that has close time stamp with ${}^{G_j}\breve{\mathbf{p}}_{b_j,\kappa}$ to participate in trajectory matching. Besides, to avoid large computing time due to too much data, we use a sliding window of the most recent $\mathcal{K}$ positions for matching. We implement the aforementioned trajectory matching process as a function `TrajMatching`(${}^{G_j}\mathcal{T}_j$, ${}^{G_i}\mathcal{T}_m$) in line 8.

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

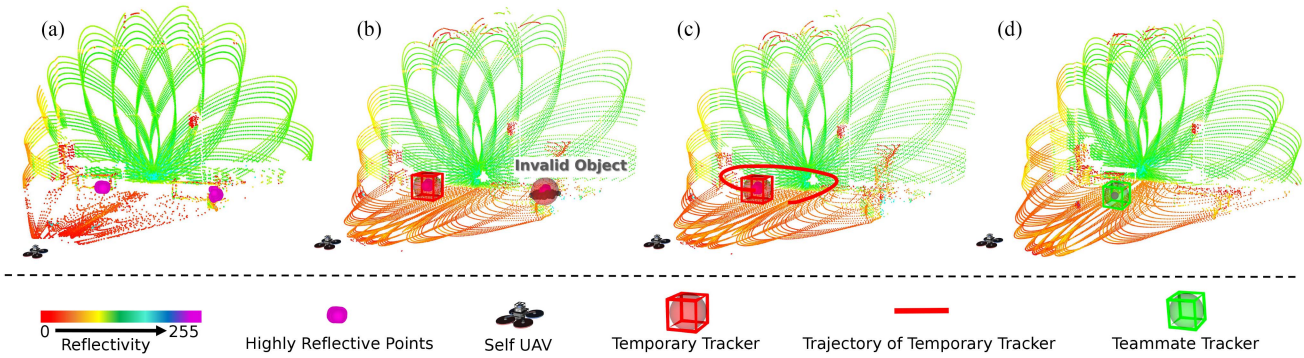

Fig. 4. Illustration of the initialization for newly detected objects; point cloud is colored by reflectivity. Here, the self-AAV needs to detect and identify other teammate AAVs in its FoV. The sphere represents the predicted region and the center of the bounding box represents the updated position of the tracker. (a) Reflectivity filtering. (b) Outlier rejection by discarding objects with too large size. (c) Track real potential teammates and accumulate the trajectory. (d) After trajectory matching, the object is identified as a teammate AAV with a correct AAV ID.

Since no unique transformation can be determined from (1) if the involved trajectories are straight lines [57], the trajectories of those tracked objects are constantly evaluated by $\texttt{TrajExcited}({}^{G_i}\mathcal{T}_m)$ in line 6. Once the condition of $\texttt{TrajExcited}({}^{G_i}\mathcal{T}_m)$ is satisfied, the trajectory matching will be performed. Let ${}^{G_i}\bar{\mathbf{p}}_m^c$ represent the centroid of ${}^{G_i}\mathcal{T}_m$; then, $\texttt{TrajExcited}({}^{G_i}\mathcal{T}_m)$ assesses the excitation (shape) of ${}^{G_i}\mathcal{T}_m$ by computing the singular values of matrix $\mathcal{H} \in \mathbb{R}^{3\times 3}$

$$\mathcal{H} \triangleq \sum_{\kappa=1}^{\mathcal{K}} ({}^{G_i}\bar{\mathbf{p}}_{m,\kappa} - {}^{G_i}\bar{\mathbf{p}}_m^c) \cdot ({}^{G_i}\bar{\mathbf{p}}_{m,\kappa} - {}^{G_i}\bar{\mathbf{p}}_m^c)^T. \quad (2)$$

If the second largest singular value is larger than a given threshold, it means that the positions of the trajectory ${}^{G_i}\mathcal{T}_m$ do not lie on a straight line, which ensures a unique solution in $\texttt{TrajMatching}({}^{G_j}\mathcal{T}_j, {}^{G_i}\mathcal{T}_m)$ [57]. The matching for temporary tracker trajectory ${}^{G_i}\mathcal{T}_m$ is performed with all received teammate AAV's trajectory ${}^{G_j}\mathcal{T}_j, j = 1, \ldots, N$ until the matching error is smaller than a given threshold, indicating that the object m is essentially the observation of teammate with AAV ID j, and the solution of (1) gives an initial estimation of the global extrinsic ${}^{G_i}\breve{\mathbf{T}}_{G_j}$. Meanwhile, the mth temporary tracker will be removed from the temporary tracker list and become a teammate tracker that will be used in the state estimation module (see Section V). An example of the trajectory-matching-based initialization pipeline is illustrated in Fig. 4.

Finally, the extrinsic obtained by trajectory matching ${}^{G_i}\breve{\mathbf{T}}_{G_j}$ is sent to the third submodule of the self-AAV as well as all teammate AAVs on the teammate list. Note that the sending of ${}^{G_i}\breve{\mathbf{T}}_{G_j}$ occurs only once since after the temporary tracker has been removed, no trajectory matching will be performed to produce the extrinsic ${}^{G_i}\breve{\mathbf{T}}_{G_j}$ in the next cycle.

C. Factor-Graph-Optimization-Based Global Extrinsic Calibration of Not Directly Observed Teammates

Apart from the trajectory-matching-based identification method detailed earlier, which was originally presented in our previous work [36], a novel decentralized factor graph optimization method is proposed to calibrate the global extrinsic transformations w.r.t. not directly observed teammates, which expedites the identification and the swarm initialization. In Swarm-LIO2, each AAV will share the global extrinsic transformations obtained via trajectory matching with all teammate AAVs in the teammate list. Then, each AAV constructs and maintains a factor graph (see Fig. 5) where the variables $G_i, G_j, \ldots$ are the global reference frames of all AAVs (including teammates with or without direct observation) and the factors ${}^{G_i}\breve{\mathbf{T}}_{G_j}$ are the global extrinsic transformation between any two AAVs, which could be calibrated by the self-AAV using the trajectory matching or received from teammate AAVs. By fixing the global frame G_i of the self-AAV, it can use the global extrinsic transformations ${}^{G_i}\breve{\mathbf{T}}_{G_j}$ as constraints to solve for the global frames of all other AAVs who are connected to the self-AAV in the factor graph. Subsequently, the global extrinsic between each AAV (with and without direct observations) and the ego AAV can be deduced as ${}^{G_i}\hat{\mathbf{T}}_{G_j} = G_i G_j^{-1}$.

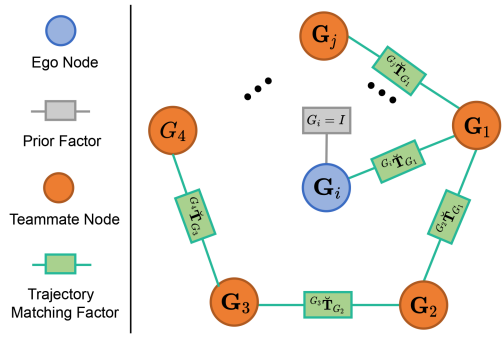

Fig. 5. Illustration of the decentralized factor-graph-optimization-based global extrinsic calibration. To handle the gauge freedom of the factor graph, we insert a prior factor as $G_i = I$. Note that the factors ${}^{G_i}\breve{\mathbf{T}}_{G_j}$ are global extrinsic transformations received from teammate AAVs or obtained by direct trajectory matching on the self-AAV.

The third submodule runs after the second submodule, hence running recurrently at the scan rate too. Specifically, it receives extrinsic ${}^{G_i}\breve{\mathbf{T}}_{G_j}$ from the second submodule and extrinsic ${}^{G_k}\breve{\mathbf{T}}_{G_l}$ from the network. If the received extrinsic (either from the second submodule or from the network) corresponds to a new edge that did not exist before in the factor graph,

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

an edge corresponding to this extrinsic will be created in the factor graph. Otherwise, the received extrinsic will be dumped to avoid information reuse. On the other hand, if there are multiple global extrinsic transformations on the same edge, such as the ${}^{G_i}\breve{\mathbf{T}}_{G_j}$, which is obtained through trajectory matching on the self-AAV i, and ${}^{G_j}\breve{\mathbf{T}}_{G_i}$, which is obtained through trajectory matching on the teammate AAV (received on the network), the average of these global extrinsic transformations is computed and used as a factor, which can effectively save the number of factors in the factor graph. In case the factor graph is updated, an optimization process is performed using iSAM2 [58], and the optimized global extrinsic, if not sent before, is sent to the state estimation module as the initial estimation ${}^{G_i}\hat{\mathbf{T}}_{G_j}$ for the online global extrinsic refinement.

Remark 1: As can be seen, the first submodule runs concurrently with the second and the third submodules at different frequencies. The first submodule runs at 1 Hz, while the second and the third submodules run at the scan rate. The recurrent nature of the three submodules allows the swarm to discover, identify, and calibrate the global extrinsic w.r.t. new joining AAVs in the middle of a mission.

Remark 2: Each of the three submodules breaks into two parts. The first parts of the three submodules monitor for new teammates on the network, new teammates observed by LiDAR measurements, or new edges in the factor graph, respectively. The second parts conduct the temporal calibration, trajectory matching, and factor graph optimization, respectively. While the first parts run at their respective frequencies, the second parts run only when the first parts detect new teammates or edges.

V. Decentralized Swarm State Estimation

In this section, we will introduce the fully decentralized swarm state estimation module, including mutual state estimation, ESIKF-based ego state estimation, and global extrinsic refinement.

A. Mutual State Estimation

One crucial mission of each AAV in a swarm system is to estimate any other AAV j's state ${}^{G_i}\mathbf{x}_{b_j}$ (including pose ${}^{G_i}\mathbf{T}_{b_j}$ and velocity ${}^{G_i}\mathbf{v}_{b_j}$) in AAV i's global frame, called mutual state estimation. This is significant for various swarm applications, such as mutual collision avoidance, formation flight, etc. However, estimating the full state ${}^{G_i}\mathbf{x}_{b_j}$ of all the teammate AAVs is a high-dimensional task that is computationally demanding. To reduce the system complexity, we propose to estimate the ego state only on each AAV, denoted by ${}^{G_i}\bar{\mathbf{x}}_{b_i}$. The estimated ego states are exchanged through the network. Then, a AAV i can estimate a teammate j's state, denoted by ${}^{G_i}\bar{\mathbf{x}}_{b_j}$, by directly projecting the received AAV j's ego state into AAV i's global frame using the global extrinsic transformation ${}^{G_i}\mathbf{T}_{G_j} = ({}^{G_i}\mathbf{R}_{G_j}, {}^{G_i}\mathbf{p}_{G_j})$

$$\begin{aligned} {}^{G_i}\bar{\mathbf{T}}_{b_j} &= {}^{G_i}\mathbf{T}_{G_j}\,{}^{G_j}\breve{\mathbf{T}}_{b_j} \\ {}^{G_i}\bar{\mathbf{v}}_{b_j} &= {}^{G_i}\mathbf{R}_{G_j}\,{}^{G_j}\breve{\mathbf{v}}_{b_j}. \end{aligned} \tag{3}$$

where ${}^{G_j}\breve{\mathbf{T}}_{b_j}$ is the received ego state of AAV j. Note that ${}^{G_j}\breve{\mathbf{T}}_{b_j}$ is denoted as ${}^{G_j}\bar{\mathbf{T}}_{b_j}$ on AAV j (since it is an estimation on AAV j), but has an accent $\breve{(\cdot)}$ on AAV i (since it acts as a measurement for AAV i).

A problem in the aforementioned process is that it requires knowing the global extrinsic transformation ${}^{G_i}\mathbf{T}_{G_j} = ({}^{G_i}\mathbf{R}_{G_j}, {}^{G_i}\mathbf{p}_{G_j})$. Although it can be calibrated by the initialization process described in Section IV, possible errors could still remain. We propose to continually refine this extrinsic transformation along with the ego state estimation in the state estimation module. Denoting ${}^{G_i}\bar{\mathbf{T}}_{G_j} = ({}^{G_i}\bar{\mathbf{R}}_{G_j}, {}^{G_i}\bar{\mathbf{p}}_{G_j})$ as the refined extrinsic transformation, the mutual state of teammate j can be computed by (3), with ${}^{G_i}\mathbf{T}_{G_j} = ({}^{G_i}\mathbf{R}_{G_j}, {}^{G_i}\mathbf{p}_{G_j})$ being replaced by ${}^{G_i}\bar{\mathbf{T}}_{G_j} = ({}^{G_i}\bar{\mathbf{R}}_{G_j}, {}^{G_i}\bar{\mathbf{p}}_{G_j})$. This mechanism enables smooth and stable mutual state estimation even in situations where frequent mutual observation losses occur due to occlusions or teammates entering and exiting the FoV.

To refine the extrinsic transformation in the state estimation, for each new teammate j whose extrinsic was calibrated in the initialization module (see Section IV), we append ${}^{G_i}\mathbf{T}_{G_j} = ({}^{G_i}\mathbf{R}_{G_j}, {}^{G_i}\mathbf{p}_{G_j})$ to the existing state vector of AAV i, so its value can be estimated along with other states in a unified ESIKF framework. Moreover, the calibrated extrinsic from the initialization module immediately serves as the initial estimation ${}^{G_i}\widehat{\mathbf{T}}_{G_j}$ of the appended state component ${}^{G_i}\mathbf{T}_{G_j}$.

