2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Multi-Robot Joint Visual-Inertial Localization and 3-D Moving Object Tracking

Pengxiang Zhu and Wei Ren

Abstract— **In this paper, we present a novel distributed algorithm to track a moving object's state by utilizing a heterogenous mobile robot network in a three-dimensional (3-D) environment, wherein the robots' poses (positions and orientations) are unknown. Each robot is equipped with a monocular camera and an inertial measurement unit (IMU), and has the ability to communicate with its neighbors. Rather than assuming a known common global frame for all the robots (which is often the case in the literature regarding multi-robot systems), we allow each robot to perform motion estimation locally. For localization, we propose a multi-robot visual-inertial navigation systems (VINS) where one robot builds a prior map and then the map is used to bound the long-term drifts of the visual-inertial odometry (VIO) running on the other robots. Moreover, a novel distributed Kalman filter is introduced and employed to cooperatively track the six degree-of-freedom (6-DoF) motion of the object which is represented as a point cloud. Further, the object can be totally invisible to some robots during the tracking period. The proposed algorithm is extensively validated in Monte-Carlo simulations.**

I. Introduction

Tracking the 6-DoF poses of moving objects in a 3-D environment is a key component in many applications such as area surveillance, region monitoring, rescue and autonomous driving. When mobile robot networks are employed to track the moving objects, a larger area can be covered and more observations to the objects can be obtained. Further, each robot in the networks are allowed to have only occasional observations of the objects, which makes the tracking system more robust in complex environments where obstacles might block some robots' views to the objects. To achieve successful tracking, the robots need to have good knowledge of their own poses. However, absolute measurements (e.g., GPS or motion-capture system) might not be available in many scenarios. In such scenarios, the cheap, lightweight sensor suite of a monocular camera and an IMU is a popular choice for motion estimation. Moreover, in multi-robot applications, it is usually assumed that a common global frame encoding all the robots' states is available. Additionally, distributed algorithms outperform centralized ones in multi-robot applications due to the strengths in scalability, processing and communication efficiency, and robustness. As such, we are interested in simultaneously estimating both the robots' poses and the object's state *locally* with only the monocular visual-inertial sensor fixed on each robot in a distributed matter.

The objective of our VINS is to achieve multi-robot localization rather than cooperative mapping [1], [2]. The single robot VINS problem has been studied extensively in recent years [3]–[7]. Among the proposed algorithms, filtering-based approaches remain the most popular for resource-constrained platforms. One of the most favorable filtering solutions is the multi-state constraint Kalman filter (MSCKF) [3] based VIO which is efficient yet accurate for real-time motion estimation. This approach only includes a constant-size sliding window of IMU poses in the state vector without storing the features. The MSCKF is extended to solve the multi-robot localization problem in [8] where the state vector includes a sliding window of every robot's IMU poses. Common environmental features observed over a sliding-window time horizon are used to add extra constraints. Recently, cooperative VINS is studied in [9], [10] where they rely on robot-to-robot camera measurements. But the same as [8], it inherits the drawback of VIO that the estimator exhibits long-term navigation drifts. In contrast, visual-inertial SLAM (VI-SLAM) [4]–[6] enables "loop closure" to provide bounded navigation errors by building a map of surroundings. However, cooperative VI-SLAM where each robot runs VI-SLAM and shares the local maps and states requires expensive communication, storage and computational cost. An extra server is used in [11] to handle computationally expensive and non-time-critical tasks. It is worth noting that the above mentioned multi-robot VINS algorithms are all centralized and running in a known common global frame.

Single robot VINS algorithms have been extended to concurrently estimate a moving object's state in recent works [12]–[15]. Ref. [12] addresses the problem of tracking a moving target using a quadrotor in cluttered environments. The quadrotor's state is estimated using a VI-SLAM algorithm and the target's trajectory is recovered using polynomial fitting with relative observations of the target's position provided by a camera. In [13], a monocular VINS is built to track the 6-DoF pose of the target. Camera poses are estimated with VINS [14], while the object's state is obtained by the combination of an object region-based bundle adjustment (BA) and metric scale estimation. A tightly-coupled estimator for visual-inertial localization and target tracking is proposed in [15] where the MSCKF is generalized to incorporate tracking of a 3D object. The target object is represented as a rigid body built from features and three motion models are proposed to capture the target's actual motion. However, the above mentioned single robot tracking requires continuous observation of the whole or partial target body. Multiple cameras are used in [16] and [17] where [16] jointly estimates the 6-DoF trajectory of a flying object and the cameras' poses while [17] propose a spatio-temporal BA

Pengxiang Zhu and Wei Ren are with the Department of Electrical and Computer Engineering, University of California, Riverside, CA 92521, USA (pzhu008@ucr.edu, ren@ee.ucr.edu).

to jointly estimate the 3-D trajectories of dynamic points and camera intrinsics and extrinsics. Both [16] and [17] are limited by the centralized approaches, the use of static cameras and the assumption of known motion dynamics of the target. Distributed Kalman filters (DKF) have been used to track targets over sensor networks [18]–[21] in 2-D scenarios. However, the proposed DKF are not suitable for the quaternion-based 3-D motion tracking, as quaternions are not valid vector quantities.

There exist several approaches for solving the problem of multi-robot joint localization and target tracking (JLATT) in a centralized way [22]–[24] or distributed way [25]. Theses algorithms share the following common limitations: (1) Only address the problem in 2-D setting, which limits their applications in many real-world scenarios which require 3-D motion. (2) The target is represented as a point particle. But vision algorithms can yield many features on the target object, which means a great amount of useful information is discarded. (3) The actual target motion model is assumed known to the robots implicitly, as they either directly simulate the target motion using the model adopted in the estimator design or use the proprioceptive sensor on the target for prediction in the experiments. As a result, the performance is not fully tested when there exists model mismatch. (4) They implicitly assume that the absolute measurements are available for setting a common global frame for all the robots.

