2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

D-GVIO: A Buffer-Driven and Efficient Decentralized GNSS-Visual-Inertial State Estimator for Multi-Agent Systems

Yarong Luo[1], Wentao Lu[2], Chi Guo[1] and Ming Li[1]

***Abstract*— Cooperative localization is essential for swarm applications like collaborative exploration and search-and-rescue missions. However, maintaining real-time capability, robustness, and computational efficiency on resource-constrained platforms presents significant challenges. To address these challenges, we propose D-GVIO, a buffer-driven and fully decentralized GNSS-Visual-Inertial Odometry (GVIO) framework that leverages a novel buffering strategy to support efficient and robust distributed state estimation. The proposed framework is characterized by four core mechanisms. Firstly, through covariance segmentation, covariance intersection and buffering strategy, we modularize propagation and update steps in distributed state estimation, significantly reducing computational and communication burdens. Secondly, the left-invariant extended Kalman filter (L-IEKF) is adopted for information fusion, which exhibits superior state estimation performance over the traditional extended Kalman filter (EKF) since its state transition matrix is independent of the system state. Thirdly, a buffer-based re-propagation strategy is employed to handle delayed measurements efficiently and accurately by leveraging the L-IEKF, eliminating the need for costly re-computation. Finally, an adaptive buffer-driven outlier detection method is proposed to dynamically cull GNSS outliers, enhancing robustness in GNSS-challenged environments.**

I. INTRODUCTION

In recent years, multi-agent systems have demonstrated great potential in a variety of applications, including collaborative autonomous exploration [1], [2] and search-and-rescue missions [3], [4]. Robust collaborative localization enables agent teams to operate autonomously in unstructured and dynamic scenarios. A fundamental requirement for real-world multi-agent deployments is distributed state estimation, in which each agent estimates its own state from local observations.

Currently, in outdoor scenes, Global Navigation Satellite System (GNSS), particularly Real-Time Kinematic (RTK), is widely used for estimating the agent's position, as GNSS provides global position without accumulating drift over time [5]. In contrast, in GNSS-denied environments such as indoor scenes, visual-inertial odometry (VIO) [6] or LiDAR-inertial odometry (LIO) [7] are much preferred. Previous studies [5], [8] have shown that combining GNSS positioning with VIO enables high-precision localization while maintaining low-drift performance at the global scale. Therefore, the integration of GNSS and VIO offers an effective solution for achieving accurate and robust distributed state estimation in multi-agent systems (see Fig. 1).

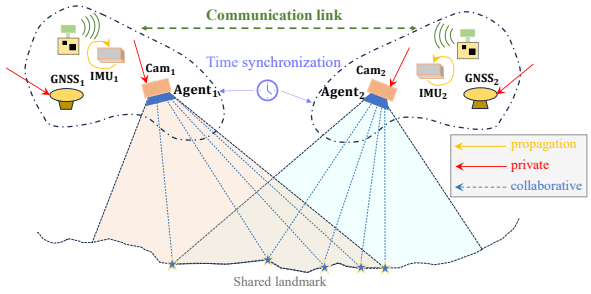

Fig. 1. Decentralized GVIO system with two agents (Agent$_1$, Agent$_2$). Each agent runs a local filter for propagation and private updates using IMU, GNSS, and camera data. Collaborative updates occur when both agents observe common landmarks via feature matching.

However, applying distributed state estimation to multi-agent systems remains challenging. As the number n of sensors per agent increases, the computational burden of naive estimators grows cubically [9], imposing heavy computational loads and degrading real-time performance, especially on resource-constrained platforms. Additionally, maintaining cross-covariance with other agents leads to communication and memory burdens that grow linearly with the number of encountered agents [10]. Thus, methods like distributed approximated cross-covariance [11] are not well-suited for large-scale swarms and long-duration missions involving frequent encounters. Moreover, information fusion from different sensors remains challenging due to varying sampling rates and computational delays. Communication and computational delays between agents are also inevitable [12]. These issues can lead to delayed measurements, which in turn may trigger extensive re-computations to maintain filter consistency.

To this end, we propose a decentralized GVIO system. This system features low computational and memory burdens, effective handling of delayed measurements, and robustness to unreliable GNSS observations. These characteristics make D-GVIO well-suited for resource-constrained platforms. More precisely, our contributions are:

- A buffer-driven decentralized collaborative GVIO system enabling efficient and robust distributed state estimation. This system features low computational and memory costs, and integrates an efficient delayed measurement handling strategy using invariant extended

[1]Yarong Luo, Chi Guo and Ming Li are with the School of Robotics, Wuhan University, Wuhan 430072, China. {yarongluo, guochi, liwhuer}@whu.edu.cn

[2]Wentao Lu is with the School of Electronic Information of Wuhan University, Hubei Luojia Laboratory, Wuhan 430072, China. wentaolu@whu.edu.cn

This work is supported in part by the National Natural Science Foundation of China under Grant 42404025 (Corresponding author: Chi Guo).

Our code is publicly available: https://github.com/braveryyyy/D-GVIO.

Kalman filter (IEKF).
- An efficient GNSS outlier detection and culling strategy. Unlike traditional chi-square test approaches that evaluate Mahalanobis distance, this system employs a buffer-based method that detects outliers by comparing GNSS velocities against an entropy-adaptive threshold derived from VIO kinematics and the Boltzmann criterion.
- Comprehensive experiments on open-source and real-world datasets. Experimental results demonstrate the efficiency and reliability of D-GVIO. Furthermore, in order to support the community's development, we have open-sourced our code and datasets on GitHub.

II. RELATED WORK

In recent years, cooperative localization for multi-agent systems has received significant attention. Existing approaches in cooperative localization can be divided into centralized and decentralized [13]. The former relies on a central unit acting as a server, where all agents upload their sensor data for fusion. In contrast, the latter performs data fusion locally on each agent, which requires efficient fusion mechanisms and minimal data exchange [14], [15]. Although a centralized system is easier to implement, it relies on a central entity that must always be reachable and fault-free. Therefore, this work focuses on the decentralized approach, which is more robust and scalable.

In decentralized framework, using sensors such as cameras, inertial measurement unit (IMU), or LiDAR to perform collaborative simultaneous localization and mapping (SLAM) for multi-agent systems is an intuitive approach. For cooperative SLAM in large-scale swarms, Schmuck et al. proposed COVINS [16], which leverages the dynamic redundancy pruning strategy, and efficient pose graph management to achieve sub-decimeter localization accuracy in multi-agent systems. However, this system relies on pose graph optimization, incurring high computational costs and poor real-time performance. Consequently, filter-based approaches become preferable for resource-limited platforms. Zhu et al. proposed Swarm-LIO [17], a decentralized filter-based swarm LIO system achieving centimeter-level accuracy, and its successor Swarm-LIO2 [14], which improves scalability by using dynamic state compression and time compensation to limit state dimension growth with swarm size. The aforementioned studies for cooperative localization rely on sensors such as IMU, LiDAR, and camera. Nevertheless, these sensors cannot provide global positioning, leading to drift accumulation [17]. Thus, GNSS integration, particularly with high-precision RTK positioning, becomes crucial in GNSS-available environments. Meanwhile, in challenged scenarios such as tunnels and dense urban areas, efficient outlier detection and robust measurement weighting strategies are required to handle degraded and unreliable observations.