B. State and Covariance Prediction

We first introduce the scheme of the state and covariance prediction. For illustration, we select AAV i as the self-AAV and assume that N–1 teammate AAVs have been found and calibrated in the initialization module. Let τ denote the IMU measurement index during the kth LiDAR frame; then, the discrete state transition model is shown as follows:

$$\mathbf{x}_{i,\tau+1} = \mathbf{x}_{i,\tau} \boxplus (\Delta t_\tau \mathbf{f}_i(\mathbf{x}_{i,\tau}, \mathbf{u}_{i,\tau}, \mathbf{w}_{i,\tau})) \tag{4}$$

where Δt_τ is the time interval between two consecutive IMU measurements, and $\mathbf{x}_{i,\tau}$ denotes the ground truth of the state at the τth IMU measurement of the ith AAV, whose time stamp is $t_{i,\tau}$. Furthermore, we use the notation $\boxplus/\boxminus$ defined in [59] to compactly represent the "plus" on the state manifold. Specifically, for the state manifold $SO(3) \times \mathbb{R}^n$ in (5), the $\boxplus$ operation and its inverse operation $\boxminus$ are defined as

$$\begin{bmatrix}\mathbf{R}\\ \mathbf{a}\end{bmatrix} \boxplus \begin{bmatrix}\mathbf{r}\\ \mathbf{b}\end{bmatrix} = \begin{bmatrix}\mathbf{R}\mathrm{Exp}(\mathbf{r})\\ \mathbf{a}+\mathbf{b}\end{bmatrix}; \begin{bmatrix}\mathbf{R}_1\\ \mathbf{a}\end{bmatrix} \boxminus \begin{bmatrix}\mathbf{R}_2\\ \mathbf{b}\end{bmatrix} = \begin{bmatrix}\mathrm{Log}(\mathbf{R}_2^T\mathbf{R}_1)\\ \mathbf{a}-\mathbf{b}\end{bmatrix}$$

where $\mathbf{R}, \mathbf{R}_1, \mathbf{R}_2 \in SO(3), \mathbf{r} \in \mathbb{R}^3, \mathbf{a}, \mathbf{b} \in \mathbb{R}^n$, $\mathrm{Exp}(\cdot): \mathbb{R}^3 \mapsto SO(3)$ is the exponential map on $SO(3)$[59], and $\mathrm{Log}(\cdot): SO(3) \mapsto \mathbb{R}^3$ is its inverse logarithmic map.

The state vector $\mathbf{x}_i$, the process noise vector $\mathbf{w}_i$, and the input $\mathbf{u}_i$, omitting the time index, are defined as

$$\begin{aligned} \mathbf{x}_i &\triangleq \Big[{}^{G_i}\mathbf{R}_{b_i}^T \;\; {}^{G_i}\mathbf{p}_{b_i}^T \;\; {}^{G_i}\mathbf{v}_{b_i}^T \;\; \mathbf{b}_{g_i}^T \;\; \mathbf{b}_{a_i}^T \;\; {}^{G_i}\mathbf{g}^T \\ &\qquad \cdots \;\; {}^{G_i}\mathbf{R}_{G_j}^T \;\; {}^{G_i}\mathbf{p}_{G_j}^T \;\; \cdots\Big]^T \in \mathcal{M} \\ \mathbf{w}_i &\triangleq \begin{bmatrix}\mathbf{n}_{g_i}^T & \mathbf{n}_{a_i}^T & \mathbf{n}_{\mathbf{b}_{gi}}^T & \mathbf{n}_{\mathbf{b}_{ai}}^T\end{bmatrix}^T \\ \mathbf{u}_i &\triangleq \begin{bmatrix}\boldsymbol{\omega}_{m_i}^T & \mathbf{a}_{m_i}^T\end{bmatrix}^T \end{aligned} \tag{5}$$

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

where $j = 1, 2, \ldots, i-1, i+1, \ldots, N$.

The discrete state transition function $\mathbf{f}_i$ is defined as

$$\mathbf{f}_i \triangleq \begin{bmatrix} \boldsymbol{\omega}_{m_i} - \mathbf{b}_{g_i} - \mathbf{n}_{g_i} \\ {}^{G_i}\mathbf{v}_{b_i} + \frac{1}{2}({}^{G_i}\mathbf{R}_{b_i}(\mathbf{a}_{m_i} - \mathbf{b}_{a_i} - \mathbf{n}_{a_i}) + {}^{G_i}\mathbf{g})\Delta t_\tau \\ {}^{G_i}\mathbf{R}_{b_i}(\mathbf{a}_{m_i} - \mathbf{b}_{a_i} - \mathbf{n}_{a_i}) + {}^{G_i}\mathbf{g} \\ \mathbf{n}_{\mathbf{b}_{gi}} \\ \mathbf{n}_{\mathbf{b}_{ai}} \\ \mathbf{0}_{3\times 1} \\ \vdots \\ \mathbf{0}_{3\times 1} \\ \mathbf{0}_{3\times 1} \\ \vdots \end{bmatrix} \tag{6}$$

where $\boldsymbol{\omega}_{m_i}$ and $\mathbf{a}_{m_i}$ represent the IMU (gyroscope and accelerometer) measurements of AAV i, $\mathbf{n}_{g_i}$ and $\mathbf{n}_{a_i}$ are the white noise of IMU measurements, and $\mathbf{b}_{g_i}$ and $\mathbf{b}_{a_i}$ are the IMU bias modeled as the random walk process with Gaussian noises $\mathbf{n}_{\mathbf{b}_{gi}}$ and $\mathbf{n}_{\mathbf{b}_{ai}}$. The meaning of each element in state vector $\mathbf{x}_i$ is introduced in the Nomenclature, the state manifold $\mathcal{M}$ is defined in (7), and its dimension is $18 + 6(N-1)$

$$\mathcal{M} \triangleq \underbrace{SO(3) \times \mathbb{R}^{15}}_{\dim=18} \times \underbrace{\cdots \times SO(3) \times \mathbb{R}^3 \times \cdots}_{\dim=6(N-1)}. \tag{7}$$

Following the state model in (4), the state and covariance prediction is implemented under the ESIKF framework once receiving a new IMU measurement. More specifically, the state and covariance are predicted following (4) by setting the process noise $\mathbf{w}_{i,\tau}$ to zero. The detailed demonstration of predication can be referred to [11] and [12].

C. Error State Iterative State Update

The update step is implemented iteratively at the end time of the new LiDAR scan at $t_{i,k}$, fusing point cloud measurements and mutual observation measurements (if any). In the following sections, we will introduce the measurement model of the point cloud measurements and the novel mutual observation measurements, which were not present in [36].

1) Modeling of Measurements: In the general ESIKF framework, for any measurement $\mathbf{y}_k$ at the kth round, we can write the measurement model as

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{v}_k) \tag{8}$$

where $\mathbf{h}(\mathbf{x}_k, \mathbf{v}_k)$ is the measurement model depending on the true state $\mathbf{x}_k$ and the measurement noise $\mathbf{v}_k$, which is assumed to be zero-mean multivariate Gaussian noise. For convenience and simplification of the description, we omit the subscript k in the following formulations.

Once receiving a new LiDAR scan, motion compensation will be performed to obtain the undistorted point clouds. Then, the point-to-plane distance will be calculated to generate point cloud constraints. The details of the motion compensation can be referred to [11]. The nth undistorted point of the current scan projected into the body frame is denoted by ${}^{b_i}\mathbf{p}_n$, let $\mathbf{u}_n$ represent the normal vector of the corresponding plane in the global frame G_i, on which lies a point ${}^{G_i}\mathbf{q}_n$. Considering the LiDAR measurement noise $\mathbf{n}_{p,n}$ of the nth point, we obtain the measurement model of the nth point measurement as [11], [12]

$$\mathbf{0} = \underbrace{\mathbf{u}_n^T({}^{G_i}\mathbf{T}_{b_i} \circ ({}^{b_i}\mathbf{p}_n + \mathbf{n}_{p,n}) - {}^{G_i}\mathbf{q}_n)}_{\mathbf{h}_{p,n}(\mathbf{x}_i, \mathbf{n}_{p,n})} \tag{9}$$

which defines an implicit measurement equation about the state vector $\mathbf{x}_i$ containing ego pose ${}^{G_i}\mathbf{T}_{b_i}$. The normal vector $\mathbf{u}_n$ and the point ${}^{G_i}\mathbf{q}_n$ are known vectors, and $\mathbf{n}_{p,n}$ is point measurement noise, both can be referred to [11] and [12].

Apart from the point cloud measurements, the mutual observation measurements are also used for state updates, which can be obtained by the teammate tracker evolved from the temporary tracker in the initiation module. Specifically, with the predicted pose ${}^{G_i}\widehat{\mathbf{T}}_{b_i}$ obtained in Section V-B, we can acquire the predicted position of each teammate AAV j described in AAV i's body frame as ${}^{b_i}\widehat{\mathbf{p}}_{b_j} = ({}^{G_i}\widehat{\mathbf{T}}_{b_i}^{-1}{}^{G_i}\bar{\mathbf{T}}_{G_j}) \circ {}^{G_j}\breve{\mathbf{p}}_{b_j}$. The LiDAR points around the predicted position ${}^{b_i}\widehat{\mathbf{p}}_{b_j}$ will be removed from the LiDAR raw points. The rest of the points ${}^{b_i}\mathcal{P}$ will be used by $\texttt{ReflectivityFiltering}({}^{b_i}\mathcal{P})$ in the initialization module (see Algorithm 1) for new teammate detection. Moreover, the points around the predicted teammate positions will be used for Euclidean clustering to obtain the mutual observation measurement. If a valid object is clustered, the centroid position of the object will be regarded as the actual position of AAV j observed by AAV i, called "active observation measurement" for AAV i w.r.t. AAV j, which is denoted by ${}^{b_i}\breve{\mathbf{p}}_{b_j}$. Each AAV would share this active observation measurement with all teammates and meanwhile receive teammates' ones via the ad hoc network. The received observation measurement from AAV j is referred to as "passive observation measurements" and is denoted by ${}^{b_j}\breve{\mathbf{p}}_{b_i}$, representing the self-position of AAV i observed by teammate j.

The explicit measurement model of the active observation measurement ${}^{b_i}\breve{\mathbf{p}}_{b_j}$ can be obtained by projecting AAV j's ground-truth position ${}^{G_j}\mathbf{p}_{b_j}$ into AAV i's body frame using the ground-truth global extrinsic ${}^{G_i}\mathbf{T}_{G_j}$ and the ground-truth ego pose ${}^{G_i}\mathbf{T}_{b_i} = ({}^{G_i}\mathbf{R}_{b_i}, {}^{G_i}\mathbf{p}_{b_i})$ of AAV i. Further considering that the active observation may have measurement noise $\mathbf{n}_{ao,ij} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{ao,ij})$ due to incomplete point measurements on the teammate AAV j, the model of the active observation measurement is:

$${}^{b_i}\breve{\mathbf{p}}_{b_j} = \left({}^{G_i}\mathbf{T}_{b_i}^{-1}{}^{G_i}\mathbf{T}_{G_j}\right) \circ {}^{G_j}\mathbf{p}_{b_j} + \mathbf{n}_{ao,ij}. \tag{10}$$

This measurement equation, unfortunately, involves the ground-truth position ${}^{G_j}\mathbf{p}_{b_j}$ of AAV j, which is not a part of the state vector $\mathbf{x}_i$ as defined in (5). To fix this issue, we leverage the estimated ego position ${}^{G_j}\breve{\mathbf{p}}_{b_j}$ of AAV j and its covariance $\breve{\boldsymbol{\Sigma}}_{\mathbf{p}_j}$, both are received from AAV j. Then, the ground-truth position of AAV j is modeled as ${}^{G_j}\mathbf{p}_{b_j} = {}^{G_j}\breve{\mathbf{p}}_{b_j} + \mathbf{n}_{\mathbf{p}_j}$, where the noise $\mathbf{n}_{\mathbf{p}_j} \sim \mathcal{N}(\mathbf{0}, \breve{\boldsymbol{\Sigma}}_{\mathbf{p}_j})$. Consequently, the measurement model of the active observation measurement can be derived as:

$${}^{b_i}\breve{\mathbf{p}}_{b_j} = \underbrace{\left({}^{G_i}\mathbf{T}_{b_i}^{-1}{}^{G_i}\mathbf{T}_{G_j}\right) \circ ({}^{G_j}\breve{\mathbf{p}}_{b_j} + \mathbf{n}_{\mathbf{p}_j}) + \mathbf{n}_{ao,ij}}_{\mathbf{h}_{ao,ij}(\mathbf{x}_i, \mathbf{n}_{\mathbf{p}_j}, \mathbf{n}_{ao,ij})} \tag{11}$$

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

which defines a valid measurement equation about the state vector $\mathbf{x}_i$ containing ego pose ${}^{G_i}\mathbf{T}_{b_i}$ and global extrinsic ${}^{G_i}\mathbf{T}_{G_j}$. The received teammate position ${}^{G_j}\breve{\mathbf{p}}_{b_j}$ is known, and $\mathbf{n}_{\mathbf{p}_j}, \mathbf{n}_{ao,ij}$ are measurement noises.