The above observations motivate us to study the 3-D multi-robot JLATT problem with the minimal sensor suite (monocular visual-inertial sensor) mounted on each robot. As shown in Fig. 1, a robot network is employed to track the 6-DoF motion of a target object whose actual motion model is unknown. Each robot's own pose is also unknown and the robots perform motion estimation locally. Without loss of generality, we let each robot's gravity-aligned global frame have the same origin as each robot's initial IMU frame. The proposed distributed algorithm consists of multi-robot VINS and cooperative target state tracking. In particular, the main contributions of this work are summarized as follows:

- We formulate the JLATT problem in a more realistic scenario with the monocular visual-inertial setting. Specifically, we consider changing communication topologies and dynamic blind robots (the robots losing sight of the whole target object), and do not assume a known common global frame.
- We propose a multi-robot VINS system where one robot runs the VI-SLAM and builds a prior map using environmental features to improve the performance of the VIO running on the other robots.
- We propose a distributed Kalman filter to achieve 6-DoF target tracking of a moving object cooperatively. The cooperative tracking algorithm is robust to the changing blind robots and achieves good performance even if there is significant mismatch between the adopted target model and the actual one that is unknown.

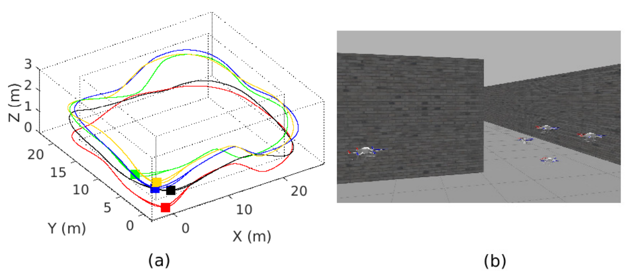

Fig. 1: Four Firefly drones equipped with visual-inertial sensors track a Pelican drone in a corridor: (a) 3D trajectories for robot 1 (red), robot 2 (green), robot 3 (blue), robot 4 (yellow) and the target (black). The corresponding squares denote the trajectory starts ; (b) Gazebo environment [26].

II. Preliminaries

A. Notations and Definitions

Let the quantity $\mathbf{x}$ represent the true value, $\hat{\mathbf{x}}$ denote the estimated value and $\tilde{\mathbf{x}}$ be the corresponding error. The superscript l/j associated with $\hat{\mathbf{x}}$ refers to the estimator of $\mathbf{x}$ at timestep l, after processing all the measurements up to timestep j. We use both the rotation matrix $\mathbf{R}$ and the unit quaternion $\bar{q}$ [27] to represent a rotation. We denote G_i, I_i and C_i, respectively, as the global frame, the IMU frame and the camera frame of robot i. T represents the target's body frame. Further, $I_{i,k}$, $C_{i,k}$ and T_k represent the corresponding frames at timestep k. ${}^{I_i}_{G_i}\mathbf{R}$ and ${}^{I_i}_{G_i}\bar{q}$ describe the same rotation from G_i to I_i. ${}^{G_i}\mathbf{v}_{I_i}$ and ${}^{G_i}\mathbf{p}_{I_i}$ are the velocity and position of I_i expressed in G_i. ${}^{G_i}\mathbf{p}_{f_i}$ and ${}^{C_i}\mathbf{p}_{f_i}$ are, respectively, feature i's position in G_i and C_i. $\{{}^{C_i}_{I_i}\mathbf{R}, {}^{C_i}\mathbf{p}_{I_i}\}$ is the set of camera-IMU extrinsic parameters. We here assume both the extrinsic and intrinsic parameters of each camera are known via prior calibration [28]. For the orientation error, we use the minimal 3-dimensional representation ${}^{G_i}\tilde{\boldsymbol{\theta}}_{I_i}$ [29] which is encoded in G_i. For all the other quantities, $\tilde{\mathbf{x}}$ is defined as the standard additive error $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$ (e.g., ${}^{G_i}\tilde{\mathbf{p}}_{I_i} = {}^{G_i}\mathbf{p}_{I_i} - {}^{G_i}\hat{\mathbf{p}}_{I_i}$). For a vector $\mathbf{x} = [x\ y\ z]^\mathsf{T}$, the perspective projection function is defined as $\Pi(\mathbf{x}) = \frac{1}{z}[x\ y]^\mathsf{T}$.

B. Communication Graph

Consider a network of M robots, we define a directed communication graph $\mathcal{G}^k = (\mathcal{V}, \mathcal{E}^k)$, where $\mathcal{V}$ is the robot set defined as $\mathcal{V} = \{R_1, \ldots, R_M\}$ and the edge set $\mathcal{E}^k$ ($\mathcal{E}^k \subseteq \mathcal{V} \times \mathcal{V}$) stands for the communication links between robots at time k. We assume that self edge $(i,i) \in \mathcal{E}^k$, $\forall i \in \mathcal{V}$, exists in the communication graph. If there exists an edge $(j,i) \in \mathcal{E}^k$, where $j \neq i$, which means that robot i can receive information from robot j, then robot j is a communicating neighbor of robot i. The communicating neighbor set of robot i at time k can be defined as $\mathcal{N}_i^k = \{i|(l,i) \in \mathcal{E}^k, l \in \mathcal{V}\}$. A directed path is a sequence of edges in a directed graph of the form $(i_0, i_1), (i_1, i_2), \ldots$, where $i_j \in \mathcal{V}$.

III. Multi-Robot VINS

In this section, we present the proposed multi-robot VINS framework where one of the robots labeled as robot 1 runs the extended Kalman filter (EKF) based VI-SLAM. Consider the fact that when a group of robots is employed to achieve a task, they usually explore the same area. Then certain features detected by robot 1 will be detected by another robot j ($j \in \mathcal{V}$, $j \neq 1$). So we can leverage the prior information about those common environmental features from robot 1 to improve the estimation performance of robot j. For robot j, the received prior map will be tightly fused into the MSCKF VIO to bound the long-term navigation drifts while maintaining the computational efficiency.

A. IMU State

For any robot i ($i \in \mathcal{V}$), the IMU state represented in G_i is described by

$$\mathbf{x}_{I_i} = \begin{bmatrix} {}^{I_i}_{G_i}\bar{q}^\top & \mathbf{b}_{\omega_i}^\top & {}^{G_i}\mathbf{v}_{I_i}^\top & \mathbf{b}_{a_i}^\top & {}^{G_i}\mathbf{p}_{I_i}^\top \end{bmatrix}^\top,$$

where $\mathbf{b}_{\omega_i}$ and $\mathbf{b}_{a_i}$ are the gyroscope and accelerometer biases. These biases are modeled as a Gaussian random walk process. The corresponding IMU error state is defined as

$$\tilde{\mathbf{x}}_{I_i} = \begin{bmatrix} {}^{G_i}\tilde{\boldsymbol{\theta}}_{I_i}^\top & \tilde{\mathbf{b}}_{\omega_i}^\top & {}^{G_i}\tilde{\mathbf{v}}_{I_i}^\top & \tilde{\mathbf{b}}_{a_i}^\top & {}^{G_i}\tilde{\mathbf{p}}_{I_i}^\top \end{bmatrix}^\top.$$

With the IMU dynamics [30], each robot can perform the EKF propagation to evolve the current IMU state and the covairance matrix according to [29].