Distributed state estimation on resource-limited platforms is vital for cooperative localization. Polizzi et al. [13] proposed a decentralized VIO using a vector of locally aggregated descriptors (VLAD) [18] based request-response strategy for low bandwidth. However, the high-dimensional covariance matrix required during propagation step results in a great computational burden. Addressing this, Bromme et al. [9] proposed a filtering-based multi-sensor fusion method using covariance segmentation. Jung et al. [19] then extended this to multi-agent systems, enabling efficient distributed state estimation. This system minimizes communication and computation by only exchanging data during joint observations. However, it scales poorly to large swarms as maintaining inter-agent dependencies incurs memory and communication burdens that grow linearly with the number of encountered agents.

Moreover, handling delayed measurements from communication and computation delays remains challenging. Existing re-computation approaches [9], [13] incur high computational burdens when high-frequency sensors are used. For multi-agent systems, Tasooji et al. [20] proposed a decentralized event-triggered method to mitigate accuracy loss induced by delayed measurements, but this method requires state augmentation, which increases the computational burdens. To efficiently handle delayed measurements, Alexander [21] proposed a re-propagation method that propagates the covariance and mean of error-state from the measurement occurrence moment to the update completion moment. However, the traditional EKF-based re-propagation method cannot avoid accuracy degradation, since its state transition matrix depends on the estimated state and thus becomes inaccurate when the estimated state deviates from the true state. Thus, it is meaningful to investigate how to process delayed measurements rapidly while maintaining state estimation accuracy.

III. METHODOLOGY

A. Overview

This work considers a system of n agents, each equipped with an IMU, camera, and GNSS, maintaining a local filter and capable of inter-agent communication (e.g., via WiFi). A tightly-coupled approach is employed for camera-IMU fusion due to its superior accuracy and robustness [22]. Conversely, a loosely-coupled approach is used for GNSS-IMU fusion to facilitate decoupling during GNSS outages. As illustrated in Fig. 2, the proposed D-GVIO is organized into three major modules: (i) The pre-processing module handles GNSS position input, track management, and IMU error compensation with SINS mechanization; (ii) The measurement update module performs GNSS and visual information fusion efficiently and robustly through private updates, supported by outlier detection, cross-covariance propagation, and re-propagation for delayed measurements; (iii) The collaborative update module establishes feature correspondences across agents and performs either collaborative multi-state constraint Kalman filter (MSCKF) or SLAM updates, with consistency ensured via covariance intersection (CI) algorithm [23]. Due to the space constraints, the specific procedures for collaborative MSCKF and SLAM updates can be found in [13].

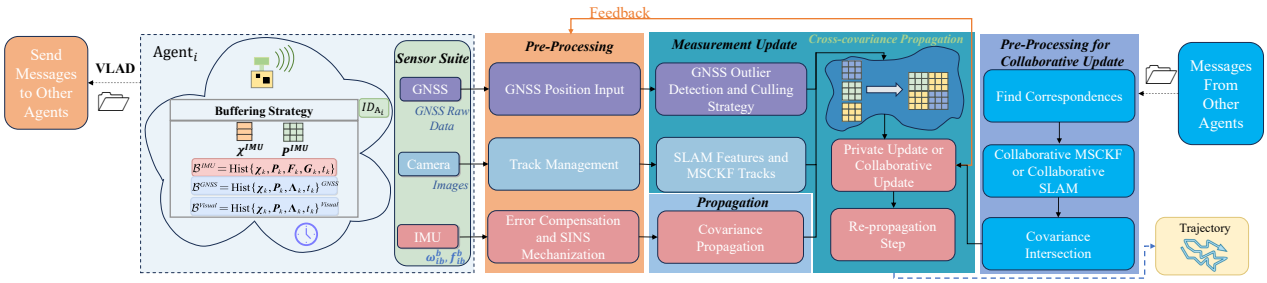

Fig. 2. Block diagram of the proposed system. Each agent maintains local SLAM features and MSCKF tracks for measurement updates. When receiving messages from other agents, the system finds feature correspondences and performs collaborative MSCKF or SLAM updates.

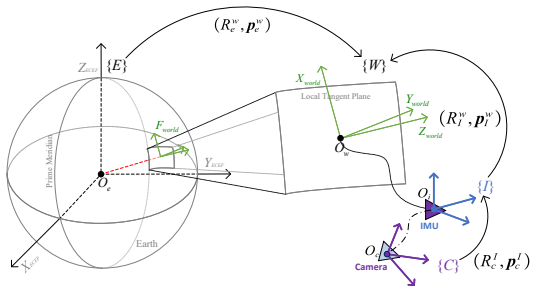

Fig. 3. Coordinate frames in the GVIO system and their transformations between each other. The transformation of frame $\{C\}$ into the frame $\{I\}$ is $(\boldsymbol{R}_c^I, \boldsymbol{p}_c^I)$ and the transformation of frame $\{I\}$ into the frame $\{W\}$ is $(\boldsymbol{R}_I^w, \boldsymbol{p}_I^w)$.

B. Reference Frames and Notations

The coordinate frames used in this work are shown in the Fig. 3. $\{W\}$ denotes the world frame. $\{E\}$ represents the Earth-Centered Earth-Fixed (ECEF) frame, which is the reference frame of GNSS position measurements. These measurements can be transformed into the world frame $\{W\}$ using the rotation matrix $\boldsymbol{R}_e^w$ and translation vector $\boldsymbol{p}_e^w$. $\{I\}$ represents the IMU frame rigidly attached to the IMU, and $\{C\}$ denotes the camera frame rigidly attached to the camera.

In this work, the notation $(\bullet)^w$ represents a quantity expressed in the frame $\{W\}$. The velocity and position of the IMU w.r.t. the frame $\{W\}$ are expressed as $\boldsymbol{v}_I^w$ and $\boldsymbol{p}_I^w$, respectively. The notation $\boldsymbol{R}_I^w$ represents the rotation matrix which rotates a vector $\boldsymbol{x}^I$ defined in the frame $\{I\}$ to a vector $\boldsymbol{x}^w = \boldsymbol{R}_I^w \boldsymbol{x}^I$ defined in the $\{W\}$ frame. Furthermore, $[\bullet]\times$ denotes the skew-symmetric matrix of a three-dimensional vector, while $[\bullet]^{\mathrm{T}}$ is used to represent the transpose of a matrix.