Similarly, the explicit measurement model of the passive observation measurement ${}^{b_j}\breve{\mathbf{p}}_{b_i}$ for AAV i can be obtained by projecting AAV i's ground-truth position ${}^{G_i}\mathbf{p}_{b_i}$ into AAV j's body frame using the ground-truth global extrinsic ${}^{G_i}\mathbf{T}_{G_j}$ and the ground-truth ego pose ${}^{G_j}\mathbf{T}_{b_j} = ({}^{G_j}\mathbf{R}_{b_j}, {}^{G_j}\mathbf{p}_{b_j})$ of AAV j. Then, considering the measurement noise $\mathbf{n}_{po,ij} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{po,ij})$ of the passive observation measurement, the measurement model is

$$ {}^{b_j}\breve{\mathbf{p}}_{b_i} = \left({}^{G_j}\mathbf{T}_{b_j}^{-1}{}^{G_i}\mathbf{T}_{G_j}^{-1}\right) \circ {}^{G_i}\mathbf{p}_{b_i} + \mathbf{n}_{po,ij}. \tag{12} $$

Since AAV j's ground-truth pose ${}^{G_j}\mathbf{T}_{b_j}$ is not a part of the state vector $\mathbf{x}_i$, as defined in (5), we similarly utilize the estimated ego pose ${}^{G_j}\breve{\mathbf{T}}_{b_j}$ of AAV j and the covariance $\breve{\mathbf{\Sigma}}_{\mathbf{T}_j}$ received from AAV j, to model the ground-truth pose of AAV j as ${}^{G_j}\mathbf{T}_{b_j} = {}^{G_j}\breve{\mathbf{T}}_{b_j} \boxplus \mathbf{n}_{\mathbf{T}_j}$, where the noise $\mathbf{n}_{\mathbf{T}_j} \sim \mathcal{N}(\mathbf{0}, \breve{\mathbf{\Sigma}}_{\mathbf{T}_j})$. Consequently, the passive observation measurement model is

$$ {}^{b_j}\breve{\mathbf{p}}_{b_i} = \underbrace{\left(({}^{G_j}\breve{\mathbf{T}}_{b_j} \boxplus \mathbf{n}_{\mathbf{T}_j})^{-1}{}^{G_i}\mathbf{T}_{G_j}^{-1}\right) \circ {}^{G_i}\mathbf{p}_{b_i} + \mathbf{n}_{po,ij}}_{\mathbf{h}_{po,ij}(\mathbf{x}_i, \mathbf{n}_{\mathbf{T}_j}, \mathbf{n}_{po,ij}),} \tag{13} $$

which defines a valid measurement equation about the state vector $\mathbf{x}_i$ containing ego position ${}^{G_i}\mathbf{p}_{b_i}$ and global extrinsic ${}^{G_i}\mathbf{T}_{G_j}$. The received teammate pose ${}^{G_j}\breve{\mathbf{T}}_{b_j}$ is known, and $\mathbf{n}_{\mathbf{T}_j}$ and $\mathbf{n}_{po,ij}$ are measurement noises.

To sum up, the entire measurement vector $\mathbf{y}$, the observation function $\mathbf{h}$, and the observation noise $\mathbf{v}$ (the subtract k is omitted for simplification) are, (14) shown at the bottom of this page.

2) Temporal Compensation of Mutual Observation Measurements: For the measurement models (11) and (13) to be valid, the involved states and measurements should be at the same time. However, due to the asynchronous nature of state estimation among different AAVs and the presence of transmission delays, the states and measurements from different AAVs are usually asynchronous. Therefore, it is necessary to compensate for the temporal mismatch between the received measurements or states and the ego state in the measurement models. While the previous work [36] ignores this temporal mismatch, this article carefully addresses this problem based on a constant velocity model.

For the active observation measurement model (11), the measurement ${}^{b_i}\breve{\mathbf{p}}_{b_j}$ is a cluster of points, which are undistorted and projected to the scan end time $t_{i,k}$ (see Section IV-B). The received AAV j's position ${}^{G_j}\breve{\mathbf{p}}_{b_j}$, however, is estimated at time stamp $t_{j,k}$ in AAV j's system time. To make a valid measurement model at time $t_{i,k}$, AAV j's position ${}^{G_j}\breve{\mathbf{p}}_{b_j}$ should be temporally compensated from its time of estimation (i.e., $t_{j,k}$) to the time the measurement model is established (i.e., $t_{i,k}$), according to a constant velocity model from its estimated velocity ${}^{G_j}\breve{\mathbf{v}}_{b_j}$

$$ {}^{G_j}\breve{\mathbf{p}}_{b_j}^{\text{comp}} = {}^{G_j}\breve{\mathbf{p}}_{b_j} + {}^{G_j}\breve{\mathbf{v}}_{b_j}(t_{i,k} - t_{j,k} + {}^{i}\tau_j) \tag{15} $$

which should be substituted into (11) to supply the original measurement ${}^{b_i}\breve{\mathbf{p}}_{b_j}$. The resultant measurement model with temporal compensation is hence

$$ \begin{aligned} {}^{b_i}\breve{\mathbf{p}}_{b_j} = {} & \left({}^{G_i}\mathbf{T}_{b_i}^{-1}{}^{G_i}\mathbf{T}_{G_j}\right) \circ \left({}^{G_j}\breve{\mathbf{p}}_{b_j} \right. \\ & \left. + {}^{G_j}\breve{\mathbf{v}}_{b_j}(t_{i,k} - t_{j,k} + {}^{i}\tau_j) + \mathbf{n}_{\mathbf{p}_j}\right) + \mathbf{n}_{ao,ij} \end{aligned} \tag{16} $$

which is a measurement equation about the state $\mathbf{x}_i$ containing ego pose ${}^{G_i}\mathbf{T}_{b_i}$ and global extrinsic ${}^{G_i}\mathbf{T}_{G_j}$.

For the passive observation measurement model (13), the passive observation measurement ${}^{b_j}\breve{\mathbf{p}}_{b_i}$ is transmitted from AAV j and is estimated at time stamp $t_{j,k}$ of AAV j's system time. To establish a valid measurement model at the time indicated by $t_{j,k}$, all the states and other measurements in (13) should also be at $t_{j,k}$. The received AAV j's state ${}^{G_j}\breve{\mathbf{T}}_{b_j}$ is already stamped with $t_{j,k}$, while the ego position ${}^{G_i}\mathbf{p}_{b_i}$, which is the state at $t_{i,k}$, can be compensated using a constant velocity model as follows:

$$ {}^{G_i}\mathbf{p}_{b_i}^{\text{comp}} = {}^{G_i}\mathbf{p}_{b_i} + {}^{G_i}\mathbf{v}_{b_i}(t_{j,k} - t_{i,k} - {}^{i}\tau_j) \tag{17} $$

which should be substituted into (13) to supply the original state ${}^{b_j}\mathbf{p}_{b_i}$. The temporally compensated measurement model is hence

$$ \begin{aligned} {}^{b_j}\breve{\mathbf{p}}_{b_i} = {} & ({}^{G_j}\breve{\mathbf{T}}_{b_j} \boxplus \mathbf{n}_{\mathbf{T}_j})^{-1}{}^{G_i}\mathbf{T}_{G_j}^{-1} \circ ({}^{G_i}\mathbf{p}_{b_i} \\ & + {}^{G_i}\mathbf{v}_{b_i}(t_{j,k} - t_{i,k} - {}^{i}\tau_j)) + \mathbf{n}_{po,ij} \end{aligned} \tag{18} $$

which is a measurement equation about the state $\mathbf{x}_i$ containing ego position ${}^{G_i}\mathbf{p}_{b_i}$, velocity ${}^{G_i}\mathbf{v}_{b_i}$, and global extrinsic ${}^{G_i}\mathbf{T}_{G_j}$.

3) State and Covariance Update: Based on the LiDAR point measurement model (9) and mutual observation models (11) and (13) with temporal compensation explained previously, we leverage an iterated Kalman filter (ESIKF) [53] to update the state repeatedly. This process will repeat until convergence; then, the optimal state estimation and covariance are obtained. The detailed computation of Kalman gain and update steps can be referred to [11] and [12]. After the update, covariance reinitialization will be implemented following Section V-D for the next round of state estimation.

$$ \begin{aligned} \mathbf{y} &= \left[\underbrace{\ldots, \mathbf{0}, \ldots}_{\text{point measurements}}, \underbrace{\ldots, {}^{b_i}\breve{\mathbf{p}}_{b_j}^T, \ldots}_{\text{active observation measurements}}, \underbrace{\ldots, {}^{b_j}\breve{\mathbf{p}}_{b_i}^T, \ldots}_{\text{passive observation measurements}}\right]^T \\ \mathbf{h} &= \left[\ldots, \mathbf{h}_{p,n}^T, \ldots, \ldots, \mathbf{h}_{ao,ij}^T, \ldots, \ldots, \mathbf{h}_{po,ij}^T, \ldots\right]^T \\ \mathbf{v} &= \left[\ldots, \mathbf{n}_{p,n}^T, \ldots, \ldots, \mathbf{n}_{\mathbf{p}_j}^T, \mathbf{n}_{ao,ij}^T, \ldots, \ldots, \mathbf{n}_{\mathbf{T}_j}^T, \mathbf{n}_{po,ij}^T, \ldots\right]^T. \end{aligned} \tag{14} $$

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

D. Marginalization

The dimension of the state defined in (5) would increase linearly with the swarm size, leading to an almost cubic growth of computation complexity of the ESIKF. To address the problem of explosion of state dimension and computational complexity in the previous work [36], we propose a novel marginalization method. In the flight of aerial swarm systems, due to the restricted detecting range and FoV of the LiDAR sensor, the AAVs are typically unable to observe all teammate AAVs at all times. The observed teammates (either active or passive) have their global extrinsic transformations ${}^{G_i}\mathbf{T}_{G_j}$ persistently excited, as shown in (11) and (13), while others do not. Therefore, we only need to update the global extrinsic of teammate AAVs, which can observe the self-AAV (contributing a passive observation measurement) or are observed by the self-AAV (contributing an active observation measurement). This is achieved by a marginalization operation as follows.

For simplification, we omit the subtract k and i, which represent the kth estimation of AAV i. After receiving the kth LiDAR scan, we identify the mutual observations, as detailed in Section V-C1. Let $\mathcal{A}$ denote the set of teammate AAVs that are observed in the current scan and $\mathcal{B}$ the set of teammate AAVs that are not. Let $\mathbf{x}_1$ represent the substate consisting of the ego state and global extrinsic w.r.t. teammates in the set $\mathcal{A}$, while $\mathbf{x}_2$ represents the complementary state consisting of global extrinsic w.r.t. teammates in the set $\mathcal{B}$. We get $\dim(\mathbf{x}_1) = 18 + 6K$ and $\dim(\mathbf{x}_2) = 6(N-1-K)$, where K represent the number of teammates with mutual observation (i.e., $\dim(\mathcal{A}) = 6K$). Furthermore, in the current round of state estimation, assume $(\widehat{\mathbf{x}}, \widehat{\mathbf{P}})$ as propagated state and covariance after a normal ESIKF prediction step (i.e., Section V-B). Then, they can be partitioned as

$$\mathbf{x} \sim \mathcal{N}\left(\widehat{\mathbf{x}}, \widehat{\mathbf{P}}\right) = \mathcal{N}\left(\begin{bmatrix}\widehat{\mathbf{x}}_1 \\ \widehat{\mathbf{x}}_2\end{bmatrix}, \begin{bmatrix}\widehat{\mathbf{\Sigma}}_{11} & \widehat{\mathbf{\Sigma}}_{12} \\ \widehat{\mathbf{\Sigma}}_{21} & \widehat{\mathbf{\Sigma}}_{22}\end{bmatrix}\right). \tag{19}$$

Since $\mathbf{x}_2$ will not be updated due to the lack of persistent excitation, we marginalize it out from $\mathbf{x}$, leading to the prior distribution of the two substates

$$\mathbf{x}_1 \sim \mathcal{N}(\widehat{\mathbf{x}}_1, \widehat{\mathbf{\Sigma}}_{11}), \quad \mathbf{x}_2 \sim \mathcal{N}(\widehat{\mathbf{x}}_2, \widehat{\mathbf{\Sigma}}_{22}). \tag{20}$$

To update the substate $\mathbf{x}_1$, we notice the measurement model

$$\mathbf{y} = \mathbf{h}(\mathbf{x}, \mathbf{v}) = \mathbf{h}(\mathbf{x}_1, \mathbf{v}_1) \tag{21}$$

where $\mathbf{y}$ includes point measurements and mutual observation measurements (both active and passive), which depend only on $\mathbf{x}_1$. Then, $\mathbf{x}_1$ can be updated by fusing the prior distribution $\mathbf{x}_1 \sim \mathcal{N}(\widehat{\mathbf{x}}_1, \widehat{\mathbf{\Sigma}}_{11})$ with the measurements $\mathbf{y}$ by following the normal ESIKF update step (i.e., Section V-C3). Assume that the updated state estimate and covariance are $\bar{\mathbf{x}}_1$ and $\bar{\mathbf{\Sigma}}_{11}$, respectively. Then, we have $\mathbf{x}_1 \sim \mathcal{N}(\bar{\mathbf{x}}_1, \bar{\mathbf{\Sigma}}_{11})$ and that the substate $\mathbf{x}_2$ still remains at $\mathbf{x}_2 \sim \mathcal{N}(\widehat{\mathbf{x}}_2, \widehat{\mathbf{\Sigma}}_{22})$. Now that $\mathbf{x}_1$ and $\mathbf{x}_2$ are two independent distributions, they should evolve separately in the subsequent ESIKF steps. Specifically, $\mathbf{x}_2$ is subject to its state transition function

$$\mathbf{x}_{2,\tau+1} = \mathbf{x}_{2,\tau} \tag{22}$$

while $\mathbf{x}_1$ is subject to

$$\mathbf{x}_{1,\tau+1} = \mathbf{x}_{1,\tau} \boxplus (\Delta t_\tau \mathbf{f}_1(\mathbf{x}_{1,\tau}, \mathbf{u}_\tau, \mathbf{w}_\tau)) \tag{23}$$

where $\mathbf{f}_1(\mathbf{x}_{1,\tau}, \mathbf{u}_\tau, \mathbf{w}_\tau)$ takes the first $18 + 6K$ elements of $\mathbf{f}(\mathbf{x}_\tau, \mathbf{u}_\tau, \mathbf{w}_\tau)$ in (6).