B. Update Strategy for Robot 1

1) Localization State Vector: At the imaging timestep k, the localization state for robot 1 is given by

$$\begin{aligned} \mathbf{x}_{R_1}^k &= \begin{bmatrix} \mathbf{x}_{I_1}^k & \mathbf{x}_{C_1}^k & \mathbf{x}_S \end{bmatrix}^\top, \\ \mathbf{x}_{C_1}^k &= \begin{bmatrix} {}^{I_{1,k}}_{G_1}\bar{q}^\top & {}^{G_1}\mathbf{p}_{I_{1,k}}^\top & \cdots & {}^{I_{1,k-m}}_{G_1}\bar{q}^\top & {}^{G_1}\mathbf{p}_{I_{1,k-m}}^\top \end{bmatrix}^\top, \\ \mathbf{x}_S &= \begin{bmatrix} {}^{G_1}\mathbf{p}_{f_1}^\top & \cdots & {}^{G_1}\mathbf{p}_{f_n}^\top \end{bmatrix}^\top, \end{aligned}$$

where $\mathbf{x}_{I_1}^k$ is robot 1's IMU state at timestep k, $\mathbf{x}_{C_1}^k$ is a sliding window of m cloned historical IMU poses of robot 1, and $\mathbf{x}_S$ contains n SLAM features' positions in G_1. We refer $\mathbf{x}_{r_1} = [\mathbf{x}_{I_1}^\top \ \mathbf{x}_{C_1}^\top]^\top$ as the robot state. The error states of $\mathbf{x}_{C_1}^k$ and $\mathbf{x}_S$ take the following form

$$\begin{aligned} \tilde{\mathbf{x}}_{C_1}^k &= \begin{bmatrix} {}^{G_1}\tilde{\boldsymbol{\theta}}_{I_{1,k}}^\top & {}^{G_1}\tilde{\mathbf{p}}_{I_{1,k}}^\top & \cdots & {}^{G_1}\tilde{\boldsymbol{\theta}}_{I_{1,k-m}}^\top & {}^{G_1}\tilde{\mathbf{p}}_{I_{1,k-m}}^\top \end{bmatrix}^\top, \\ \tilde{\mathbf{x}}_S &= \begin{bmatrix} {}^{G_1}\tilde{\mathbf{p}}_{f_1}^\top & \cdots & {}^{G_1}\tilde{\mathbf{p}}_{f_n}^\top \end{bmatrix}^\top. \end{aligned}$$

2) Localization State Update: Static environmental features are captured by robot 1's onboard camera. The measurements corresponding to the same tracked feature f_i are collected over the sliding window. Each measurement is associated with the corresponding cloned pose and the feature's position. The measurement of f_i at timestep k is given by

$$\begin{aligned} \mathbf{z}_{R_1}^k &= \Pi({}^{C_{1,k}}\mathbf{p}_{f_i}) + \mathbf{n}_1^k, \\ {}^{C_{1,k}}\mathbf{p}_{f_i} &= {}^{C_1}_{I_1}\mathbf{R}^{I_{1,k}}_{G_1}\mathbf{R}\left({}^{G_1}\mathbf{p}_{f_i} - {}^{G_1}\mathbf{p}_{I_{1,k}}\right) + {}^{C_1}\mathbf{p}_{I_1}, \end{aligned} \tag{1}$$

where $\mathbf{n}_1^k$ is the zero-mean white Gaussian noise.

Next we briefly describe the adopted VI-SLAM update strategy presented in [31]. The tracked environmental features are divided into two types: (1) SLAM features that can be tracked beyond the window size m and are kept in $\mathbf{x}_S$; (2) MSCKF features that can be tracked for a short period of time or beyond m but not in $\mathbf{x}_S$. Both types of features will be used to update the localization state vector. The SLAM features in $\mathbf{x}_S$ enable "loop closure" to limit the long-term navigation drifts.

For an MSCKF feature f_i whose track has been lost or reached m, we perform the standard MSCKF update [3]. Specifically, we first perform BA to triangulate ${}^{G_1}\mathbf{p}_{f_i}$ by using the cloned poses and all the collected measurements corresponding to f_i. We then linearize each measurement to obtain the Jacobians associated with the robot state and the feature together with the measurement residual. By stacking all the values of each measurement, we get

$$\tilde{\mathbf{z}}_{R_1} = \mathbf{H}_{r_1}\tilde{\mathbf{x}}_{r_1}^{k/k-1} + \mathbf{H}_{f_i}{}^{G_1}\tilde{\mathbf{p}}_{f_i} + \mathbf{n}_1, \tag{2}$$

where $\tilde{\mathbf{z}}_{R_1}$ is the stacked measurement residual; $\mathbf{H}_{r_1}$ and $\mathbf{H}_{f_i}$ are the stacked robot state and feature Jacobians. Next, project $\tilde{\mathbf{z}}_{R_1}$ onto the left nullspace of $\mathbf{H}_{f_i}$ and we get

$$\tilde{\mathbf{z}}'_{R_1} = \mathbf{H}'_{r_1}\tilde{\mathbf{x}}_{r_1}^{k/k-1} + \mathbf{n}'_1 = \mathbf{H}'_{R_1}\tilde{\mathbf{x}}_{R_1}^{k/k-1} + \mathbf{n}'_1, \tag{3}$$

where $\tilde{\mathbf{z}}'_{R_1} = \mathbf{N}^\top\tilde{\mathbf{z}}_{R_1}$, $\mathbf{H}'_{r_1} = \mathbf{N}^\top\mathbf{H}_{r_1}$, $\mathbf{n}'_1 = \mathbf{N}^\top\mathbf{n}_1$, and $\mathbf{H}'_{R_1} = [\mathbf{H}'_{r_1} \ \mathbf{0}_{3\times 3n}]$ with $\mathbf{N}^\top\mathbf{H}_{f_i} = \mathbf{0}$. Here, (3) is independent of the feature f_i and then can be directly used to perform the standard EKF update without storing the features in the localization state.