C. State Definition and Information Fusion via IEKF

In D-GVIO, the system state $\boldsymbol{\chi}$ of each agent can be divided into IMU-related core states $\boldsymbol{\chi}_I$, visual-related states $\boldsymbol{\chi}_v$, and GNSS-related states $\boldsymbol{\chi}_g$

$$\boldsymbol{\chi} = (\boldsymbol{\chi}_I, \boldsymbol{\chi}_v, \boldsymbol{\chi}_g) = (\boldsymbol{R}_I^w, \boldsymbol{v}_I^w, \boldsymbol{p}_I^w, \boldsymbol{b}_g, \boldsymbol{b}_a, \boldsymbol{\chi}_v, \boldsymbol{\chi}_g) \quad (1)$$

$$\boldsymbol{\chi}_v = \left(\left(\boldsymbol{R}_{c_1}^w, \boldsymbol{p}_{c_1}^w \right) \ldots \left(\boldsymbol{R}_{c_M}^w, \boldsymbol{p}_{c_M}^w \right), \boldsymbol{f}_1 \ldots \boldsymbol{f}_N \right) \quad (2)$$

where $\boldsymbol{\chi}_g = \boldsymbol{l}_b$ is the arm lever between IMU center and the GNSS antenna phase center; $\boldsymbol{\chi}$ evolves on a product manifold $\mathcal{M} = \mathbb{SE}_2(3) \times \left(\mathbb{R}^3\right)^2 \times \left(\mathbb{SO}(3) \times \mathbb{R}^3\right)^M \times \left(\mathbb{R}^3\right)^N \times \mathbb{R}^3$. $(\boldsymbol{R}_I^w, \boldsymbol{v}_I^w, \boldsymbol{p}_I^w) \in \mathbb{SE}_2(3)$ is the extended pose and $(\boldsymbol{b}_g, \boldsymbol{b}_a) \in \left(\mathbb{R}^3\right)^2$ are the gyroscope and accelerometer biases, respectively. The visual-related states include the sliding window states and the feature states. The sliding window states include the poses $(\boldsymbol{R}_c^w, \boldsymbol{p}_c^w) \in \mathbb{SO}(3) \times \mathbb{R}^3$ of the camera frame at the past M image time instances. The feature states include the inverse depth parameterization of landmarks $\boldsymbol{f}_i$ with $i = 1, ..., N$ [13]. (3) defines the $(18 + 3N + 6M)$-dimensional logarithmic error on the manifold $\mathcal{M}$ based on IEKF.

$$\begin{aligned} \boldsymbol{\eta} &= \begin{bmatrix} \boldsymbol{R}_w^I & -\boldsymbol{R}_w^I \boldsymbol{v}_I^w & -\boldsymbol{R}_w^I \boldsymbol{p}_I^w \\ \boldsymbol{0}_{1\times 3} & 1 & 0 \\ \boldsymbol{0}_{1\times 3} & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{R}}_I^w & \hat{\boldsymbol{v}}_I^w & \hat{\boldsymbol{p}}_I^w \\ \boldsymbol{0}_{1\times 3} & 1 & 0 \\ \boldsymbol{0}_{1\times 3} & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{R}_w^I \hat{\boldsymbol{R}}_I^w & \boldsymbol{R}_w^I \hat{\boldsymbol{v}}_I^w - \boldsymbol{R}_w^I \boldsymbol{v}_I^w & \boldsymbol{R}_w^I \hat{\boldsymbol{p}}_I^w - \boldsymbol{R}_w^I \boldsymbol{p}_I^w \\ \boldsymbol{0}_{1\times 3} & 1 & 0 \\ \boldsymbol{0}_{1\times 3} & 0 & 1 \end{bmatrix} \end{aligned} \quad (3)$$

$$\delta \boldsymbol{b}_g = \hat{\boldsymbol{b}}_g - \boldsymbol{b}_g, \delta \boldsymbol{b}_a = \hat{\boldsymbol{b}}_a - \boldsymbol{b}_a, \exp(\boldsymbol{\phi}^c \times) = \boldsymbol{R}_w^{c_i} \hat{\boldsymbol{R}}_{c_i}^w$$

$$\delta \boldsymbol{p}_c^w = \hat{\boldsymbol{p}}_c^w - \boldsymbol{p}_c^w, \delta \boldsymbol{f}_i = \hat{\boldsymbol{f}}_i - \boldsymbol{f}_i, \delta \boldsymbol{l}_b = \hat{\boldsymbol{l}}_b - \boldsymbol{l}_b$$

where $\boldsymbol{\eta}$ is the left invariant error; $\delta \boldsymbol{b}_g$ and $\delta \boldsymbol{b}_a$ are the errors of gyroscope and accelerometer biases, respectively; $\exp(\cdot)$ denotes the exponential map of the group $\mathbb{SE}_2(3)$; $\boldsymbol{\phi}^c$ is the attitude error of the camera expressed in the form of a rotation vector; $\delta \boldsymbol{p}_c^w$ is the position error of the camera; $\delta \boldsymbol{f}_i$ and $\delta \boldsymbol{l}_b$ are the errors of the inverse depth parameterization of landmarks and arm lever, respectively.

The IEKF [24] is used for information fusion from different sensors and agents. The discrete propagation step at time t_k is as follows:

$$\hat{\boldsymbol{\chi}}_{k|k-1} = f\left(\hat{\boldsymbol{\chi}}_{k-1|k-1}, u_k\right) \quad (4)$$

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_k \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_k^{\mathrm{T}} + \boldsymbol{G}_k \boldsymbol{Q}_k \boldsymbol{G}_k^{\mathrm{T}} \quad (5)$$

where $\hat{\boldsymbol{\chi}}_{k|k-1}$ is *a priori* estimate with covariance $\boldsymbol{P}_{k|k-1}$; $f(\cdot)$ is the process function; u_k is the control input (e.g. inertial measurements); $\boldsymbol{F}_k$ is the discrete state transition matrix; $\boldsymbol{Q}_k$ is the covariance of the process noise; $\boldsymbol{G}_k$ is the process noise matrix. It is noteworthy that the propagation step requires propagating the core states covariance and cross-covariance between the core states and the measurement sensor states.

The observation information is fused into the estimated state through the update steps [25]:

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k{}^{\mathrm{T}} \left(\boldsymbol{N}_k + \boldsymbol{H}_k^{\mathrm{T}} \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k\right)^{-1} \quad (6)$$

$$\boldsymbol{P}_{k|k} = \left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k{}^{\mathrm{T}}\right) \boldsymbol{P}_{k|k-1} \triangleq \boldsymbol{\Lambda}_k \boldsymbol{P}_{k|k-1} \quad (7)$$

$$\boldsymbol{\xi}_k = \boldsymbol{K}_k \left(\boldsymbol{y}_k - h\left(\hat{\boldsymbol{\chi}}_{k|k-1}\right)\right), \hat{\boldsymbol{\chi}}_{k|k} = \hat{\boldsymbol{\chi}}_{k|k-1} \boxminus \boldsymbol{\xi}_k \quad (8)$$

where $\boldsymbol{K}_k$ is the gain matrix at time t_k; $\boldsymbol{H}_k$ is the measurement matrix; $\boldsymbol{N}_k$ is the measurement noise covariance matrix; $h(\cdot)$ is a function of the state; $\boldsymbol{y}_k$ is the measurement of the sensor at t_k; $\boldsymbol{\Lambda}_k = \left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k{}^{\mathrm{T}}\right)$ is the correction term [26]; $\boldsymbol{\xi}_k$ is the mean of error-state; $\hat{\boldsymbol{\chi}}_{k|k}$ is *a posteriori* estimate with covariance $\boldsymbol{P}_{k|k}$; *box minus* $\boxminus$ represent the "minus" on the state manifold [14]. During the update steps, the covariances and cross-covariances of all participating agents or sensors will be updated.