In the next round of ESIKF, each of the two substate will propagate starting from their respective initial distribution, $\mathbf{x}_1 \sim \mathcal{N}(\bar{\mathbf{x}}_1, \bar{\mathbf{\Sigma}}_{11})$ and $\mathbf{x}_2 \sim \mathcal{N}(\widehat{\mathbf{x}}_2, \widehat{\mathbf{\Sigma}}_{22})$, and following their respective state transition function (23) and (22). This process can be expressed compactly by propagating the complete system following (4) from an initial distribution $\mathbf{x} \sim \mathcal{N}(\bar{\mathbf{x}}, \bar{\mathbf{P}})$ defined as follows:

$$\bar{\mathbf{x}} = \begin{bmatrix}\bar{\mathbf{x}}_1 \\ \widehat{\mathbf{x}}_2\end{bmatrix}, \quad \bar{\mathbf{P}} = \begin{bmatrix}\bar{\mathbf{\Sigma}}_{11} & \mathbf{0} \\ \mathbf{0} & \widehat{\mathbf{\Sigma}}_{22}\end{bmatrix}. \tag{24}$$

Packing the posterior distribution $\mathbf{x}_1 \sim \mathcal{N}(\bar{\mathbf{x}}_1, \bar{\mathbf{\Sigma}}_{11})$ and the prior distribution $\mathbf{x}_2 \sim \mathcal{N}(\widehat{\mathbf{x}}_2, \widehat{\mathbf{\Sigma}}_{22})$ into the joint distribution $\mathbf{x} \sim \mathcal{N}(\bar{\mathbf{x}}, \bar{\mathbf{P}})$ is termed as "covariance reinitialization." With the covariance reinitialization, the propagation of the next step can simply follow the standard ESIKF prediction step of the complete system (4), which is detailed in Section V-B.

E. Degeneration Evaluation

The ESIKF presented previously would update the global extrinsic of teammate AAVs along with the ego state. However, the update is valid only when the LiDAR scan contains sufficient geometric features. In some extreme environments, LiDAR sensors may encounter degeneration where the point cloud fails to provide sufficient constraints to determine its ego pose, making it impossible to distinguish the global extrinsic from ego motion given mutual observation measurements, which is a problem suffered by our previous work [36]. To address this problem, we propose to automatically detect LiDAR degeneration. If it occurs, the previously estimated global extrinsic is used with mutual observation measurements to provide constraints for determining the ego pose. The switching between the two cases (i.e., updating global extrinsic along with ego state and using currently estimated global extrinsic for ego state update) can be achieved automatically by leveraging the marginalization operation as follows.

When LiDAR degeneration occurs, we marginalize all global extrinsic out from the state vector by setting $\mathcal{A}$ to null and $\mathcal{B}$ to the full set of all teammate AAVs. Thus, substate $\mathbf{x}_1$ only includes ego state with $\dim(\mathbf{x}_1) = 18$, and $\mathbf{x}_2$ contains the global extrinsic transformation w.r.t. to all the teammates with $\dim(\mathbf{x}_2) = 6(N-1)$. For the measurement model (21), we rewrite it as

$$\mathbf{y} = \mathbf{h}(\mathbf{x}, \mathbf{v}) = \mathbf{h}(\mathbf{x}_1, \mathbf{x}_2, \mathbf{v}) = \mathbf{h}\left(\mathbf{x}_1, \underbrace{[\mathbf{x}_2, \mathbf{v}]}_{\mathbf{v}_{\text{ext}}}\right) = \mathbf{h}(\mathbf{x}_1, \mathbf{v}_{\text{ext}}) \tag{25}$$

where the marginalized substate $\mathbf{x}_2$ is an exogenous random signal (i.e., it is independent of the state $\mathbf{x}_1$) just like the measurement noise $\mathbf{v}$, so it is grouped with $\mathbf{v}$ to form the

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

extended measurement noise $\mathbf{v}_{\text{ext}}$. The distribution of the "measurement noise" $\mathbf{x}_2$ is obtained by propagating its subsystem following (22). The rest of the steps, including the update of $\mathbf{x}_1$ and the subsequent prediction step, will be identical to that in Section V-D. Besides the aforementioned marginalization, the mutual observation noise $\mathbf{v} = [\mathbf{n}_{ao,ij}, \mathbf{n}_{po,ij}]$ (see Section V-C1) of AAV i would be adjusted to a smaller value to provide adequate constraints for ego pose determination.

To achieve the aforementioned operations, a degeneration evaluation module is required. Inspired by [60] and [61], we evaluate the degeneration situation of AAV i by implementing singular value decomposition of the Jacobian matrix $\mathbf{J_T}$ of $\mathbf{h}_{p,n}(\mathbf{x}_i, \mathbf{0})$ in (9) w.r.t. the ego pose ${}^{G_i}\mathbf{T}_{b_i}$

$$\mathbf{J_T} = \begin{bmatrix} -\mathbf{u}_n^T {}^{G_i}\mathbf{R}_{b_i} \lfloor {}^{b_i}\mathbf{p}_n \rfloor_\wedge & \mathbf{u}_n^T \end{bmatrix} \tag{26}$$

where the notation $\lfloor \mathbf{a} \rfloor_\wedge$ represents the skew-symmetric matrix of vector $\mathbf{a} \in \mathbb{R}^{3\times 1}$ that maps the cross-product operation. By calculating the singular values of $\mathbf{J_T}$, finding the smallest one λ, and comparing it with a predefined degeneration threshold ϵ_d, we can obtain the evaluation result. If $\lambda < \epsilon_d$, AAV i is regarded as encountering LiDAR degeneration, and the corresponding responses mentioned above will be activated for this round of updates.

It is worth mentioning that the proposed method can achieve automatic switchover between the two modes: when degeneration is detected, the global extrinsic transformations and mutual observation measurements are utilized to accurately determine the ego state; when no degeneration occurs, the point cloud measurements of LiDAR are used to refine the global extrinsic states.

F. Broadcast of State Estimation Results

After state estimation completes, the results, including updated ego pose ${}^{G_i}\bar{\mathbf{T}}_{b_i}$, velocity ${}^{G_i}\bar{\mathbf{v}}_{b_i}$, pose covariance $\bar{\mathbf{P}}_i$, and refined global extrinsic transformations ${}^{G_i}\bar{\mathbf{T}}_{G_j}$, are shared with all teammate AAVs through the decentralized ad hoc network. The ego pose and velocity sent to teammates are utilized for their mutual state estimation following (3). The ego pose, pose covariance, and refined extrinsic transformations are sent to teammates to construct their mutual observation measurements (see Section V-C1) for the next step estimation.

Remark 3: The broadcast of the estimation results will also cause the refined global extrinsic transformations to be shared with a new AAV joining the swarm in the middle of a mission. The shared extrinsic will trigger the factor graphs of the new AAV to be updated, by inserting the refined extrinsic transformations received from the network. Optimizing the factor graph will then obtain the extrinsic between the new AAV and existing swarms. On the other hand, the shared extrinsic transformations will not trigger any factor graph update of existing AAVs in the swarm, as this edge has already existed in the factor graph.

VI. Simulation Evaluation

In this section, we conducted simulation experiments to evaluate the performance of the Swarm-LIO2 framework.

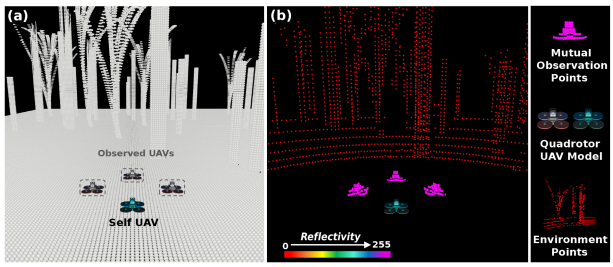

Fig. 6. Illustration of MARSIM simulator and the corresponding rendered point cloud. (a) Multi-AAV simulator scenario. (b) Rendered point cloud of self-AAV colored by each point's reflectivity.

TABLE I
Total Flight Distance in Initialization

Distance (m) \ Size / Method	5	10	15	20	30	40
Swarm-LIO2	**23.2**	**23.2**	**23.2**	**23.2**	**30.7**	**43.5**
Swarm-LIO	120.5	243.8	367.1	496.6	763.1	1032.2

A. Simulator Setup

In our simulation experiments, we utilize the MARSIM simulator [62], a lightweight point-realistic simulator for LiDAR-based AAVs. As shown in Fig. 6, MARSIM supports a variety of common LiDAR models, and we select Livox LiDAR sensors, including Livox Avia and Livox Mid360, to maintain consistency with the real-world experimental setup. It is worth mentioning that MARSIM is capable of simulating the mutual observation scenarios among multiple AAVs, which is essential for validating the method proposed in this article. To simulate the scenario where each AAV, in reality, is equipped with reflection tapes, the reflectivity of the mutual observation points observed by each AAV is set to large saturated values. In all simulation experiments, the simulator is running on a laptop with i9-12900H CPU and NVIDIA GeForce RTX 3080 Ti GPU, and the LiDAR scan rate is set to 10 Hz.

B. Initialization Efficiency and Accuracy Evaluation

A key step in the initialization of an aerial swarm is the global extrinsic calibration. In our previous work, Swarm-LIO [36], the identification and global extrinsic calibration are achieved solely through trajectory matching, necessitating each AAV to fly a certain trajectory. Each AAV performing this initialization trajectory in turn will lead to successive long flight distances, especially when the swarm size is large. In contrast, the decentralized pose graph optimization in Swarm-LIO2 requires only one AAV to fly a certain trajectory that can be observed by teammate AAVs, significantly reducing the initialization complexity.

We validate the initialization efficiency by comparing it to Swarm-LIO [36] at swarm size varying from 0 to 40. At each swarm size, for Swarm-LIO2, only one AAV executes a figure-8 trajectory that can be observed by the rest AAVs. For Swarm-LIO, each drone needs to fly the figure-8 trajectory in other AAVs' FoV. Table I shows the total flight distance for all AAVs in the initialization at different swarm sizes. As can be seen, as the

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

TABLE II
INITIALIZATION ACCURACY COMPARISON

RMSE	Size / Algorithm	5	10	15	20	30	40
Trans (m)	**Swarm-LIO2**	**0.1035**	0.1206	**0.1138**	**0.1395**	0.1323	0.1547
	Swarm-LIO	0.1088	**0.1193**	0.1195	0.1440	**0.1264**	**0.1496**
Rot (rad)	**Swarm-LIO2**	**0.0623**	0.0739	0.0684	**0.0717**	0.0860	0.0848
	Swarm-LIO	0.0652	**0.0698**	**0.0644**	0.0762	**0.0813**	**0.0821**

swarm size increases, the total flight distance in [36] increases linearly, while that of Swarm-LIO2 increases very slowly and nearly remains unchanged regardless of the number of AAVs. This indicates that the proposed method effectively mitigates the need for individual AAVs to fly extensive distances during initialization, compared to [36]. This contributes to significant energy savings and increased effective operational flight time for the swarm system.

We also evaluate the initialization accuracy of Swarm-LIO2 and Swarm-LIO [36] using their respective initialization trajectories (i.e., only one AAV flies a figure-8 trajectory for Swarm-LIO2 versus all AAVs fly figure-8 trajectories for Swarm-LIO). By comparing the root-mean-square error (RMSE) of the global extrinsic transformations obtained by Swarm-LIO2 with Swarm-LIO [36] at swarm size varying from 0 to 40, it can be observed from Table II that the two methods have similar initialization accuracy, which means that the proposed factor graph optimization nearly does not deteriorate the initialization accuracy.

C. State Estimation and Global Extrinsic Accuracy Evaluation

Swarm-LIO2 can achieve robust accurate ego state and mutual state estimation and provide effective global extrinsic transformation. This capability is indispensable for various swarm applications, such as multi-AAV formation flight, mutual collision avoidance, collaborative exploration, etc. As explained in Section V-A, the mutual state estimation is robust to mutual observation loss. To evaluate such performance, the simulation experiments are conducted in a randomly generated 3-D forest-like scenario of dimension $60 \times 40 \times 8\ \mathrm{m}^3$ with a swarm composed of five AAVs (see Fig. 7). In this evaluation, each AAV needs to perform ego state estimation as well as mutual state estimation of the other four AAVs in the simulated forest. After the swarm initialization, the AAVs fly through the forest from one side to the other, causing frequent mutual observation losses between any two AAVs due to the dense obstacles. As shown in Fig. 7 in which AAV1 is selected as the self-AAV, despite the frequent mutual observation losses caused by occlusions, the ego trajectory and teammate trajectories estimated by Swarm-LIO2 on AAV1 can maintain smoothness and continuity. We also transform the point cloud maps constructed by each AAV to the global frame of AAV1, using the estimated global extrinsic transformations. As can be seen, the merged point cloud map maintains a high level of consistency, which qualitatively showcases the excellent accuracy of the global extrinsic estimation.