For a SLAM feature f_1 (for notation simplicity, consider the first feature in $\mathbf{x}_S$) that can be tracked longer than m, we first triangulate its position and initialize it into $\mathbf{x}_S$ by using the first m measurements. After initialization, whenever we obtain a measurement of a SLAM feature, we trigger the update process. Linearization of the measurement at timestep k yields the following residual

$$\tilde{\mathbf{z}}_{R_1}^k = \mathbf{H}_{r_1}^k\tilde{\mathbf{x}}_{r_1}^{k/k-1} + \mathbf{H}_{f_i}^k{}^{G_1}\tilde{\mathbf{p}}_{f_1}^{k/k-1} + \mathbf{n}_1^k, \tag{4}$$

which can be further written as

$$\tilde{\mathbf{z}}_{R_1}^k = \mathbf{H}_{R_1}^k\tilde{\mathbf{x}}_{R_1}^{k/k-1} + \mathbf{n}_1^k, \tag{5}$$

where $\mathbf{H}_{R_1}^k = [\mathbf{H}_{r_1}^k \ \mathbf{H}_{f_1}^k \ \mathbf{0}_{3\times(3n-3)}]$. We can perform the standard EKF updates using (5). By observing the fact that when SLAM features become matured, there will be no significant updates in their states and covariances, we can gain computational savings by performing Schmidt EKF update for those matured features according to [31]. Specifically, we avoid updating the states and covariances of the matured features, while maintaining and updating their cross-correlations with the other states in the localization state vector. By doing this, the computational complexity becomes linear with respect to the number of SLAM features.

Robot 1 transmits a prior map including $\mathbf{x}_M$ ($\mathbf{x}_M \subseteq \mathbf{x}_S$) and the corresponding covariance set $\mathbf{P}_M$ with the corresponding descriptors to the other robots. When a new

SLAM feature loses its track and the prior map has not reached the maximum size n, we register it in the prior map and then transmit it. The descriptors are only sent once, but the prior map needs to be renewed and transmitted every transmitting time, as we include $\mathbf{x}_S$ in the localization state and the SLAM features' states are kept being refined. Moreover, if a SLAM feature become matured, no need to renew its state and covariance. So when $\mathbf{x}_S$ is matured, robot 1 can stop transmitting the prior map.

C. Update Strategy for Robot j

1) Localization State Vector: Note that robot j runs in G_j, which is different from G_1 where the prior map is encoded. To make use of the prior map, we online estimate the transformation ${}^{G_j}\mathbf{F}_{G_1} = \{{}^{G_j}_{G_1}\bar{q}, {}^{G_j}\mathbf{p}_{G_1}\}$ from G_1 to G_j. Therefore, at the imaging timestep k, the localization state for robot j is given by

$$\mathbf{x}^k_{R_j} = \begin{bmatrix}\mathbf{x}^k_{I_j} & \mathbf{x}^k_{C_j} & {}^{G_j}\mathbf{F}_{G_1}\end{bmatrix}^\top,$$
$$\mathbf{x}^k_{C_j} = \begin{bmatrix}{}^{I_{j,k}}_{G_j}\bar{q}^\top & {}^{G_j}\mathbf{p}^\top_{I_{j,k}} & \cdots & {}^{I_{j,k-m}}_{G_j}\bar{q}^\top & {}^{G_j}\mathbf{p}^\top_{I_{j,k-m}}\end{bmatrix}^\top,$$

where $\mathbf{x}^k_{I_j}$ and $\mathbf{x}^k_{C_j}$ are the current IMU state and a sliding window containing m cloned historical IMU poses of robot j. We also refer $\mathbf{x}_{r_j} = [\mathbf{x}^\top_{I_j}\ \mathbf{x}^\top_{C_j}]^\top$ as the robot state. The error state of $\mathbf{x}^k_{C_j}$ is defined the same as $\mathbf{x}^k_{C_1}$ and the error state of ${}^{G_j}\mathbf{F}_{G_1}$ is defined as ${}^{G_j}\tilde{\mathbf{F}}_{G_1} = [{}^{G_1}\tilde{\boldsymbol{\theta}}^\top_{G_j}\ \ {}^{G_j}\tilde{\mathbf{p}}^\top_{G_1}]^\top$. Note that to estimate ${}^{G_j}\mathbf{F}_{G_1}$, we need an initial guess which can be obtained with Horn's method [32] by using the first few (more than three) detected map features.

2) Localization State Update: We divide the static environmental features tracked by robot j's camera into two types: (1) map features that are inside $\mathbf{x}_M$ received from robot 1; (2) MSCKF features that can be tracked for a short period of time or beyond m but not in $\mathbf{x}_M$. Both types of features will be used to update the localization state. Similar to (1), the measurements corresponding to the same tracked MSCKF feature are collected over the sliding window. At timestep k, the observation model for an MSCKF feature f_j is given by

$$\mathbf{z}^k_{R_j} = \Pi({}^{C_{j,k}}\mathbf{p}_{f_j}) + \mathbf{n}^k_j, \tag{6a}$$
$${}^{C_{j,k}}\mathbf{p}_{f_j} = {}^{C_j}_{I_j}\mathbf{R}\,{}^{I_{j,k}}_{G_j}\mathbf{R}\left({}^{G_j}\mathbf{p}_{f_j} - {}^{G_j}\mathbf{p}_{I_{j,k}}\right) + {}^{C_j}\mathbf{p}_{I_j}, \tag{6b}$$

where $\mathbf{n}^k_j$ is the zero-mean white Gaussian noise with covariance $\mathbf{Q}^k_j$. The standard MSCKF update can be performed for the MSCKF feature as described in Section III-B by using the cloned poses and the collected measurements.