D. Efficient Distributed State Estimation via Buffer-Enhanced Covariance Management and Modular Architecture

D-GVIO employs a modular framework, where each agent maintains sliding buffers based on sensor frequency. These include: a high-rate IMU buffer $\mathcal{B}^{IMU} = \text{Hist}\{\boldsymbol{\chi}_k, \boldsymbol{P}_k, \boldsymbol{F}_k, \boldsymbol{G}_k, t_k\}$ that stores core states (states and covariance), state transition matrices, and noise transition matrices; as well as low-rate sensor buffers $\mathcal{B}^{GNSS} = \text{Hist}\{\boldsymbol{\chi}_k, \boldsymbol{P}_k, \boldsymbol{\Lambda}_k, t_k\}^{GNSS}$ and $\mathcal{B}^{Visual} = \text{Hist}\{\boldsymbol{\chi}_k, \boldsymbol{P}_k, \boldsymbol{\Lambda}_k, t_k\}^{Visual}$ for storing sensor states and correction terms. $\text{Hist}\{\cdot\}$ denotes a time-horizon sliding buffer that chronologically maintains elements within a fixed duration [26], with older elements being discarded when exceeding the horizon. Buffers are fixed-size and determined by sensor rates: $\mathcal{B}^{IMU}$ is larger due to the higher IMU frequency, while $\mathcal{B}^{GNSS}$ and $\mathcal{B}^{Visual}$ are smaller to reduce the memory burden caused by storing cross-covariances. This strategy maintains constant system maintenance complexity and decouples different sensor states to enable modular processing.

It is worth noting that D-GVIO adopts a tightly-coupled VIO framework [22], which increases the dimensionality of the covariance matrix compared to loosely-coupled approaches, leading to higher computational costs. To address this, the covariance segmentation method is adopted to decouple cross-covariances from full-covariance propagation. The full-covariance $\boldsymbol{P}_{k|k}$ is partitioned into core covariance $\boldsymbol{P}^{C}_{k|k}$ and cross-covariance $\boldsymbol{P}^{C,S}_{k|k}$. Instead of propagating the full-covariance matrix at each time step, the cross-covariance propagation is deferred and reconstructed in the next update step using stored state transition matrices $\boldsymbol{F}_k$ and correction terms $\boldsymbol{\Lambda}_k$ in the buffers (see Fig. 4). This cross-covariance propagation strategy improves efficiency by avoiding full-covariance propagation and enables modular sensor integration by keeping the core state estimates independent.

Moreover, when agent A_0 and others co-observe a landmark or match SLAM features, a collaborative update is triggered on A_0. While maintaining consistency typically requires tracking inter-agent cross-covariances, this becomes memory-intensive as the number of agents grows. To address this, D-GVIO adopts the CI algorithm, which enables consistent data fusion without explicitly maintaining inter-agent correlations [27]. Specifically, the CI algorithm produces a more conservative covariance estimate than the real estimate one could obtain considering the cross-correlation, ensuring the IEKF remains consistent and avoids overconfidence. More importantly, CI algorithm entirely eliminates the need to store and communicate inter-agent cross-covariances, significantly reducing memory and bandwidth requirements. This makes the CI particularly well-suited for resource-constrained multi-agent systems. Furthermore, to achieve low-bandwidth request-response communication, D-GVIO employs VLAD to generate compact binary scene descriptors, which can reduce inter-agent data exchange volume. Each agent sends a RequestUAV message at 10 Hz containing a binary VLAD descriptor generated from ORB features using a vocabulary of 64 centroids. The RequestUAV message weighs approximately 2.05 kB, significantly smaller than a full MessageUAV (approximately 190.7 kB).

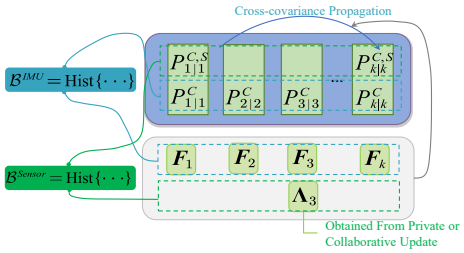

Fig. 4. Cross-covariance propagation is delayed until the next update step (at time t_k), and is propagated using the state transition matrix and correction term stored in the buffer.

E. Efficient Delayed Measurement Handling Strategy Using IEKF

In order to enhance real-time performance and better adapt to the distributed nature of multi-agent systems, D-GVIO decouples IMU processing from other sensor measurements (e.g., vision or GNSS) into parallelized independent threads. However, in multithreaded systems, since the processing time of the IMU is often shorter than the update steps of other sensors, computational or communication delays will lead to extensive re-computation steps. In general, when delayed measurements occur (see Fig. 5), the re-propagation strategy [21], which propagates both the covariance and mean from t_k to t_m using the state transition matrices stored in the buffer, can efficiently handle delayed measurements, and this method can handle delayed measurements in private update and cooperative update. The re-propagation steps are as follows:

$$\boldsymbol{P}^{C}_{m|m-1} = \left(\prod_{i=k+1}^{m} \boldsymbol{F}_i^{C}\right) \boldsymbol{P}^{C}_{k|k} \left(\prod_{i=k+1}^{m} \boldsymbol{F}_i^{C}\right)^{\mathrm{T}} + \boldsymbol{M}_{m,k+1}, \boldsymbol{\xi}_m = \left(\prod_{i=k+1}^{m} \boldsymbol{F}_i^{C}\right) \boldsymbol{\xi}_k \tag{9}$$

$$\boldsymbol{M}_{m,k+1} = \textstyle\sum_{i=k+1}^{m-1} \boldsymbol{F}_i^{C} \boldsymbol{G}_{i-1} \boldsymbol{Q}_{i-1} \left(\boldsymbol{F}_i^{C} \boldsymbol{G}_{i-1}\right)^{\mathrm{T}} + \boldsymbol{G}_m \boldsymbol{Q}_m \boldsymbol{G}_m^{\mathrm{T}} \tag{10}$$

where $\boldsymbol{M}_{m,k+1}$ is the accumulated process noise from time t_{k+1} to t_m. Therefore, using the state transition matrices

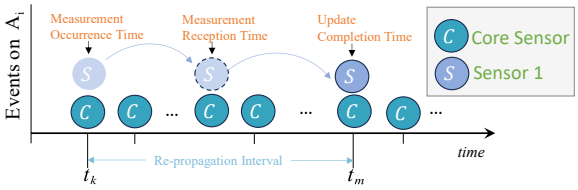

Fig. 5. Delayed measurement scenario: Sensor 1 performs update step at time t_k. Due to the communication and computational delays, the update step is not completed until time t_m.

stored in the buffer, the covariance and mean at time t_m can be efficiently computed through (9).

However, in traditional EKF, $\boldsymbol{F}_k = \frac{\partial f}{\partial \boldsymbol{\xi}}|(\hat{\boldsymbol{\chi}}_{k|k-1}, u_k)$ is linearized at the estimated state $\hat{\boldsymbol{\chi}}_{k|k-1}$. An initial estimate that deviates from the true state may yield an inaccurate $\boldsymbol{F}_k$, resulting in inconsistency issues. This issue is further exacerbated when delayed measurements occur, as re-propagation of the covariance and mean via (9) is required. The time-varying $\boldsymbol{F}_k$ during re-propagation introduces inconsistency in the covariance propagation path. Although recomputing each $\boldsymbol{F}_k$ would be accurate, it is computationally expensive. To address this, the IEKF is adopted for information fusion. In IEKF, the system dynamics are required to be group affine. This property guarantees that the left-invariant error or the right-invariant error is autonomous, resulting in their state transition matrix only depend on the control input u_k, not on the estimated state $\hat{\boldsymbol{\chi}}_{k|k-1}$ [28]. According to [29], the state transition matrices and process noise matrix of L-IEKF and right-invariant EKF (R-LIEKF) are as follows:

$$\boldsymbol{F}_k^L = e^{\boldsymbol{A}^L \Delta t}, \quad \boldsymbol{A}^L = \begin{bmatrix} -\boldsymbol{\omega}_{ib}^b\times & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ -\boldsymbol{f}_{ib}^b\times & -\boldsymbol{\omega}_{ib}^b\times & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \\ \mathbf{0}_{3\times} & \mathbf{I}_{3\times} & -\boldsymbol{\omega}_{ib}^b\times & \mathbf{0}_{3\times} & \mathbf{0}_{3\times} \\ \multicolumn{3}{c}{\mathbf{0}_{6\times9}} & \multicolumn{2}{c}{\mathbf{0}_{6\times6}} \end{bmatrix} \tag{11}$$

$$\boldsymbol{F}_k^R = e^{\boldsymbol{A}^R \Delta t}, \quad \boldsymbol{A}^R = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \hat{\boldsymbol{R}}_I^w & \mathbf{0}_{3\times3} \\ \boldsymbol{g}\times & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \hat{\boldsymbol{v}}_I^w \times \hat{\boldsymbol{R}}_I^w & \hat{\boldsymbol{R}}_I^w \\ \mathbf{0}_{3\times} & \mathbf{I}_{3\times} & \mathbf{0}_{3\times3} & \hat{\boldsymbol{p}}_I^w \times \hat{\boldsymbol{R}}_I^w & \mathbf{0}_{3\times} \\ \multicolumn{3}{c}{\mathbf{0}_{6\times9}} & \multicolumn{2}{c}{\mathbf{0}_{6\times6}} \end{bmatrix} \tag{12}$$

$$\boldsymbol{G}_k^L = \begin{bmatrix} \mathbf{I}_{6\times6} & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times6} \\ \mathbf{0}_{3\times6} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times6} \\ \multicolumn{2}{c}{\mathbf{0}_{6\times9}} & \mathbf{I}_{6\times6} \end{bmatrix}, \boldsymbol{G}_k^R = \begin{bmatrix} \hat{\boldsymbol{R}}_I^w & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times6} \\ \hat{\boldsymbol{v}}_I^w \times \hat{\boldsymbol{R}}_I^w & \hat{\boldsymbol{R}}_I^w & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times6} \\ \hat{\boldsymbol{p}}_I^w \times \hat{\boldsymbol{R}}_I^w & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times6} \\ \multicolumn{2}{c}{\mathbf{0}_{6\times9}} & \multicolumn{2}{c}{\mathbf{I}_{6\times6}} \end{bmatrix} \tag{13}$$

where $\boldsymbol{\omega}_{ib}^b$ and $\boldsymbol{f}_{ib}^b$ represent the outputs of the gyroscope and accelerometer, respectively; $\boldsymbol{g}$ is the gravity vector; $\boldsymbol{A}^L$ and $\boldsymbol{A}^R$ are the linearized error state matrices of L-IEKF and R-IEKF, respectively; $\boldsymbol{G}_k^L$ and $\boldsymbol{G}_k^R$ are the process noise matrices of L-IEKF and R-IEKF, respectively; Δt is the sampling period.

As shown in (11), (12) and (13), $\boldsymbol{F}_k^L$ and $\boldsymbol{G}_k^L$ are independent of the state estimate, whereas $\boldsymbol{F}_k^R$ and $\boldsymbol{G}_k^R$ exhibit state dependence. We attribute this to the inclusion of biases $\boldsymbol{b}_g$ and $\boldsymbol{b}_a$ in the system state, which is defined in the body frame. L-IEKF defines all error states in the body frame, naturally aligning with the bias coordinate system and avoiding state dependence introduced by coordinate transformation. Therefore, state-independent L-IEKF is used in D-GVIO for information fusion.

F. GNSS Outlier Detection and Culling Strategy

GNSS measurements in challenging scenarios (e.g., tunnels, dense urban areas) are prone to outliers, which can severely degrade fusion-based odometry accuracy. However, conventional outlier detection methods like the chi-square test [30] suffer from covariance sensitivity and high computational cost (due to covariance inversion operations). Therefore, a buffer-based position constraint outlier culling (PCOC) algorithm is proposed to overcome these challenges. PCOC estimates velocity $\boldsymbol{v}_k^{GNSS}$ from consecutive GNSS positions $\boldsymbol{p}_k^{GNSS}$ and $\boldsymbol{p}_{k-1}^{GNSS}$ and compares it to a dynamic threshold derived from the mean and standard deviation within the GNSS buffer $\mathcal{B}^{GNSS}$. If $\boldsymbol{v}_k^{GNSS}$ significantly exceeds the threshold, the GNSS position $\boldsymbol{p}_k^{GNSS}$ is regarded as an outlier and subsequently removed.

To enhance outlier detection robustness, we propose an entropy-adaptive velocity threshold strategy that jointly exploits GNSS and VIO kinematic information. In this strategy, the local VIO module provides stable velocity estimates $\boldsymbol{v}_k^{VIO}$, from which we derive the acceleration $|\boldsymbol{a}_k^{VIO}| = |\boldsymbol{v}_k^{VIO} - \boldsymbol{v}_{k-1}^{VIO}|/\Delta t$. The acceleration $|\boldsymbol{a}_k^{VIO}|$ quantifies the system's dynamic uncertainty through the lens of information entropy, where higher acceleration implies greater entropy production. This entropy-acceleration relationship follows from the Boltzmann entropy principle $S = k_B \ln \Omega$, with Ω representing the accessible state volume and k_B is the Boltzmann constant. Specifically, the threshold adjustment factor k is modeled as a function of the normalized velocity r and acceleration $|\boldsymbol{a}_k^{VIO}|$ according to the 3σ criterion:

$$V_{threshold} = \mu + k\sigma, \quad k = 3 + (1 + r) \cdot \ln\left(1 + |\boldsymbol{a}_k^{VIO}|\right) \tag{14}$$

where $V_{threshold}$ is the velocity threshold, $r = min(\boldsymbol{v}_k^{VIO}/\boldsymbol{v}_k^{emp}, 1)$ is the normalized velocity with $\boldsymbol{v}_k^{emp}$ as the empirical maximum velocity. The logarithmic term $\ln(1+|\boldsymbol{a}_k^{VIO}|)$ encodes the entropy production rate, and the factor $(1 + r)$ incorporates velocity-dependent effects. This design ensures 3σ bound for static scenarios while enabling adaptive threshold relaxation during acceleration. The specific process is depicted in Algorithm I.