For quantitative evaluation, we compute the error (RMSE) of all the estimated AAV trajectories by comparing them to

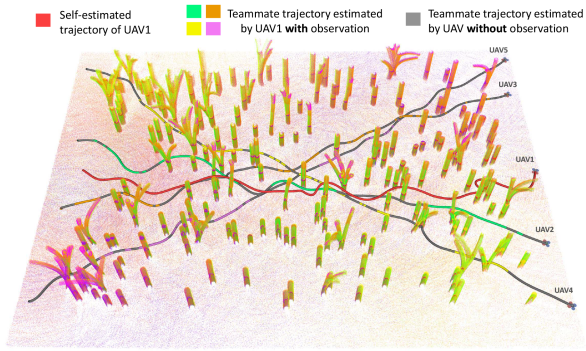

Fig. 7. Estimated ego trajectory and teammate trajectories on AAV1 in a five-AAV swarm system. The point cloud map is generated in a postprocessing stage where the maps of different AAVs are merged using the estimated global extrinsic transformations.

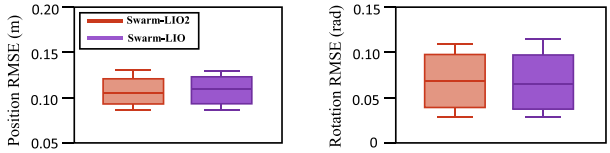

Fig. 8. Error (RMSE) distribution of the five AAVs' trajectories estimated by different methods.

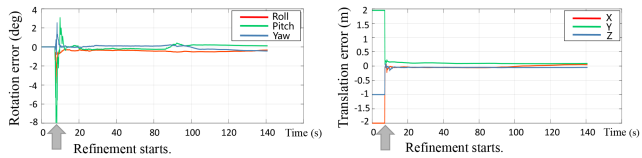

Fig. 9. Global extrinsic estimation error of AAV1 w.r.t. AAV2 (i.e., ${}^{G_1}\mathbf{R}_{G_2}$ and ${}^{G_1}\mathbf{p}_{G_2}$).

the ground truth offered by the simulator. Since each of the N AAV trajectories is estimated N times by itself and the rest of the teammates, we compute the RMSE of all the N^2 estimated trajectories and compare them with Swarm-LIO [36]. The distribution of all the N^2 RMSE, separated by position and rotation, is illustrated in Fig. 8. It can be observed that compared to Swarm-LIO, Swarm-LIO2 achieves a similar accuracy despite the introduced marginalization operations.

Finally, for the global extrinsic estimation, Fig. 9 shows the initialization and online refinement of the global extrinsic transformation; we select AAV1 and AAV2 for analysis and depict the error of the estimated global extrinsic ${}^{G_1}\mathbf{R}_{G_2}$, ${}^{G_1}\mathbf{p}_{G_2}$, in which the ground truth is provided by the simulator. It can be seen that the estimation error gradually converges during the online refinement, and the final error of the global extrinsic is less than 1° (for rotation) and 0.2 m (for translation). With the accurate global extrinsic, we can merge the point cloud map produced by different AAVs, which is extremely useful for large-scale collaborative mapping. As shown in Fig. 7, all the point cloud maps (points of teammate AAVs are filtered as they are dynamic) are transformed into AAV1's global frame using the estimated global extrinsic transformations. The consistently aligned map

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

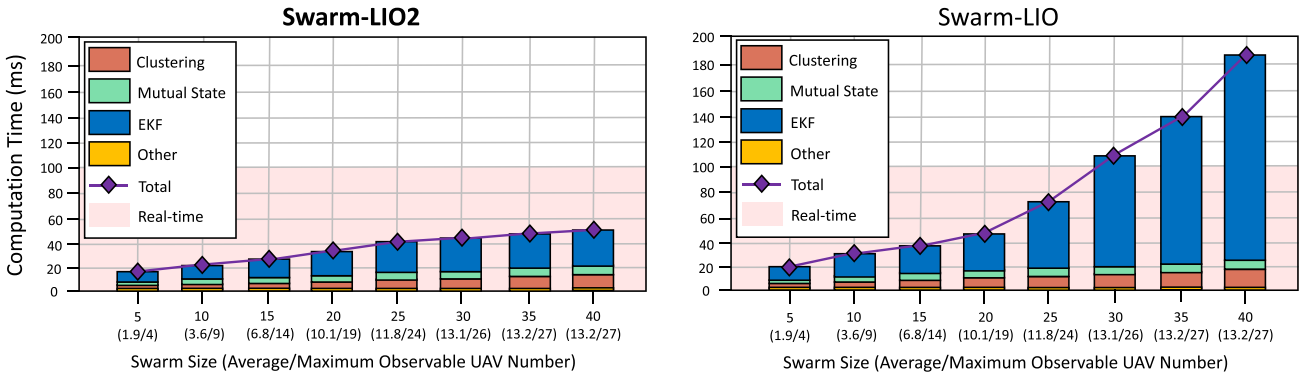

Fig. 10. Overall and module-specific computation times per LiDAR scan under swarm scales varying from 0 to 40. The x-label shows the swarm size along with the average/maximum teammates observed by any AAV in the swarm, which are shown in the parenthesis.

shown in Fig. 7 indicates the accurate global extrinsic estimation of Swarm-LIO2.

D. Scalability Analysis in Time Consumption and Communication Bandwidth

To validate that Swarm-LIO2 possesses high scalability and can maintain efficient computation even at a large swarm size, a comparative experiment is conducted in a sparse simulated 3-D forest-like scenario. We compare the time consumption of Swarm-LIO2 to Swarm-LIO [36] at different swarm sizes. The entire framework of each method can be partitioned into several modules, including point clustering, mutual state estimation, ESIKF-based state estimation, etc. We analyze the time consumption of each module of the two methods, and the results are shown in Fig. 10.

As can be seen, the computation time of clustering and mutual state estimation in the two methods linearly increases as the swarm size increases. This is because these two submodules need to be performed for every teammate AAV in the swarm system. Moreover, since Swarm-LIO2 employs fast Euclidean clustering (FEC) [56] for clustering, which is extremely faster compared to traditional Euclidean clustering provided by the PCL library used in Swarm-LIO, the overall time consumption of clustering in Swarm-LIO2 is lower than that in Swarm-LIO. For the ESIKF-based state estimation module, its time complexity is cubic to the state dimension in theory. In Swarm-LIO [36], the state contains the ego state and the global extrinsic transformations of all teammates, leading to a time consumption rapidly increasing with the swarm size. By contrast, in Swarm-LIO2, the state only includes the ego state as well as the global extrinsic of observed teammates or teammates observing the self-AAV, which often saturates at a relatively small number due to mutual occlusions and LiDAR FoV limit (see Fig. 10). As a result, as the swarm size increases, the time consumption of Swarm-LIO2 increases sublinearly and at a rather low rate, even exhibiting a saturation trend when the swarm size reaches a certain size. To sum up, Swarm-LIO2 is highly scalable compared with Swarm-LIO in terms of time consumption, reducing the total consumed time by 7.83, 31.65, and 133.09 ms at swarm sizes of 10, 25, and 40, respectively.

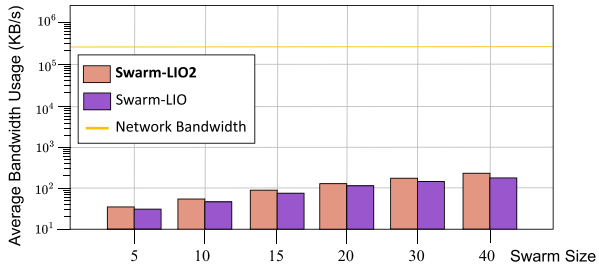

Fig. 11. Average transmitting bandwidth usages (kB/s) of Swarm-LIO2 and Swarm-LIO at different swarm sizes.

In addition to computational time consumption, the communication overhead might be another bottleneck that prevents the swarm scale from growing. Therefore, it is crucial to evaluate the communication bandwidth usage under different swarm scales. We count the average data transfer volume per second, which is the average transmitting bandwidth usage, of Swarm-LIO2 and the previous version Swarm-LIO [36], at the swarm size varying from 0 to 40. From the results shown in Fig. 11, it can be observed that the average bandwidth usages of both methods increase linearly as the swarm size grows, but still remains at a low level since all the information to be communicated is of low dimension. When the swarm size is 40, the bandwidth usage of Swarm-LIO2 is below 250 kB/s. Compared to the theoretical bandwidth of the Intel Wi-Fi 6E AX211 (Gig+)[1] adapter used in our real systems, which is 2.4 Gb/s (approximately 300 MB/s), the actual bandwidth usage of Swarm-LIO2 is almost negligible, indicating that the bandwidth is not a bottleneck at all. Besides, the transmitting bandwidth usage of Swarm-LIO2 is slightly larger than that of Swarm-LIO because there is additional information (e.g., global extrinsic transformation) to be exchanged in Swarm-LIO2.

E. Fly Through a Degenerated Corridor

In this section, we conduct a simulation experiment in which five AAVs equipped with Livox Mid360 LiDARs need to fly

[1][Online]. Available: https://www.intel.cn/content/www/cn/zh/products/sku/204837/intel-wifi-6e-ax211-gig/specifications.html

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

TABLE III
STATE ESTIMATION ACCURACY UNDER DIFFERENT PLRS

Average RMSE \ PLR(%)	0	25	50	75	100
Position (m)	0.0754	0.0772	0.0851	0.0882	0.0865
Rotation (rad)	0.0446	0.0489	0.0515	0.0526	0.0523

Note: the average RMSE is not calculated for the time in which the corresponding teammate is marked as "disconnected," during which the teammate state is not estimated.

through a degenerated corridor. In this case, the measurements of a single LiDAR cannot provide sufficient constraints for pose determination, but Swarm-LIO2 can perform robust and stable state estimation thanks to the mutual observation measurements from teammates. We compare the localization accuracy of our method to Swarm-LIO and some state-of-the-art LiDAR-inertial odometry for a single-AAV system, and the results showcase the superior robustness of Swarm-LIO2 to degenerated scenes. Due to the page limit, we put the detailed descriptions, illustrations, qualitative, and quantitative results in the supplementary material in [63].

F. Localization Accuracy With Communication Loss

The wireless communication is assumed to be perfect in all the previous simulation tests. However, in reality, communication issues like dropouts are inevitable since various interference sources, e.g., electromagnetic interference and physical occlusions would impact communication stability. In the case of communication loss, Swarm-LIO2 would still hold the connection status for 2 s more (see Section IV-A), during which the teammates' states are predicted via (3) using the constant velocity model and the last updated extrinsic. Once the teammate connection status changes to "disconnected," the teammate states will no longer be estimated until the teammate is reconnected. Holding the connection status for 2 s more can effectively reduce the false alarm caused by temporary communication loss such as temporary network congestion. Regardless of the communication loss, Swarm-LIO2 can reliably estimate the ego state based on the measurements of LiDAR and IMU. In the case of complete communication loss, Swarm-LIO2 would degrade to FAST-LIO2 [12] to estimate the ego state only.

To validate the robustness of Swarm-LIO2 to communication loss, we evaluate the state estimation accuracy on a swarm composed of five AAVs in the simulation environment shown in Fig. 7, under different simulated packet loss rates (PLRs). We evaluate the accuracy by averaging the RMSEs of the N^2 trajectories, which are shown in Table III. As can be seen, as the PLR increases, the localization accuracy of Swarm-LIO2 does not deteriorate obviously, which clearly illustrates the remarkable robustness of Swarm-LIO2 to communication dropouts.

VII. REAL-WORLD APPLICATIONS

To comprehensively demonstrate the properties of the proposed swarm state estimation method and its capability to support different applications, we conduct various experiments in real-world environments.

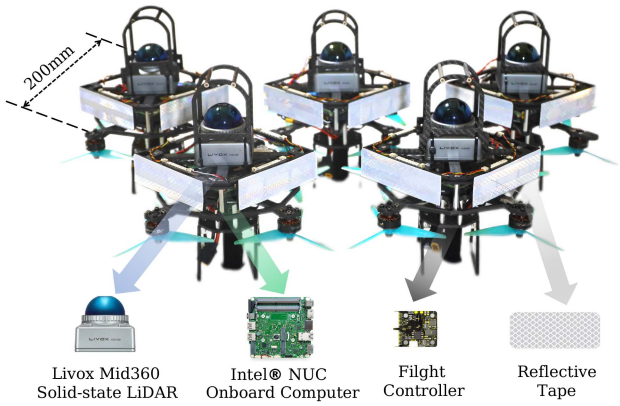

Fig. 12. AAV platform of the proposed swarm system. Each AAV is equipped with a 3-D LiDAR, a flight controller with built-in IMU, and an onboard Intel NUC computer. Several reflective tapes are attached to the AAVs for teammate detection.