Unlike the MSCKF feature, whenever we obtain a measurement of a map feature, we trigger the update process. For the observation model of a map feature f_j, we replace (6b) with

$$\begin{aligned}{}^{C_{j,k}}\mathbf{p}_{f_j} &= {}^{C_j}_{I_j}\mathbf{R}\,{}^{I_{j,k}}_{G_j}\mathbf{R} \\ &\left({}^{G_j}_{G_1}\mathbf{R}\,{}^{G_1}\mathbf{p}_{f_j} + {}^{G_j}\mathbf{p}_{G_1} - {}^{G_j}\mathbf{p}_{I_{j,k}}\right) + {}^{C_j}\mathbf{p}_{I_j}.\end{aligned} \tag{7}$$

Note that (7) provides not only the constraints of the IMU pose, but also the constraints of the transformation between two global frames. Linearizaton of (6a), (7) yields the following residual

$$\tilde{\mathbf{z}}^k_{R_j} = \mathbf{H}^k_{R_j}\tilde{\mathbf{x}}^{k/k-1}_{R_j} + \mathbf{H}^k_{f_j}\,{}^{G_1}\tilde{\mathbf{p}}_{f_j} + \mathbf{n}^k_j, \tag{8}$$

where $\mathbf{H}^k_{R_j}$ and $\mathbf{H}^k_{f_j}$ are the corresponding Jacobians. Unlike (2) and (5), here ${}^{G_1}\mathbf{p}_{f_j}$ is known from the prior map $\mathbf{x}_M$ with the covariance $\mathbf{P}_{f_j} \in \mathbf{P}_M$. Define $\tilde{\mathbf{n}}^k_j = \mathbf{H}^k_{f_j}\,{}^{G_1}\tilde{\mathbf{p}}_{f_j} + \mathbf{n}^k_j$ with the covariance $\tilde{\mathbf{Q}}^k_j = \mathbf{H}^k_{f_j}\mathbf{P}_{f_j}(\mathbf{H}^k_{f_j})^\top + \mathbf{Q}^k_j$. Then (8) turns into

$$\tilde{\mathbf{z}}^k_{R_j} = \mathbf{H}^k_{R_j}\tilde{\mathbf{x}}^{k/k-1}_{R_j} + \tilde{\mathbf{n}}^k_j. \tag{9}$$

Equation (9) can be used to update the localization state directly with the standard EKF update. Note that we have taken into account the prior map's uncertainty in (9) which further improves the accuracy.

The size of robot j's state vector is $(16 + 6m + 6)$ which is comparable to that of a standard MSCKF $(16 + 6m)$ [3], but much smaller than that of the VI-SLAM $(16+6m+3n)$, especially for a large-scale environment ($n \gg m$). So robot j maintains the computationally efficiency of MSCKF while avoiding long-term drift with the aid of the prior map built by robot 1.

IV. Cooperative Target State Tracking

In this section, we present the proposed cooperative target state tracking approach that is based on a novel distributed Kalman filter. In our setting, each robot maintains an estimator of the common target's state in addition to its own pose estimator.

A. Tracking State Vector

As we do not assume a known common global frame, the target's state would express different values in different global frames. However, a prerequisite for using a neighboring robot's information is that this information is encoded in the same frame. So each robot tracks the target in its own global frame independently before initializing the transformations between the global frames. After initialization, we can convert the estimated target state of robot j ($j \in \mathcal{V}$, $j \neq 1$) from G_j to G_1 with the estimated value of ${}^{G_j}\mathbf{F}_{G_1}$. After initialization of the transformation, the target state is encoded in G_1. We define the target state as [15]

$$\mathbf{x}_T = \begin{bmatrix}{}^{T}_{G_1}\bar{q}^\top & {}^{G_1}\mathbf{p}_T{}^\top & {}^T\boldsymbol{\omega}^\top & {}^{G_1}\mathbf{v}_T{}^\top\end{bmatrix}^\top,$$

where ${}^T_{G_1}\bar{q}$ describes the rotation from G_1 to T, ${}^{G_1}\mathbf{p}_T$ is the position of T in G_1, ${}^{G_1}\mathbf{v}_T$ is target's global linear velocity and ${}^T\boldsymbol{\omega}$ is the target's local angular velocity. Both ${}^{G_1}\mathbf{v}_T$ and ${}^T\boldsymbol{\omega}$ are treated as continuous-time random walks driven by noises $\mathbf{n}_v$ and $\mathbf{n}_\omega$, respectively.

Like [15], we represent the 3D rigid-body target as a point cloud consisting of corner features that can be tracked by the robots' cameras. One of these target features is chosen as the representative point where the pose of the target is defined while the other features are the non-representative features that provide additional observations. Note that none of the target features is required to be reliably tracked by each robot over time. It could be the case that the target is totally invisible to some of the robots in the group. A sparse feature set of the target is extracted and tracked. As we employ multiple robots, more observations and constraints can be obtained for every target feature. This makes it possible to limit the number of tracked features for a successful tracking. So unlike [15], instead of maintaining a sliding window of cloned historical target poses to triangulate none-representative features' positions, we add these features' relative positions in the target's body frame to our tracking state to provide reobservation constraints. Therefore, at timestep k, the tracking state for each robot is given by

$$\mathbf{x}_O^k = \begin{bmatrix}\mathbf{x}_T^k & {}^T\mathbf{p}_t\end{bmatrix}^\top, \quad {}^T\mathbf{p}_t = \begin{bmatrix}{}^T\mathbf{p}_{t_1}^\top & \cdots & {}^T\mathbf{p}_{t_s}^\top\end{bmatrix}^\top,$$

where $\mathbf{x}_T^k$ is the target state at timestep k, and ${}^T\mathbf{p}_t$ contains s non-representative features' positions in T. Note that ${}^T\mathbf{p}_t$ does not evolve over time as we assume a rigid-body target. Robot i $(i \in \mathcal{V})$ maintains an estimator $\hat{\mathbf{x}}_{O_i}$ of $\mathbf{x}_O$ and the corresponding error state is given by

$$\tilde{\mathbf{x}}_{O_i} = \begin{bmatrix}\tilde{\mathbf{x}}_{T_i} & {}^{T_i}\tilde{\mathbf{p}}_t\end{bmatrix}^\top, \quad {}^{T_i}\tilde{\mathbf{p}}_t = \begin{bmatrix}{}^{T_i}\tilde{\mathbf{p}}_{t_1}^\top & \cdots & {}^{T_i}\tilde{\mathbf{p}}_{t_s}^\top\end{bmatrix}^\top,$$
$$\tilde{\mathbf{x}}_{T_i} = \begin{bmatrix}{}^{G_1}\tilde{\boldsymbol{\theta}}_{T_i}^\top & {}^{G_1}\tilde{\mathbf{p}}_{T_i}^\top & {}^{T_i}\tilde{\boldsymbol{\omega}}^\top & {}^{G_1}\tilde{\mathbf{v}}_{T_i}^\top\end{bmatrix}^\top,$$

where the orientation error ${}^{G_1}\tilde{\boldsymbol{\theta}}_{T_i}$ is expressed in G_1 and the subscript i associated with T denotes the quantity obtained by robot i.