Algorithm 1: Position Constraint Outlier Culling

Input: GNSS positions $\boldsymbol{p}_k^{GNSS}$, VIO positions $\boldsymbol{p}_k^{VIO}$, $\mathcal{B}^{GNSS}$
GNSS buffer size W, Parameters β and $\boldsymbol{v}^{emp}$

1. Preprocessing:
Compute GNSS velocity squared:
$\boldsymbol{v}_k^{GNSS} = \frac{1}{\Delta t}\sum_{k=1}^{3}(\boldsymbol{p}_k^{GNSS} - \boldsymbol{p}_{k-1}^{GNSS})$
Compute VIO velocity squared:
$\boldsymbol{v}_k^{VIO} = \frac{1}{\Delta t}\sum_{k=1}^{3}(\boldsymbol{p}_k^{VIO} - \boldsymbol{p}_{k-1}^{GNSS})$
Compute the mean and standard deviation:
$\mu = \text{mean}([\boldsymbol{v}_{k-W}^{\text{GNSS}}, \boldsymbol{v}_{k-W+1}^{\text{GNSS}}, ..., \boldsymbol{v}_k^{\text{GNSS}}])$
$\sigma = \text{std}([\boldsymbol{v}_{k-W}^{\text{GNSS}}, \boldsymbol{v}_{k-W+1}^{\text{GNSS}}, ..., \boldsymbol{v}_k^{\text{GNSS}}])$
2. Compute velocity threshold:
$V_{\text{threshold}} = \mu + k\sigma$
3. Outlier detection:
if $(\boldsymbol{v}_k^{GNSS} > V_{threshold})$ **then**
$\boldsymbol{p}_k^{GNSS} = \text{Null} \leftarrow$ If it is an outlier, remove it.
else
$\boldsymbol{p}_k^{GNSS} = \boldsymbol{p}_k^{GNSS} \leftarrow$ If it is not an outlier, store it in the $\mathcal{B}^{GNSS}$.
$\mathcal{B}^{GNNS}.push\ (\boldsymbol{p}_k^{GNSS})$
end if
3. return $\boldsymbol{p}_k^{GNSS}$,$\mathcal{B}^{GNSS}$

IV. Experiments

In this section, we evaluate the proposed D-GVIO using both public and self-collected datasets, including eight representative sequences. First, we compare the proposed D-GVIO with state-of-the-art (SOTA) approaches on the public simulation Castle Around dataset [13]. Next, we assess the D-GVIO using EKF, L-IEKF, and R-IEKF, along with a comparative analysis between D-GVIO's distributed state estimation approach and centralized EKF on the public S3E Square [31] as well as our real-world datasets. In the experiments, the initial yaw is derived from a dual-antenna GNSS system, which calculates the heading by measuring the phase difference of carrier signals between two physically separated antennas. In D-GVIO, the buffer sizes are

configured as follows: the IMU buffer $\mathcal{B}^{IMU}$ stores up to 300 states, while the GNSS buffer $\mathcal{B}^{GNSS}$ and visual buffer $\mathcal{B}^{Visual}$ each store up to 10 states. Given the current lack of open-source decentralized GVIO systems, we evaluate the VIO performance of D-GVIO against the state-of-the-art decentralized cooperative systems, COVINS [16] and X-VIO [13]. For GVIO performance, we compare D-GVIO with two representative single-agent systems: IC-GVINS [32], which is based on graph optimization, and InGVIO [33], which is a filter-based approach. For all three datasets, RTK positioning solutions were adopted as the reference ground truth. All experiments were conducted on a uniform hardware platform (Ubuntu 20.04 LTS, Intel Xeon Silver 4214R, 64GB RAM).

A. Simulation Experiments

The Castle Around dataset (see Fig. 6(a)) simulates four UAVs flying cooperatively around a 3D model of Inveraray Castle, following a square trajectory of 220 m. It is generated using the VI-Sensor Simulator (752×480, 30 Hz), with simulated IMU (ADIS16448, 200 Hz) and RTK positioning solution (10 Hz), featuring diverse altitudes and orientations over 30 seconds of flight.

1) Performance Comparison with SOTA Approaches: A comprehensive comparison with SOTA approaches is summarized in Tables I and II. As shown in Tables I and II, D-GVIO achieves absolute trajectory error (ATE) comparable to X-VIO, but with significantly lower central processing unit (CPU) and memory usage. Unlike the centralized EKF in X-VIO, D-GVIO propagates only core covariances and employs an efficient delayed-measurement processing strategy, substantially reducing memory and computation costs. The ATE of D-GVIO is slightly higher than that of COVINS. This outcome is expected, as the COVINS system incorporates loop closure detection and global pose and map optimization capabilities. Nevertheless, D-GVIO demonstrates reduced memory usage, lower CPU usage, and lower number of messages exchanged between agents compared to COVINS. This is because the factor graph approach requires storing the poses of historical keyframes, and landmarks. The factor graph optimization process involves iteratively solving nonlinear least-squares problems, resulting in significant memory and computational burdens. Compared with IC-GVINS, D-GVIO maintains similar sub-decimeter ATE but reduces CPU usage by 20-30% and memory consumption by over 70%, demonstrating significantly higher efficiency. In contrast, when compared with the filter-based InGVIO, D-GVIO achieves superior ATE accuracy while also exhibiting lower CPU and memory usage.

2) Consistency: Normalized Estimation Error Squared (NEES) is an important metric for evaluating the consistency of filter estimation performance, used to determine whether the filter is overly optimistic or conservative. Fig. 7 demonstrates the average normalized estimated error squared (ANEES) of the pose estimation by D-GVIO using L-IEKF, R-IEKF, and traditional EKF over 10 simulation runs. Due to space constraints, we have selected the ANEES results of UAV 1 for demonstration. Note that the ANEES of the position and orientation should be on average 3. In the experimental results, the traditional EKF yields more optimistic uncertainty estimates compared to other two filters. Meanwhile, the L-IEKF demonstrates the position and orientation ANEES closer to the reference value than the R-IEKF. These experimental results confirms the importance of keeping the state transition matrix independent of system estimates in D-GVIO.

TABLE I

ATE (METER) COMPARISON WITH THE SOTA ON THE CASTLE AROUND DATASET.

Datasets	Agent	Ours (VIO)	X-VIO (VIO)	COVINS (VIO)	Ours (GVIO)	InGVIO (GVIO)	IC-GVINS (GVIO)
Castle Around	UAV 1	0.67	0.70	0.51	0.07	0.08	**0.06**
	UAV 2	0.74	0.73	0.63	0.11	0.07	**0.06**
	UAV 3	0.82	0.82	0.76	**0.08**	0.13	0.11
	UAV 4	0.79	0.83	0.80	**0.07**	0.09	0.10

TABLE II

ALGORITHM PERFORMANCE COMPARISON WITH THE SOTA ON THE CASTLE AROUND DATASET.

Algorithm	Agent	N. of Messages sent	Max CPU (%)	Max Mem (MiB)
Ours	UAV 1	**1042**	**135.66**	**81.21**
	UAV 2	**1068**	**142.01**	**89.31**
	UAV 3	**1231**	167.75	**95.22**
	UAV 4	1256	**171.68**	**92.31**
COVINS	UAV 1	3568	200.10	657.89
	UAV 2	3576	205.21	648.48
	UAV 3	3833	199.31	701.31
	UAV 4	3842	210.21	722.65
X-VIO	UAV 1	1046	208.47	95.56
	UAV 2	1055	211.28	102.85
	UAV 3	1267	211.28	112.42
	UAV 4	**1253**	211.28	115.29
InGVIO	UAV 1	–	145.35	92.35
	UAV 2	–	157.22	94.15
	UAV 3	–	**161.21**	97.29
	UAV 4	–	175.68	93.87
IC-GVINS	UAV 1	–	180.47	305.56
	UAV 2	–	195.32	312.40
	UAV 3	–	185.48	311.14
	UAV 4	–	188.01	315.59

B. Real World Experiments

In the simulation experiments, we conduct comprehensive comparisons and analyses of D-GVIO and SOTA approaches. To further validate the performance of D-GVIO and the improvements of its submodules, we conduct experiments on two sets of real-world datasets: the open-source S3E square dataset and a self-collected dataset. The open-source S3E square dataset (Fig. 6(b)) features two wheeled robots (Robot 1 and Robot 2) following square trajectories, with synchronized 10Hz image data and RTK positioning solution. Our real-world dataset were collected in a high-rise building-dense area of Wuhan University. This dataset was collected by two remotely controlled wheeled robots (Robot 3 and Robot 4) equipped with industrial-grade cameras, GNSS receivers/antenna, and IMU sensor. Due to severe multipath effects caused by the high-rise buildings, GNSS positioning accuracy was significantly degraded in this environment. This dataset can be used to validate the robustness of the D-GVIO system and the performance of its outlier detection module. The GNSS localization results for both robots are shown in Fig. 6(c).