A. Hardware Platform

The experiment platform is a compact and cost-effective quadrotor AAV that is equipped with 3-D LiDAR and IMU sensors. The quadrotor AAV has a 280-mm wheelbase and is equipped with a Livox Mid360 LiDAR. The LiDAR is capable of generating point clouds at a rate of 200 000 points per second and possesses $360^\circ \times 59^\circ$ FoV. As for the computation unit, each AAV is equipped with an onboard Intel NUC computer featuring an i7-1260P CPU, coupled with a flight controller that provides over 200-Hz IMU measurements. In all the real-world experiments, the LiDAR scan rate is 30 Hz. Each AAV is attached with reflective tapes for easy detection. The spatiotemporal extrinsic of the LiDAR and IMU are precalibrated with [64]. The hardware platform of our swarm system is shown in Fig. 12.

B. Inter-AAV Collision Avoidance

This experiment emulates a dense air traffic scenario by flying five AAVs in interleaved directions [see Fig. 13(a1) and (b1)]. Two flight tasks are demonstrated: in the first one, five AAVs initially hovering at the five vertices of a pentagon need to fly to a target position on the opposite side of the pentagon. In the second one, five AAVs initially hovering on one side of a field need to reach the other side of the field, meanwhile interchanging their positions. In both tasks, Swarm-LIO2 serves as the infrastructure for swarm initialization and swarm state estimation, which provides accurate global extrinsic transformations and real-time mutual state for inter-AAV collision avoidance. The inter-AAV collision avoidance is achieved by a swarm planner modified from [27] and [28]. The planned trajectories are fed into the motion controller [19] for execution.

The composite snapshots illustrating the entire flight process are shown in Fig. 13(a1) and (b1), with the estimated trajectories of each AAV shown in Fig. 13(a2), (a3), (b2), and (b3). It can be observed that the estimated trajectories highly match the actual flights in the composite snapshots, which qualitatively validate the accuracy of Swarm-LIO2. We also analyze the

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

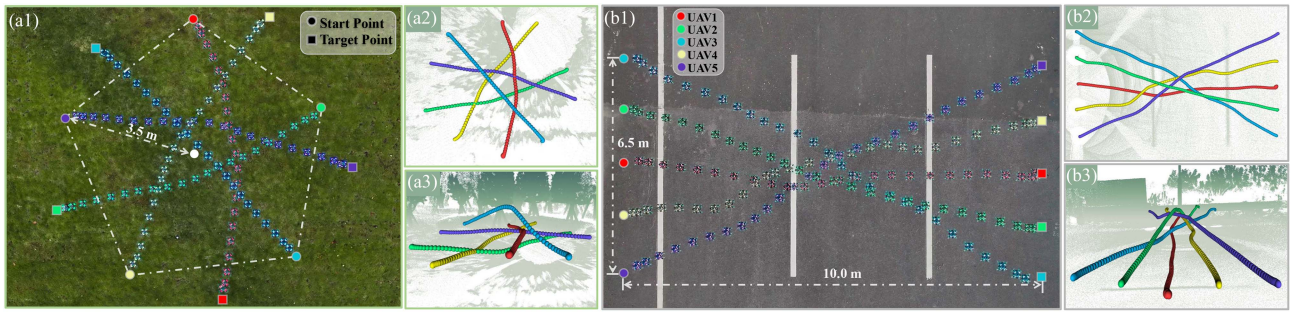

Fig. 13. (a1) and (b1) Composite image of inter-AAV collision avoidance experiments, in which different colors represent different AAVs. Five AAVs first hover above the vertices of a regular pentagon (a1) or in a straight line (b1) and then accomplish collision-free flight using state estimation from Swarm-LIO2. (a2), (a3), (b2), and (b3) Estimated trajectories are visualized in different colors.

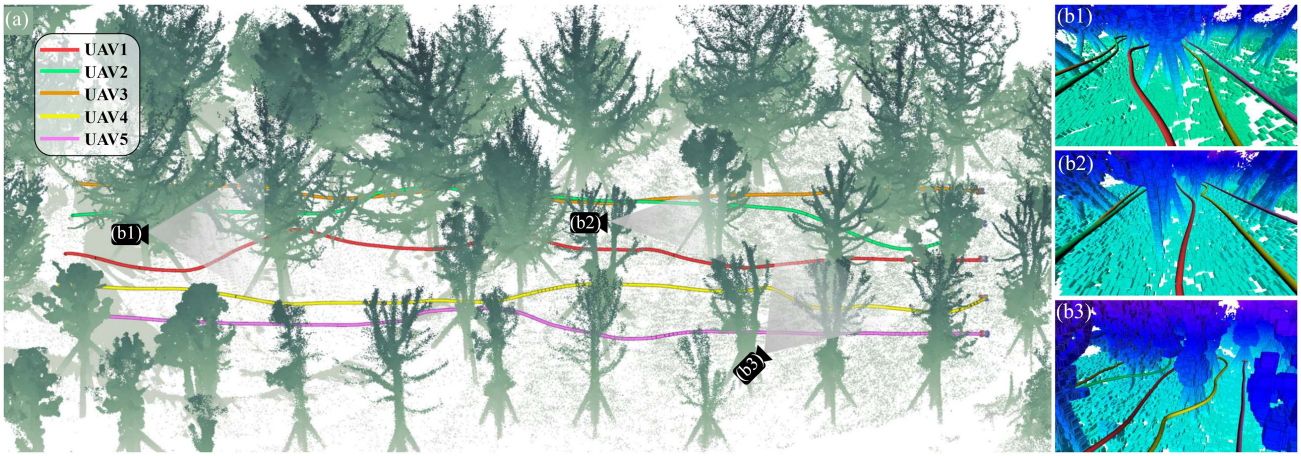

Fig. 14. (a) Swarm system with five AAVs flying through a dense forest; the illustrated ego state and the mutual state are estimated by AAV1. (b1)–(b3) Some details of the estimated trajectories when the AAVs avoid obstacles.

state estimation result quantitatively, which is demonstrated in Section VII-G.

C. Fly Through a Dense Forest

To validate the performance of Swarm-LIO2 in cases of mutual observation loss, we conduct a test in a dense forest environment using the five AAVs, as shown in Fig. 12. Each AAV needs to fly through a dense forest and reach each AAV's target point, which is 40 m away from the start point. During the whole process, no collision with obstacles in the environment or with teammate AAVs is allowed.

The initialization, goal transformation, and trajectory planning are the same as those in Section VII-B. Then, all the AAVs start to fly through the forest from one side to the other, shown in Fig. 14(a). During the flight, the dense trees lead to frequent mutual observation losses, while Swarm-LIO2 can still achieve robust and smooth mutual state estimation. The trajectories of the five AAVs and the point cloud of the forest are depicted in Fig. 14(a). The red trajectory represents the self-estimated flight trajectory of AAV1, while the green, orange, yellow, and purple trajectories represent the other AAVs' mutual state estimation results estimated by AAV1 in its respective global frame. Some details of the estimated trajectories when the AAVs avoid obstacles are illustrated in Fig. 14(b1)–(b3). Throughout the entire mission, Swarm-LIO2 provides accurate real-time state estimation results (see the quantitative analysis in Section VII-G) for the planning and control modules to achieve collision-free flights.

D. Target Tracking With Dynamic Joining and Leaving

To validate the plug-and-play property of Swarm-LIO2, which supports dynamic teammates joining and leaving, we conducted a collaborative target-tracking experiment with four AAVs. To enable fast detection of the target, the person being tracked wears a high-reflectivity vest, so his position can be easily detected from the high-reflectivity points from each AAV's LiDAR measurements. All AAVs have the same preprogrammed task: detecting and tracking a target, characterized by its high reflectivity and certain size, in a collaborative manner with teammates (if any) to maximize the overall target visibility, meanwhile avoiding the static and dynamic obstacles in the environment. In this process, Swarm-LIO2 serves as an infrastructure for automatic teammate finding, identification, and

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

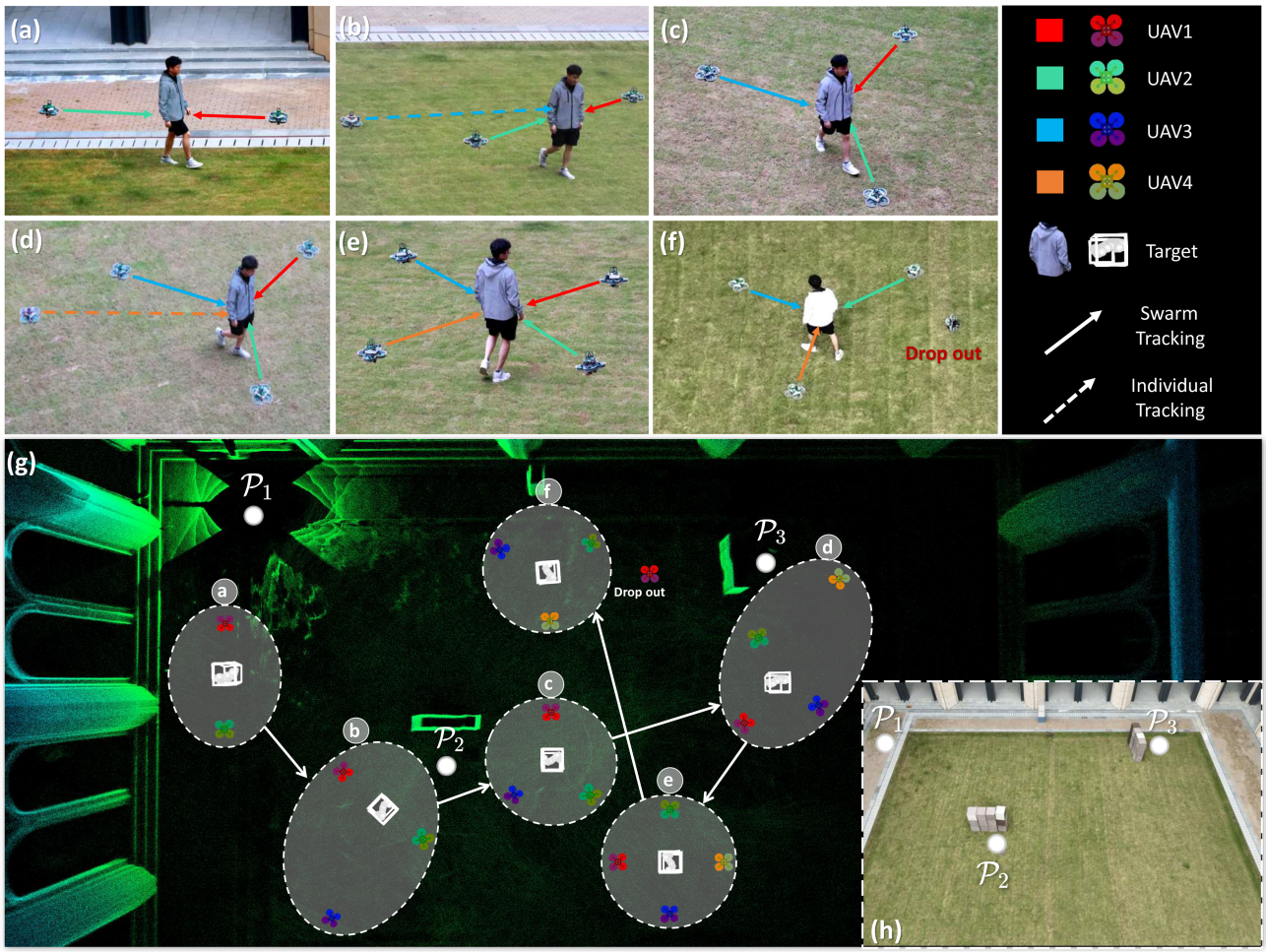

Fig. 15. Collaborative target tracking experiment in an outdoor environment. Each AAV in the swarm system estimates its own and teammate AAVs' states by Swarm-LIO2. (a) Swarm only contains two members who track the target in a straight line. (b) AAV3 detects the target and tracks it in the individual tracking mode and avoids other AAVs by treating them as dynamic obstacles. (c) AAV3 successfully joins the swarm after the online initialization, the formation changes into a triangle. (d) AAV4 detects the target, tracks the target, and tries to join the swarm. (e) AAV4 successfully joins the swarm; the formation changes to a square. (f) AAV1 is shut down intentionally. The formation transits back to a triangle, demonstrating the robustness of Swarm-LIO2 to single point of failure. It is noted that different from the previous pictures captured by a ground camera, this picture is taken from the air, which has a different color for the person's coat due to different camera parameters. (g) Top-down view of the experiment site and the illustration of the entire swarm tracking application in Rviz. (h) Aerial view of the experiment site.

mutual state estimation, while trajectory planning is achieved by a decentralized swarm tracker in our previous work [7].

Before the mission starts, AAV1 and AAV2 are placed at the same area $\mathcal{P}_1$, as shown in Fig. 15(g) and (h), where they can communicate well and are commanded to complete the swarm initialization by flying one AAV along a figure-8 trajectory. After the initialization, the two AAVs form a swarm system of size two. AAV3 and AAV4 are placed separately in different locations $\mathcal{P}_2$ and $\mathcal{P}_3$, respectively, where they cannot detect the target due to occlusion.

The mission starts when the target enters the area $\mathcal{P}_1$, where AAV1 and AAV2 successfully detect the target and start to track it collaboratively. To maximize the target visibility, the two AAVs form a straight line with the target in the middle, as shown in Fig. 15(a). AAV3 and AAV4 are actively searching for the target but did not find one due to occlusions and FoV limit; hence, they remain at their respective initial position.