B. Target Measurements

Like the static environmental features, the target features are captured by the robots' cameras. For robot 1, the measurements of the representative features take the form

$$\mathbf{z}_{T_1}^k = \Pi({}^{C_{1,k}}\mathbf{p}_{T_k}) + \mathbf{n}_1^k, \tag{10}$$
$${}^{C_{1,k}}\mathbf{p}_{T_k} = {}^{C_1}_{I_1}\mathbf{R}\,{}^{I_{1,k}}_{G_1}\mathbf{R}\left({}^{G_1}\mathbf{p}_{T_k} - {}^{G_1}\mathbf{p}_{I_{1,k}}\right) + {}^{C_1}\mathbf{p}_{I_1}. \tag{11}$$

While for a non-representative feature ${}^T\mathbf{p}_{t_j}$, (11) is replaced with

$${}^{C_{1,k}}\mathbf{p}_{T_k} = {}^{C_1}_{I_1}\mathbf{R}\,{}^{I_{1,k}}_{G_1}\mathbf{R}\left({}^{T_k}_{G_1}\mathbf{R}^{\top T}\mathbf{p}_{t_j} + {}^{G_1}\mathbf{p}_{T_k} - {}^{G_1}\mathbf{p}_{I_{1,k}}\right) + {}^{C_1}\mathbf{p}_{I_1}. \tag{12}$$

For robot j $(j \neq 1)$, as we encode the target state in G_1, the measurements of the representative feature is given by

$$\mathbf{z}_{T_j}^k = \Pi({}^{C_{j,k}}\mathbf{p}_T) + \mathbf{n}_j^k, \tag{13}$$
$${}^{C_{j,k}}\mathbf{p}_T = {}^{C_j}_{I_j}\mathbf{R}\,{}^{I_{j,k}}_{G_j}\mathbf{R}\left({}^{G_j}_{G_1}\mathbf{R}\,{}^{G_1}\mathbf{p}_{T_k} + {}^{G_j}\mathbf{p}_{G_1} - {}^{G_j}\mathbf{p}_{I_{j,k}}\right) + {}^{C_j}\mathbf{p}_{I_j}. \tag{14}$$

While for a non-representative feature ${}^T\mathbf{p}_{t_j}$, we replace (14) with

$${}^{C_{j,k}}\mathbf{p}_T = {}^{C_j}_{I_j}\mathbf{R}\,{}^{I_{j,k}}_{G_j}\mathbf{R}\left[{}^{G_j}_{G_1}\mathbf{R}({}^{T_k}_{G_1}\mathbf{R}^{\top T}\mathbf{p}_{t_j} + {}^{G_1}\mathbf{p}_{T_k}) + {}^{G_j}\mathbf{p}_{G_1} - {}^{G_j}\mathbf{p}_{I_{j,k}}\right] + {}^{C_j}\mathbf{p}_{I_j}. \tag{15}$$

The linearization residuals of these measurements take the following compatible form for notation simplicity

$$\tilde{\mathbf{z}}_{T_i}^k = \check{\mathbf{H}}_{O_i}^k \tilde{\mathbf{x}}_{O_i}^{k/k-1} + \check{\mathbf{H}}_{R_i}^k \tilde{\mathbf{x}}_{R_i}^{k/k-1} + \mathbf{n}_i^k. \tag{16}$$

Here, for robot 1, equation (16) is computed using (10), (11) and (12). While for robot j, equation (16) is computed using (13), (14) and (15). $\check{\mathbf{H}}_{O_i}^k$ and $\check{\mathbf{H}}_{R_i}^k$ are the corresponding localization and tracking state Jacobians.

C. Distributed Kalman Filter For Tracking

Unlike the robots whose onboard IMU measurements provide the propagations for the states, we do not have access to any sensor measuring the target's actual motion. In other words, we do not assume that the actual target motion model is available to the robots. We adopt the following constant linear global velocity dynamics given by [15]

$$\begin{aligned} {}^T_{G_1}\dot{\bar{q}} &= \frac{1}{2}\boldsymbol{\Omega}({}^T\boldsymbol{\omega})^T_{G_1}\bar{q}, \quad {}^{G_1}\dot{\mathbf{p}}_T = {}^{G_1}\mathbf{v}_T, \\ {}^{G_1}\dot{\mathbf{v}}_T &= \mathbf{n}_v, \quad {}^T\dot{\boldsymbol{\omega}} = \mathbf{n}_\omega, \end{aligned} \tag{17}$$

to propagate the target's state. Here, ${}^{G_1}\mathbf{v}_T$ and ${}^T\boldsymbol{\omega}$ are treated as random walk driven by the noise $\mathbf{n}_v$ and $\mathbf{n}_\omega$. By linearization and discretization of (17), each robot i can perform the EKF propagation to evolve the target state and the tracking state covariance.

To avoid degradation of the localization part caused by the poorly modeling of the target's actual motion. We decouple the localization and tacking systems by not updating the cross-covariances between them (set the cross-covariances to zero). Further, to avoid information double-counting, we do not use the target measurements to update the localization states. For any robot i at timestep k, we define $\bar{\mathbf{n}}_i^k = \check{\mathbf{H}}_{R_i}^k \tilde{\mathbf{x}}_{R_i}^{k/k-1} + \mathbf{n}_i^k$. The corresponding covariance is given by

$$\bar{\mathbf{Q}}_{T_i}^k = \check{\mathbf{H}}_{R_i}^k \mathbf{P}_{R_i}^{k/k-1}(\check{\mathbf{H}}_{R_i}^k)^\top + \mathbf{Q}_{T_i}^k, \tag{18}$$

which takes account of the localization states' uncertainties. Then, the measurement residuals (16) turn into

$$\tilde{\mathbf{z}}_{T_i}^k = \check{\mathbf{H}}_{O_i}^k \tilde{\mathbf{x}}_{O_i}^{k/k-1} + \bar{\mathbf{n}}_i^k. \tag{19}$$

Further, we define two correction terms as

$$\begin{aligned} \mathbf{s}_i^k &= (\check{\mathbf{H}}_{O_i}^k)^\top (\bar{\mathbf{Q}}_{T_i}^k)^{-1} \check{\mathbf{H}}_{O_i}^k, \\ \mathbf{y}_i^k &= (\check{\mathbf{H}}_{O_i}^k)^\top (\bar{\mathbf{Q}}_{T_i}^k)^{-1} \tilde{\mathbf{z}}_{T_i}^k, \end{aligned} \tag{20}$$

where we consider the uncertainties of the localization states. Then, a large uncertainty $\mathbf{P}_{R_l}^{k/k-1}$ in robot l's localization state will lead to a large $\bar{\mathbf{Q}}_{T_l}^k$, which makes the corresponding correction terms small. Note that by using $\bar{\mathbf{Q}}_{T_l}^k$ in the correction terms, the resulting estimator is more accurate than the one obtained by simply setting $\check{\mathbf{H}}_{R_i}^k$ to zero.