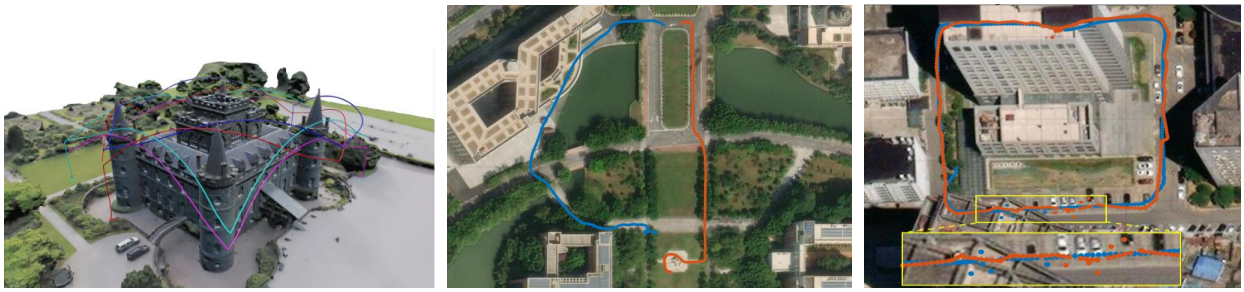

Fig. 6. Dataset scenarios (a) "Castle Around" four drones fly around the Inveraray Castle model (https://skfb.ly/6z7Rr) by Andrea Spognetta licensed under Creative Commons Attribution-NonCommercial. (b) "S3E Square" two wheeled robots driving around the square. (c) "our real-world dataset" two wheeled robots driving in parallel around a building. The subplot within the yellow box highlights GNSS outliers.

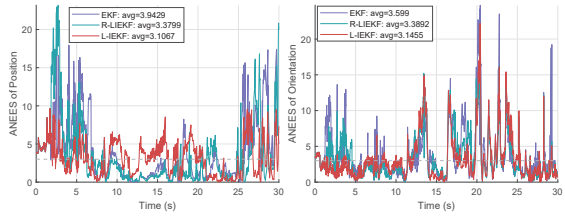

Fig. 7. The ANEES of estimated position and orientation using different filters.

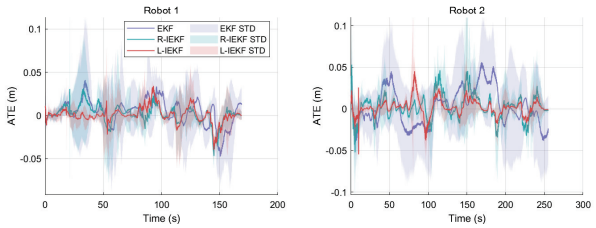

Fig. 8. The ATE of D-GVIO using different filters on the S3E Square dataset.

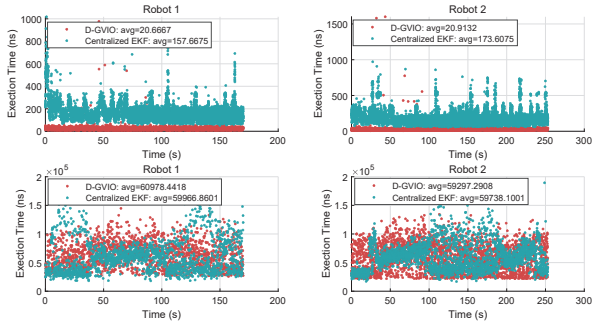

Fig. 9. The execution time of D-GVIO and centralized EKF. The legends inform about the average execution time for each robot. The two plots in the first row and second rows show the execution time of propagation and vision-related private and collaborative measurement update steps for two robots, respectively.

1) Positioning Accuracy and Computational Efficiency: Fig. 8 demonstrates the ATE of D-GVIO with EKF, L-IEKF, and R-IEKF. In the experimental results for both robots, the traditional EKF exhibits larger ATE compared to other two filters. Meanwhile, the L-IEKF demonstrates a smaller ATE and significantly more stable error curves. These experimental results are consistent with the results reflected by the ANEES for the traditional EKF, L-IEKF, and R-IEKF in our simulation experiments (see Fig. 7). Additionally, Fig. 9 shows the execution time of propagation and private/collaborative update steps in D-GVIO compared to the centralized EKF. In propagation, D-GVIO is significantly faster due to the covariance segmentation strategy, which only maintains the core covariance during propagation. In updates, D-GVIO incurs a slight time increase from cross-covariance propagation. However, given the significant reduction in propagation time and the more modular handling of sensors in D-GVIO, this modest increase in update time is acceptable.

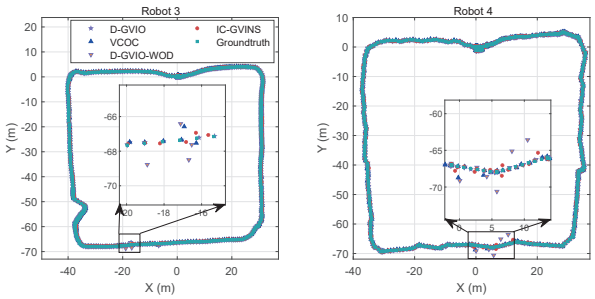

Fig. 10. Trajectories estimate results for Robot 3 and Robot 4 in the real-world dataset.

2) GNSS Outlier Detection: As shown in Fig. 10, in GNSS-challenged environments, D-GVIO demonstrates better robustness to GNSS anomalies compared to D-GVIO without the outlier detection module (D-GVIO-WOD), effectively preventing outliers from degrading positioning accuracy and achieving superior localization performance compared to IC-GVINS. Moreover, compared with the computationally efficient velocity constraint outlier culling (VCOC) algorithm [8], D-GVIO demonstrates higher sensitivity to GNSS outliers. As shown in the two subfigures of Fig. 10, D-GVIO achieves more accurate elimination of GNSS outliers, whereas VCOC is less sensitive to smaller outliers. This is because unlike VCOC, which relies on empirical values for outlier rejection, our method dynamically adjusts the threshold based on the vehicle's acceleration, thereby accounting for the dynamic nature of the driving process. These results demonstrate that our method effectively detects and removes GNSS outliers, thereby avoiding their impact on positioning accuracy.

V. Conclusion

This paper proposes D-GVIO, a decentralized GVIO system featuring a buffer-driven architecture. Unlike conventional approaches, D-GVIO decouples core states and other sensor states via a modular buffer design, significantly reducing computational and memory burdens. Furthermore, D-GVIO adopts a buffer-based re-propagation strategy with IEKF to efficiently handle delayed measurements, as well as an adaptive GNSS outlier detection mechanism to improve robustness in GNSS-challenged environments. Evaluations on three independent datasets show that D-GVIO achieves higher computational efficiency, lower memory usage, and better accuracy compared to SOTA approaches. Its efficiency and reliability make it well-suited for resource-constrained platforms in multi-agent collaborative localization. Future work will focus on dynamic collaborative networks to enhance robustness and accuracy in large-scale systems.