Subsequently, the target moves to the area $\mathcal{P}_2$, where it is detected by AAV3. Then, AAV3 takes off and starts to track the target. Since AAV3 is not yet part of the swarm (it has neither been identified as a teammate by AAV1 and AAV2 nor the global extrinsic transformations are calibrated), it tracks the target in a solo manner by treating AAV1 and AAV2 as dynamic obstacles to avoid, as shown in Fig. 15(b). Similarly, AAV1 and AAV2 remain in their current formation, which is the optimal one for the tracking mission, while treating AAV3 as a dynamic obstacle to avoid. As a consequence, the swarm of AAV1 and AAV2 and the individual AAV3 both execute their preprogrammed task, although not in a collaborative manner due to the lack of a prior initialization.

As the tracking goes on, the AAV3 would fly a trajectory, during which its identity, temporal offset, and global extrinsic w.r.t. any of AAV1 and AAV2 can be estimated by Swarm-LIO2 on the fly. With the online initialization, AAV3 joins the swarm,

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

forming a swarm system of size three. The new swarm system starts changing its form into a triangle shape, with the target in the center, to maximize the tracking visibility [see Fig. 15(c)]. This process takes place automatically without interrupting the tracking task. When the target approaches area $\mathcal{P}_3$, where AAV4 is placed, the swarm size increases further to four and the AAVs form the shape of a square for maximizing the tracking visibility [see Fig. 15(d) and (e)].

Finally, AAV1 is killed intentionally to emulate a scenario in which one agent in the swarm experiences failures. Swarm-LIO2 on each remaining AAV can detect the teammate dropout, update its teammate list, and estimate the rest of teammate states all automatically, indicating that Swarm-LIO2 is robust to single point of failure [see Fig. 15(f)]. Correspondingly, the planner quickly transforms the formation back into a triangle shape [see Fig. 15(f)].

To sum up, in the entire target-tracking process, Swarm-LIO2 can conduct initialization on the fly, discover newly joined teammates or dropout teammates dynamically, and estimate the ego state and mutual state in real time; all take place automatically without interrupting the tracking task. This enables the swarm to adapt its formations to optimize task completion. Moreover, Swarm-LIO2 can provide consistent and accurate state estimation results throughout the entire mission (see quantitative results in Section VII-G), ensuring excellent target-tracking performance. To validate that Swarm-LIO2 possesses applicability to various environments, we also experimented with an indoor setting and a low-light night setting (see the attached video online[2]).

E. More Experiments

We conduct two more real-world experiments to highlight the superior robustness and broad applicability of Swarm-LIO2. In the first experiment, two AAVs equipped with different LiDARs fly in a degenerated scenario. With mutual observation constraints provided by Swarm-LIO2, the two AAVs can achieve centimeter-level localization accuracy. In the second experiment, the accurate mutual state estimation and global extrinsic calibration capability of Swarm-LIO2 enables three AAVs to transport a payload cooperatively from an outdoor scene into a building. Due to the page limit, we put the detailed descriptions, illustrations, and quantitative results in the supplementary material in [63].

F. Time Consumption Analysis

In this section, we evaluate the average computational time per LiDAR scan (unit: ms) of the aforementioned real-world experiments (from Section VII-B to supplementary [63]), tested on the onboard computer NUC equipped with an Intel i7-1260P CPU. We compare the time consumption of Swarm-LIO2 with FAST-LIO2 [12] (an efficient single-LiDAR LIO system), LiLi-OM [65] (an optimization-based single-LiDAR LIO system), and our previous work Swarm-LIO [36]. The average computational time of different methods in the different experiments is shown in Table IV.

TABLE IV
AVERAGE TIME CONSUMPTION PER SCAN (MS)

Method	VII-B	VII-C	VII-D	Sup-II	Sup-III	Average
FAST-LIO2	5.28	5.49	7.23	8.10†	5.31	6.28
Swarm-LIO2	6.74	6.87	8.33	9.13	6.76	7.57
Swarm-LIO	10.29	10.83	11.36	10.72	7.02	10.04
LILI-OM	×	×	26.36	32.68†	22.35	27.13

† denotes the case where UAV2 (equipped with s small FoV LiDAR) fails due to LiDAR degeneration; thus, the time consumption is obtained from UAV1 only for single-LiDAR LIO methods.
× denotes that LiLi-OM fails to extract enough features for optimization in the forest scene.

As can be seen, Swarm-LIO2 improves the computation efficiency significantly when compared to Swarm-LIO, due to the introduced state marginalization. In the experiment shown in Supplementary III in [63], the time consumption of Swarm-LIO2 and Swarm-LIO are at a similar level because the three AAVs need to remain close to each other, leading to full mutual observation during the entire process. Therefore, the reduction in computation time caused by marginalization is not significant, and the slight reduction is primarily attributed to the more efficient point clustering algorithm FEC [56]. However, in the other four experiments, since the mutual observation is frequently lost due to occlusions, the computational time of Swarm-LIO2 is obviously less than that of Swarm-LIO mainly due to the proposed marginalization operation. Compared to LiLi-OM, Swarm-LIO2 consumes much less time since it avoids time-consuming feature extraction and utilizes the efficient ESIKF framework. Compared to FAST-LIO2, despite Swarm-LIO2 incorporating many additional modules and handling more complex problems, it only incurs approximately 20% more computation time on average.

Finally, the LiDAR scan rate is 30 Hz, indicating that the limit for real-time computation is approximately 33.33 milliseconds per frame. In all the real-world experiments, the computational time of Swarm-LIO2 is far less than the limit value, showcasing the excellent real-time performance of Swarm-LIO2.

G. Quantitative Analysis of State Estimation

In this section, we quantitatively evaluate the state estimation consistency of Swarm-LIO2 in the aforementioned real-world experiments. Since in most real-world experiments, no ground truth of AAVs' states can be obtained, for a swarm system containing N AAVs, we compute the standard deviation of all the N^2 estimated AAV trajectories in each experiment to quantitatively evaluate the state estimation consistency.

The computed standard deviations of rotation and translation estimated by Swarm-LIO2 and Swarm-LIO [36] are illustrated in Table V. As can be seen, the standard deviation of position and rotation is at the centimeter level and the degree level, respectively, indicating the excellent consistency of the swarm state estimation (both ego and mutual) of Swarm-LIO2. From the comparison with Swarm-LIO, it is evident that the two methods have similar state estimation consistency, indicating that the

[2][Online]. Available: https://youtu.be/Q7cJ9iRhlrY

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

TABLE V
STANDARD DEVIATION OF ALL ESTIMATED STATES

Error	Method	VII-B	VII-C	VII-D	Sup-II	Sup-III	Average
Pos (m)	**Swarm-LIO2**	0.0539	**0.0515**	**0.0349**	**0.0256**	0.0564	**0.0445**
	Swarm-LIO	**0.0537**	0.0518	0.0383	0.0291	**0.0553**	0.0456
Rot (rad)	**Swarm-LIO2**	**0.0631**	**0.0626**	**0.0570**	**0.0335**	0.0698	**0.0572**
	Swarm-LIO	0.0632	0.0629	0.0596	0.0352	**0.0670**	0.0576

TABLE VI
AVERAGE COMMUNICATION BANDWIDTH USAGE

Method	Exp	VII-B	VII-C	VII-D	Sup-II	Sup-III
	Swarm Size	5	5	4	2	3
Swarm-LIO2	TX (KB/s)	31.98	33.89	22.85	8.67	16.33
	RX (KB/s)	27.97	27.43	19.75	7.94	14.97
Swarm-LIO	TX (KB/s)	25.54	25.98	18.48	6.52	14.52
	RX (KB/s)	21.86	21.32	15.66	5.93	12.89

marginalization operations adopted in Swarm-LIO2 nearly do not impact the performance.

H. Communication Bandwidth Analysis

We quantitatively evaluate the data transfer volume (TX and RX) per second of each AAV in the aforementioned real-world experiments. The results are shown in Table VI. Both systems have extremely low average bandwidth usage, which is under 35 kB/s when the swarm system contains five AAVs. In all the real-world experiments of Swarm-LIO2, the adopted wireless network adapter on each AAV is Intel Wi-Fi 6E AX211 (Gig+), which theoretically possesses 2.4 Gb/s (approximately equals to 300 MB/s) bandwidth, four orders of magnitude higher than the actual usage. Compared with Swarm-LIO, Swarm-LIO2 consumes a slightly larger bandwidth since it exchanges more information, including global extrinsic transformations and degeneration status.

We also calculated the PLR of the two methods. The average PLR is 11.08% for Swarm-LIO2 and 10.36% for Swarm-LIO, which are similar. The accurate state estimation, as shown previously in the presence of the packet loss, indicates the excellent robustness of our system.

VIII. CONCLUSION AND FUTURE WORK

In this article, we proposed *Swarm-LIO2*, a decentralized efficient state estimation framework, based on LiDAR and IMU measurements for aerial swarm systems. A decentralized temporal calibration approach was utilized to calibrate the inter-AAV temporal offset. Novel reflective tape-based AAV detection, trajectory matching, and factor-graph-optimization-based methods were proposed to perform efficient and fast teammate identification and global extrinsic calibration. A novel marginalization module was proposed to reduce the state dimension and further improve the swarm scalability, and a degeneration evaluation module was presented to ensure robust ego state determination. Furthermore, we introduced the elaborate measurement modeling and temporal compensation of the mutual observation measurements, enhancing our state estimator's accuracy and consistency. By exchanging bandwidth-efficient information via an ad hoc network, the mutual observation measurements are tightly coupled with IMU and point cloud measurements under an ESIKF framework, fulfilling real-time accurate ego state mutual state estimation. Using simulation benchmarks, we compared our LiDAR-inertial odometry with other state-of-the-art LIO methods, demonstrating excellent robustness to LiDAR degenerated scenarios. In addition, the analysis of computational time and communication bandwidth usages at different swarm scales showcases the superior scalability of our method. Besides, we integrated our method into a AAV swarm composed of at most five AAVs with fully autonomous state estimation, planning, and control modules. Various simulated and real-world experiments were conducted, demonstrating that our method serves as an infrastructure for aerial swarm systems and can support a wide range of AAV swarm applications.

In the future, we will focus on extending Swarm-LIO2 to a more complete swarm SLAM system by incorporating loop closure modules and historical pose correction, to ensure low drift after a long time of running.

ACKNOWLEDGMENT

The authors would like to thank Yang Jiao, Wendi Dong, and Meng Li for helpful discussion and Prof. Ximin Lyu for the experiment site support. The authors would also like to thank Livox Technology for equipment support.

REFERENCES

[1] Y. Gao et al., "Meeting-merging-mission: A multi-robot coordinate framework for large-scale communication-limited exploration," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 13700–13707.
[2] B. Zhou, H. Xu, and S. Shen, "RACER: Rapid collaborative exploration with a decentralized multi-UAV system," *IEEE Trans. Robot.*, vol. 39, no. 3, pp. 1816–1835, Jun. 2023.
[3] B. Tang et al., "Bubble explorer: Fast UAV exploration in large-scale and cluttered 3D-environments using occlusion-free spheres," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 1118–1125.
[4] P. Zhu, J. Zheng, D. Du, L. Wen, Y. Sun, and Q. Hu, "Multi-drone-based single object tracking with agent sharing network," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 4058–4070, Oct. 2021.
[5] R. Bonatti et al., "Autonomous aerial cinematography in unstructured environments with learned artistic decision-making," *J. Field Robot.*, vol. 37, no. 4, pp. 606–641, 2020.
[6] R. Olfati-Saber and P. Jalalkamali, "Collaborative target tracking using distributed Kalman filtering on mobile sensor networks," in *Proc. Amer. Control Conf.*, 2011, pp. 1100–1105.
[7] L. Yin, F. Zhu, Y. Ren, F. Kong, and F. Zhang, "Decentralized swarm trajectory generation for LiDAR-based aerial tracking in cluttered environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 9285–9292.
[8] R. D. Arnold, H. Yamaguchi, and T. Tanaka, "Search and rescue with autonomous flying robots through behavior-based cooperative intelligence," *J. Int. Humanitarian Action*, vol. 3, no. 1, pp. 1–18, 2018.
[9] J. P. Queralta et al., "Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision," *IEEE Access*, vol. 8, pp. 191617–191643, 2020.
[10] X. Li et al., "Collaborative search and rescue based on swarm of H-MASSs using consensus theory," *Ocean Eng.*, vol. 278, 2023, Art. no. 114426.
[11] W. Xu and F. Zhang, "FAST-LIO: A fast, robust LiDAR-inertial odometry package by tightly-coupled iterated Kalman filter," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3317–3324, Apr. 2021.

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

[12] 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.

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

[14] J. Lin and F. Zhang, "R3live: A robust, real-time, RGB-colored, LiDAR-inertial-visual tightly-coupled state estimation and mapping package," in *Proc. Int. Conf. Robot. Autom.*, 2022, pp. 10672–10678.

[15] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," in *Proc. Robot.: Sci. Syst. Conf.*, 2014, pp. 1–9.

[16] Y. Ren et al., "Bubble planner: Planning high-speed smooth quadrotor trajectories using receding corridors," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 6332–6339.

[17] F. Kong, W. Xu, Y. Cai, and F. Zhang, "Avoiding dynamic small obstacles with onboard sensing and computation on aerial robots," *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 7869–7876, Oct. 2021.