Next, we present the distributed update procedure running on each robot i. Recall that every robot maintains an estimator of the target. After propagation, robot i receives $\{\hat{\mathbf{x}}_{O_l}^{k/k-1}, \mathbf{P}_{O_l}^{k/k-1}, \mathbf{s}_l^k, \mathbf{y}_l^k\}$ (sends $\{\hat{\mathbf{x}}_{O_i}^{k/k-1}, \mathbf{P}_{O_i}^{k/k-1}, \mathbf{s}_i^k, \mathbf{y}_i^k\}$) from (to) robot l, $\forall l \in \mathcal{N}_i^k$. We first "weighted average" the prior estimators among the communicating neighbor set $\mathcal{N}_i^k$ to reduce its uncertainty. In order to find the average orientation, we employ the following method [33] which finds a quaternion that minimizes the weighed sum of the orientation errors estimated by each robot.

$$\begin{aligned}
{}^{T_{i,k}}_{G_1}\check{\bar{q}} &= \arg\max_{\bar{q}\in\mathbb{S}^3} \bar{q}^{\mathsf{T}}\mathbf{M}\bar{q}, \\
\mathbf{M} &= \sum_{l\in\mathcal{N}_i^k} \pi_l^k ({}^{T_{l,k/k-1}}_{G_1}\hat{\bar{q}})^{\mathsf{T}}\,{}^{T_{l,k/k-1}}_{G_1}\hat{\bar{q}},
\end{aligned} \tag{21}$$

where $\mathbb{S}^3$ denotes the unit 3-sphere. For the remaining quantities in $\hat{\mathbf{x}}_{O_i}^{k/k-1}$, we compute $\check{\mathbf{x}}_{V_i}^{k} = \sum_{l\in\mathcal{N}_i^k} \pi_l^k \hat{\mathbf{x}}_{V_l}^{k/k-1}$, where $\hat{\mathbf{x}}_{V_l} = \begin{bmatrix} {}^{G_1}\hat{\mathbf{p}}_{T_l}^{\mathsf{T}} & {}^{T_l}\hat{\boldsymbol{\omega}}^{\mathsf{T}} & {}^{G_1}\hat{\mathbf{v}}_{T_l}^{\mathsf{T}} & {}^{T_l}\hat{\mathbf{p}}_{tf}^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}}$. We define a compatible symbol $\otimes$ for computing the averaged state and then we have $\sum_{l\in\mathcal{N}_i^k} \pi_l^k \otimes \hat{\mathbf{x}}_{O_l}^{k/k-1}$. As for the covariance, we can directly compute $\sum_{l\in\mathcal{N}_i^k} \pi_l^k (\mathbf{P}_{O_l}^{k/k-1})$, since all errors are represented by valid vector quantities. Then, we update the estimator according to the following novel distributed update equations

$$\begin{aligned}
\mathbf{P}_{O_i}^{k/k} &= \left[\left(\sum_{l\in\mathcal{N}_i^k} \pi_l^k \mathbf{P}_{O_l}^{k/k-1}\right)^{-1} + \sum_{l\in\mathcal{N}_i^k} \mathbf{s}_l^k\right]^{-1}, \\
\hat{\mathbf{x}}_{O_i}^{k/k} &= \sum_{l\in\mathcal{N}_i^k} \pi_l^k \otimes \hat{\mathbf{x}}_{O_l}^{k/k-1} + \mathbf{P}_{O_i}^{k/k} \sum_{l\in\mathcal{N}_i^k} \mathbf{y}_l^k,
\end{aligned} \tag{22}$$

where the weight π_l^k subject to $\pi_l^k \in [0,1]$ and $\sum_{l\in\mathcal{N}_i^k} \pi_l^k = 1$ is selected to minimize the determinant or the trace of $\mathbf{P}_{O_i}^{k/k}$. The detailed derivation of (22) can be found in [34]. In (22), the prior estimators are "weighted averaged" over the neighborhood and the target measurements from the neighboring robots are used. Therefore, a robot directly detect the target can affect the other robots through the communication topology. The target state is thus cooperatively estimated by the robots, even if certain robots cannot capture any of the target features over a time interval.

V. Results

In this section, we present the results of the Monte-Carlo simulations that demonstrate the effectiveness of the proposed algorithm. The Gazebo MAV simulator RotorS [35] is used to create the tracking scenario where four Firefly drones (tracking robots) and a Pelican drone (the target) fly following 3D trajectories in a corridor and the approximate loop period is 75 seconds. The non-representative target features are generated around the target's body frame while the static environment features are simulated on the walls. Each tracking robot is equipped with an visual-inertial sensor and has the ability to communicate with the neighboring robots. The resolution of the camera is $[752, 480]$ while its maximum sensing distance is purposely set to 5m. The groundtruth of the IMU and the image measurements obtained by each robot are corrupted by the realistic sensor characteristics as shown in Table I. The sliding window sizes m for all the robots are set to the same value 15 and robot 1's SLAM feature number n is 200. Then we perform 45 Monte-Carlo simulations and the results are quantified by the root mean squared error (RMSE).

TABLE I: Sensor parameters in simulation.