References

[1] Y. Gao, Y. Wang, X. Zhong, T. Yang, M. Wang, Z. Xu, Y. Wang, Y. Lin, C. Xu, and F. Gao, "Meeting-merging-mission: A multi-robot coordinate framework for large-scale communication-limited exploration," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 13 700–13 707.

[2] B. Zhou, H. Xu, and S. Shen, "Racer: Rapid collaborative exploration with a decentralized multi-uav system," *IEEE Transactions on Robotics*, vol. 39, no. 3, pp. 1816–1835, 2023.

[3] J. Berger and N. Lo, "An innovative multi-agent search-and-rescue path planning approach," *Computers & Operations Research*, vol. 53, pp. 24–31, 2015.

[4] X. Li, M. Gao, Z. Kang, H. Sun, Y. Liu, C. Yao, and A. Zhang, "Collaborative search and rescue based on swarm of h-masss using consensus theory," *Ocean Engineering*, vol. 278, p. 114426, 2023.

[5] J. Song, P. J. Sanchez-Cuevas, A. Richard, R. T. Rajan, and M. Olivares-Mendez, "Gps-vio fusion with online rotational calibration," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 11 906–11 912.

[6] J. Sun, S. Wu, J. Dong, and J. He, "Field-vio: Stereo visual-inertial odometry based on quantitative windows in agricultural open fields," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 1624–1630.

[7] A. Tao, Y. Luo, C. Xia, C. Guo, and X. Li, "Equivariant filter for tightly coupled lidar-inertial odometry," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025, pp. 2147–2153.

[8] J. Song, W. Li, C. Duan, and X. Zhu, "R^2-gvio: A robust, real-time gnss-visual-inertial state estimator in urban challenging environments," *IEEE Internet of Things Journal*, vol. 11, no. 12, pp. 22 269–22 282, 2024.

[9] C. Brommer, R. Jung, J. Steinbrener, and S. Weiss, "Mars: A modular and robust sensor-fusion framework," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 359–366, 2021.

[10] R. Jung and S. Weiss, "Scalable recursive distributed collaborative state estimation for aided inertial navigation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 1896–1902.

[11] L. Luft, T. Schubert, S. I. Roumeliotis, and W. Burgard, "Recursive decentralized localization for multi-robot systems with asynchronous pairwise communication," *The International Journal of Robotics Research*, vol. 37, no. 10, pp. 1152–1167, 2018.

[12] X. Jiang, G. Xia, Z. Feng, and X. Jing, "Distributed sampled-data nonfragile consensus filtering over sensor networks with topology switching and transmission delay," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 3, pp. 1379–1390, 2022.

[13] V. Polizzi, R. Hewitt, J. Hidalgo-Carrió, J. Delaune, and D. Scaramuzza, "Data-efficient collaborative decentralized thermal-inertial odometry," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 681–10 688, 2022.

[14] F. Zhu, Y. Ren, L. Yin, F. Kong, Q. Liu, R. Xue, W. Liu, Y. Cai, G. Lu, H. Li, and F. Zhang, "Swarm-lio2: Decentralized efficient lidar-inertial odometry for aerial swarm systems," *IEEE Transactions on Robotics*, vol. 41, pp. 960–981, 2025.

[15] R. Dubois, A. Eudes, and V. Fremont, "Sharing visual-inertial data for collaborative decentralized simultaneous localization and mapping," *Robotics and Autonomous Systems*, vol. 148, p. 103933, 2022.

[16] P. Schmuck, T. Ziegler, M. Karrer, J. Perraudin, and M. Chli, "Covins: Visual-inertial slam for centralized collaboration," in *2021 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE, 2021, pp. 171–176.

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

[18] D. Van Opdenbosch, G. Schroth, R. Huitl, S. Hilsenbeck, A. Garcea, and E. Steinbach, "Camera-based indoor positioning using scalable streaming of compressed binary image signatures," in *2014 IEEE International Conference on Image Processing (ICIP)*, 2014, pp. 2804–2808.

[19] R. Jung, C. Brommer, and S. Weiss, "Decentralized collaborative state estimation for aided inertial navigation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 4673–4679.

[20] T. K. Tasooji and H. J. Marquez, "Decentralized event-triggered cooperative localization in multirobot systems under random delays: With/without timestamps mechanism," *IEEE/ASME Transactions on Mechatronics*, vol. 28, no. 1, pp. 555–567, 2023.

[21] H. L. Alexander, "State estimation for distributed systems with sensing delay," in *Data Structures and Target Classification*, vol. 1470. SPIE, 1991, pp. 103–111.

[22] K. Sun, K. Mohta, B. Pfrommer, M. Watterson, S. Liu, Y. Mulgaonkar, C. J. Taylor, and V. Kumar, "Robust stereo visual inertial odometry for fast autonomous flight," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 965–972, 2018.

[23] S. J. Julier and J. K. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proceedings of the 1997 American Control Conference (Cat. No. 97CH36041)*, vol. 4. IEEE, 1997, pp. 2369–2373.

[24] A. Barrau and S. Bonnabel, "Invariant kalman filtering," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, 05 2018.

[25] J. Xu, P. Zhu, Y. Zhou, and W. Ren, "Distributed invariant extended kalman filter using lie groups: Algorithm and experiments," *IEEE Transactions on Control Systems Technology*, vol. 31, no. 6, pp. 2777–2789, 2023.

[26] R. Jung, L. Luft, and S. Weiss, "Isolated kalman filtering: theory and decoupled estimator design," *Autonomous Robots*, vol. 49, no. 1, p. 7, 2025.

[27] H. Li, Q. Zeng, H. Li, Y. Zhang, and J. Wu, "Distributed invariant kalman filter for object-level multi-robot pose slam," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025, pp. 15 835–15 841.

[28] A. Barrau and S. Bonnabel, "The invariant extended kalman filter as a stable observer," *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 1797–1812, 2017.

[29] A. Fornasier, Y. Ge, P. van Goor, R. Mahony, and S. Weiss, "Equivariant symmetries for inertial navigation systems," *Automatica*, vol. 181, p. 112495, 2025.

[30] B. Brumback and M. Srinath, "A chi-square test for fault-detection in kalman filters," *IEEE Transactions on Automatic Control*, vol. 32, no. 6, pp. 552–554, 1987.

[31] D. Feng, Y. Qi, S. Zhong, Z. Chen, Q. Chen, H. Chen, J. Wu, and J. Ma, "S3e: A multi-robot multimodal dataset for collaborative slam," *IEEE Robotics and Automation Letters*, vol. 9, no. 12, pp. 11 401–11 408, 2024.

[32] X. Niu, H. Tang, T. Zhang, J. Fan, and J. Liu, "Ic-gvins: A robust, real-time, ins-centric gnss-visual-inertial navigation system," *IEEE Robotics and Automation Letters*, vol. 8, no. 1, pp. 216–223, 2023.

[33] C. Liu, C. Jiang, and H. Wang, "Ingvio: A consistent invariant filter for fast and high-accuracy gnss-visual-inertial odometry," *IEEE Robotics and Automation Letters*, vol. 8, no. 3, pp. 1850–1857, 2023.