[18] Y. Ren, S. Liang, F. Zhu, G. Lu, and F. Zhang, "Online whole-body motion planning for quadrotor using multi-resolution search," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 1594–1600.

[19] G. Lu, W. Xu, and F. Zhang, "On-manifold model predictive control for trajectory tracking on robotic systems," *IEEE Trans. Ind. Electron.*, vol. 70, no. 9, pp. 9192–9202, Sep. 2023.

[20] H. Xu, L. Wang, Y. Zhang, K. Qiu, and S. Shen, "Decentralized visual-inertial-UWB fusion for relative state estimation of aerial swarm," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 8776–8782.

[21] H. Xu et al., "OMNI-swarm: A decentralized omnidirectional visual–inertial–UWB state estimation system for aerial swarms," *IEEE Trans. Robot.*, vol. 38, no. 6, pp. 3374–3394, Dec. 2022.

[22] A. Jaimes, S. Kota, and J. Gomez, "An approach to surveillance an area using swarm of fixed wing and quad-rotor unmanned aerial vehicles UAV(s)," in *Proc. IEEE Int. Conf. Syst. Syst. Eng.*, 2008, pp. 1–6.

[23] S. Moon, Y. Choi, D. Kim, M. Seung, and H. Gong, "Outdoor swarm flight system based on RTK-GPS," *J. KIISE*, vol. 43, no. 12, pp. 1315–1324, 2016.

[24] W. Hönig, J. A. Preiss, T. S. Kumar, G. S. Sukhatme, and N. Ayanian, "Trajectory planning for quadrotor swarms," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 856–869, Aug. 2018.

[25] A. Ledergerber, M. Hamer, and R. D'Andrea, "A robot self-localization system using one-way ultra-wideband communication," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 3131–3137.

[26] H. Zhou, Z. Yao, Z. Zhang, P. Liu, and M. Lu, "An online multi-robot slam system based on LiDAR/UWB fusion," *IEEE Sens. J.*, vol. 22, no. 3, pp. 2530–2542, Feb. 2022.

[27] X. Zhou et al., "Swarm of micro flying robots in the wild," *Sci. Robot.*, vol. 7, no. 66, 2022, Art. no. eabm5954.

[28] X. Zhou, J. Zhu, H. Zhou, C. Xu, and F. Gao, "EGO-Swarm: A fully autonomous and decentralized quadrotor swarm system in cluttered environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2021, pp. 4101–4107.

[29] P. Schmuck, T. Ziegler, M. Karrer, J. Perraudin, and M. Chli, "COVINS: Visual-inertial slam for centralized collaboration," in *Proc. IEEE Int. Symp. Mixed Augmented Reality Adjunct*, 2021, pp. 171–176.

[30] A. Weinstein, A. Cho, G. Loianno, and V. Kumar, "Visual inertial odometry swarm: An autonomous swarm of vision-based quadrotors," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 1801–1807, Jul. 2018.

[31] K. Guo, X. Li, and L. Xie, "Ultra-wideband and odometry-based cooperative relative localization with application to multi-UAV formation control," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2590–2603, Jun. 2020.

[32] 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 *Proc. 2024 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2024, pp. 8119–8125.

[33] 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 Trans. Robot.*, vol. 40, pp. 172–192, 2024.

[34] N. Chen et al., "A self-rotating, single-actuated UAV with extended sensor field of view for autonomous navigation," *Sci. Robot.*, vol. 8, no. 76, 2023, Art. no. eade4538.

[35] N. Chen et al., "Swashplateless-elevon actuation for a dual-rotor tail-sitter VTOL UAV," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 6970–6976.

[36] F. Zhu et al., "Swarm-LIO: Decentralized swarm LiDAR-inertial odometry," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 3254–3260.

[37] Y. Chang et al., "Lamp 2.0: A robust multi-robot SLAM system for operation in challenging large-scale underground environments," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 9175–9182, Oct. 2022.

[38] Y. Tian, Y. Chang, F. H. Arias, C. Nieto-Granda, J. P. How, and L. Carlone, "Kimera-Multi: Robust, distributed, dense metric-semantic SLAM for multi-robot systems," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2022–2038, Aug. 2022.

[39] A. Cramariuc et al., "maplab 2.0—A modular and multi-modal mapping framework," *IEEE Robot. Autom. Lett.*, vol. 8, no. 2, pp. 520–527, Feb. 2023.

[40] P.-Y. Lajoie and G. Beltrame, "Swarm-SLAM: Sparse decentralized collaborative simultaneous localization and mapping framework for multi-robot systems," *IEEE Robot. Automat. Lett.*, vol. 9, no. 1, pp. 475–482, 2023.

[41] P. C. Lusk, X. Cai, S. Wadhwania, A. Paris, K. Fathian, and J. P. How, "A distributed pipeline for scalable, deconflicted formation flying," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 5213–5220, Oct. 2020.

[42] L. Quan et al., "Robust and efficient trajectory planning for formation flight in dense environments," *IEEE Trans. Robot.*, vol. 39, no. 6, pp. 4785–4804, Dec. 2023.

[43] P.-Y. Lajoie, B. Ramtoula, Y. Chang, L. Carlone, and G. Beltrame, "DOOR-SLAM: Distributed, online, and outlier resilient SLAM for robotic teams," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1656–1663, Apr. 2020.

[44] H. Xu, P. Liu, X. Chen, and S. Shen, "d^2 SLAM: Decentralized and distributed collaborative visual-inertial SLAM system for aerial swarm," *IEEE Trans. Robot.*, 2024.

[45] T. Nguyen, K. Mohta, C. J. Taylor, and V. Kumar, "Vision-based multi-MAV localization with anonymous relative measurements using coupled probabilistic data association filter," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 3349–3355.

[46] R. Dubé, A. Gawel, H. Sommer, J. Nieto, R. Siegwart, and C. Cadena, "An online multi-robot SLAM system for 3D LiDARs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 1004–1011.

[47] C. E. Denniston et al., "Loop closure prioritization for efficient and scalable multi-robot SLAM," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 9651–9658, 2022.

[48] Y. Huang, T. Shan, F. Chen, and B. Englot, "DISCo-SLAM: Distributed scan context-enabled multi-robot LiDAR SLAM with two-stage global-local graph optimization," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 1150–1157, Apr. 2022.

[49] A. Wasik, R. Ventura, J. N. Pereira, P. U. Lima, and A. Martinoli, "LiDAR-based relative position estimation and tracking for multi-robot systems," in *Proc. 2nd Iberian Robot. Conf.*, 2016, pp. 3–16.

[50] V. Pritzl, M. Vrba, P. Štěpán, and M. Saska, "Fusion of visual-inertial odometry with LiDAR relative localization for cooperative guidance of a micro-scale aerial vehicle," 2023, *arXiv:2306.17544*.

[51] M. Saska et al., "System for deployment of groups of unmanned micro aerial vehicles in GPS-denied environments using onboard visual relative localization," *Auton. Robots*, vol. 41, pp. 919–944, 2017.

[52] M. Vrba et al., "On onboard LiDAR-based flying object detection," *IEEE Trans. Robot.*, 2024.

[53] D. He, W. Xu, and F. Zhang, "Symbolic representation and toolkit development of iterated error-state extended Kalman filters on manifolds," *IEEE Trans. Ind. Electron.*, vol. 70, no. 12, pp. 12533–12544, 2023.

[54] J. Wu and I. Stojmenovic, "Ad hoc networks," *Computer*, vol. 37, no. 2, pp. 29–31, 2004.

[55] S. T. Watt, S. Achanta, H. Abubakari, E. Sagen, Z. Korkmaz, and H. Ahmed, "Understanding and applying precision time protocol," in *Proc Saudi Arabia Smart Grid Conf.*, 2015, pp. 1–7.

[56] Y. Cao, Y. Wang, Y. Xue, H. Zhang, and Y. Lao, "FEC: Fast euclidean clustering for point cloud segmentation," *Drones*, vol. 6, no. 11, 2022, Art. no. 325.

[57] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-9, no. 5, pp. 698–700, Sep. 1987.

[58] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "ISAM2: Incremental smoothing and mapping using the Bayes tree," *Int. J. Robot. Res.*, vol. 31, no. 2, pp. 216–235, 2012.

[59] 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.

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

[61] W. Zhen and S. Scherer, "Estimating the localizability in tunnel-like environments using LiDAR and UWB," in *Proc. Int. Conf. Robot. Autom.*, 2019, pp. 4903–4908.

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

[62] F. Kong et al., "MARSIM: A light-weight point-realistic simulator for LiDAR-based UAVs," *IEEE Robot. Autom. Lett.*, vol. 8, no. 5, pp. 2954–2961, May 2023.
[63] F. Zhu et al., "Supplementary materials for Swarm-LIO2: Decentralized, efficient LiDAR-inertial odometry for UAV swarms," 2024. [Online]. Available: https://github.com/hku-mars/Swarm-LIO2/blob/main/documents/SupplementaryMaterials.pdf
[64] F. Zhu, Y. Ren, and F. Zhang, "Robust real-time LiDAR-inertial initialization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 3948–3955.
[65] K. Li, M. Li, and U. D. Hanebeck, "Towards high-performance solid-state-LiDAR-inertial odometry and mapping," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5167–5174, Jul. 2021.

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

His research interests include robotics and aerial swarm systems, with a focus on light-detection-and-ranging-based simultaneous localization and mapping and sensor calibration.

Mr. Zhu is a Reviewer for leading conferences and journals, including IEEE ROBOTICS AND AUTOMATION LETTERS, IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, IEEE/RSJ International Conference on Intelligent Robots and Systems, and IEEE International Conference on Robotics and Automation.

Yunfan Ren received the B.Eng. degree in automation from the Harbin Institute of Technology, Harbin, China, in 2021. He is currently working toward the Ph.D. degree with the Department of Mechanical Engineering, University of Hong Kong, Hong Kong.

He is currently a Member of the Mechatronics and Robotic Systems Laboratory, University of Hong Kong. His research interests include autonomous navigation, unmanned aerial vehicle planning, optimal control, and swarm system autonomy.

Mr. Ren was an Outstanding Navigation Paper Finalist at 2023 IEEE International Conference on Robotics and Automation (ICRA) and the Best Overall and Best Student Paper Finalist at 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). He is also a Reviewer for leading conferences and journals, including IEEE ROBOTICS AND AUTOMATION LETTERS, IROS, and ICRA.

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 Johns Hopkins University, Baltimore, MD, USA, in 2021. He is currently working toward the Ph.D. degree in mechanical engineering with the University of Hong Kong, Hong Kong.

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

Fanze Kong received the B.S. degree in flight vehicle design and engineering from the Harbin Institute of Technology, Harbin, China, in 2020. He is currently working toward the Ph.D. degree in robotics with the Department of the University of Hong Kong, Hong Kong.

His research interests include unmanned aerial vehicle (UAV) design, UAV motion planning, and multi-UAV autonomous navigation.

Qingbo Liu received the B.Eng. degree in ocean technology from the Dalian University of Technology, Dalian, China, in 2021, and the M.Sc. degree in mechanical engineering from the University of Hong Kong, Hong Kong, in 2022.

His current research interests include robotics, with a particular focus on light-detection-and-ranging-based techniques for robotic applications.

Ruize Xue received the B.E. degree from Sichuan University, Chengdu, China, in 2022, and the M.Sc. degree from the University of Hong Kong, Hong Kong, in 2024, both in mechanical engineering.

His research interests include control algorithm development, with a strong focus on advanced techniques for motor control and motion systems.

Wenyi Liu received the B.Eng. degree in communication engineering from the Harbin Institute of Technology (Shenzhen), Shenzhen, China, in 2022. He is currently working toward the Ph.D. degree with the Department of Mechanical Engineering, University of Hong Kong, Hong Kong.

His research interests include robotics, motion planning, and control.

Yixi Cai received the B.Eng. degree in automation from Beihang University, Beijing, China, in 2020, and the Ph.D. degree in robotics from the University of Hong Kong, Hong Kong, in 2024.

His research interests include robotics research, with a particular emphasis on efficient mapping techniques using light detection and ranging for purposes, such as localization, navigation, and exploration.

Guozheng Lu received the B.Eng. degree in automation from the Harbin Institute of Technology, Harbin, China, in 2016, and the Ph.D. degree in mechanical engineering from the University of Hong Kong, Hong Kong, in 2024.

He is currently a Senior Engineer in flight systems with DJI, Shenzhen, China. His research interests include robotics, with a focus on optimal control and motion planning. His Ph.D. research focused on trajectory generation and control of vertical-takeoff-and-landing unmanned aerial vehicles.

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.

Haotian Li received the B.E. degree in measuring and control technology and instrumentations from Harbin Engineering University, Harbin, China, in 2021. He is currently working toward the Ph.D. degree with the Department of Mechanical Engineering, University of Hong Kong, Hong Kong.

His current research interests include robotics, with a focus on unmanned aerial vehicle design and sensor fusion.

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

In 2018, he joined the Department of Mechanical Engineering, University of Hong Kong, Hong Kong, as an Assistant Professor and became an Associate Professor in 2024. His current research interests include robotics and controls, with a focus on unmanned aerial vehicle design, navigation, control, and light-detection-and-ranging-based simultaneous localization and mapping.

Authorized licensed use limited to: The University of Hong Kong Libraries. Downloaded on February 01,2025 at 07:08:54 UTC from IEEE Xplore. Restrictions apply.