Parameter	Value
IMU rate	100 (Hz)
Gyroscope noise density	1.6968e-4 (rad/s/$\sqrt{\text{Hz}}$)
Gyroscope random walk	1.9393e-5 (rad/$s^2/\sqrt{\text{Hz}}$)
Accelerometer noise density	2.0000e-3 (m/$s^2/\sqrt{\text{Hz}}$)
Accelerometer random walk	3.0000e-3 (m/$s^3/\sqrt{\text{Hz}}$)
Camera rate	10 (Hz)
Image noise	1 (pixel)

A. Localization

To validate the performance, we compare the proposed multi-robot VINS (MR-VINS) approach to the case where robot j ($j = 2,3,4$) works independently with the standard MSCKF-VIO. The averaged RMSE results of the robots' poses over all Monte-Carlo trials are shown in Fig. 2 and Fig. 3, while Table II provides the results over all Monte-Carlo trials and all timesteps for the estimated transformations of the global frames. As expected, in both the position and orientation, running independently with MSCKF-VIO exhibits accumulated long-term drift while the MR-VINS provides much smaller and bounded errors without long term drift. In addition, robot j's ($j = 2,3,4$) pose estimator is less accurate than the one of robot 1. It is mainly caused by two reasons: (1) SLAM features included in $\mathbf{x}_S$ are not all captured by robot j; (2) The cross-correlations with $\mathbf{x}_S$ maintained by robot 1 are used to update the localization state and covariances. However, robot j gains significant computational savings and is as efficient as MSCKF-VIO. As is evident from Table II, the estimated transformation from G_j to G_1 is very accurate.

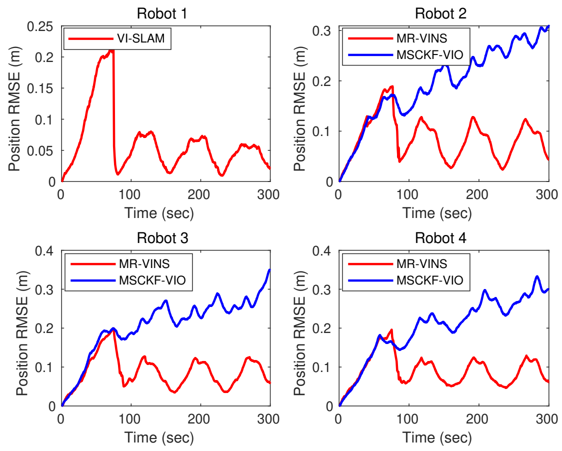

Fig. 2: Averaged RMSE for the estimated robots' positions.

B. Tracking

To show the benefits of cooperative tracking, we assume that each robot can communicate with the other robots with certain percentages. For example, 50% means that each robot communicates with the other robots with the probability of 50% at every timestep. The communication graph is time

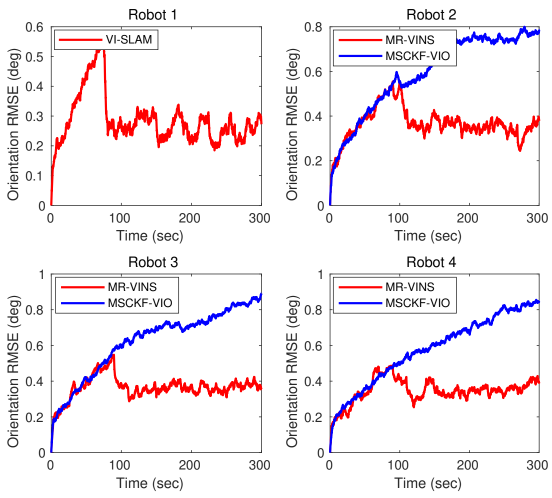

Fig. 3: Averaged RMSE for the estimated robots' orientations.

TABLE II: Averaged RMSE for the estimated global frame transformations.

Time (sec)		Initial	50	100
${}^{G_2}\mathbf{F}_{G_1}$	${}^{G_2}_{G_1}\bar{q}$ (deg)	4.191	0.150	0.149
	${}^{G_2}\mathbf{p}_{G_1}$ (cm)	24.253	2.981	2.506
${}^{G_3}\mathbf{F}_{G_1}$	${}^{G_3}_{G_1}\bar{q}$ (deg)	2.564	0.144	0.138
	${}^{G_3}\mathbf{p}_{G_1}$ (cm)	18.564	3.86.	3.692
${}^{G_4}\mathbf{F}_{G_1}$	${}^{G_4}_{G_1}\bar{q}$ (deg)	2.186	0.200	0.193
	${}^{G_4}\mathbf{p}_{G_1}$ (cm)	15.001	3.442	3.211

varying. Table III provides the averaged RMSE for the estimated target's pose over all Monte-Carlo trials and all timesteps for different communication percentages. It becomes clear that as the communication probability increases, the estimation errors reduce significantly for all the robots in both the positions and orientations. The errors of the estimated target's pose for all the robots are very large when the communication percentage is 0 (no collaboration between robots in tracking). This is because we simulate a realistic scenario where each robot can become a blind robot during different time intervals.

Further, Fig. 4 shows the averaged RMSE results for the estimated target's pose over all trials with 25% and 50% communications. It is interesting to point out that the results are not as smooth as those for the robots' poses. This is most likely due to the fact that the actual simulated target motion does not follow the model (17) or exhibits constant global velocity. In particular, when the communication percentage is 50%, several peaks appear periodically. This is caused by the larger motion modelling error around the corners where the target's actual velocities change quickly. However, as we increase the communication percentage to 50%, the RMSE values become smaller especially for the values around the corners. This demonstrates the strength of cooperative tracking. Additionally, the larger errors at the beginning are caused by the fact that the robots work independently before initializing the global frame transformations. It is clear that all the robots can well track the target's 6-DOF motion over a long time period with 50% communication.

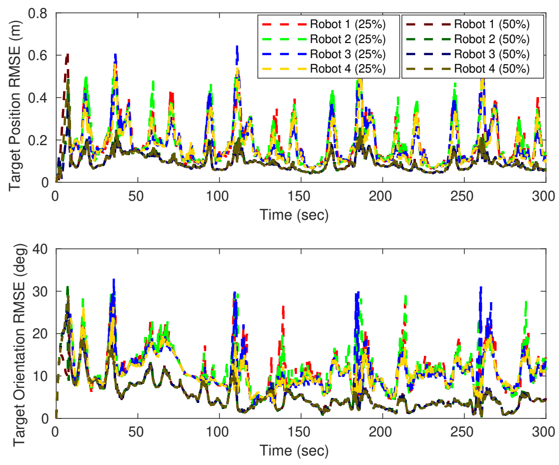

Fig. 4: Averaged RMSE for the estimated target's poses obtained by the four robots when the communication percentages are 25% and 50%.

TABLE III: Averaged RMSE for the estimated target pose obtained by the robots with different communication percentages.

communication		0 %	25 %	50%	80%
Robot1	${}^{T}_{G_1}\bar{q}$ (deg)	45.530	12.553	5.404	2.153
	${}^{G_1}\mathbf{p}_T$ (m)	3.032	0.206	0.127	0.070
Robot2	${}^{T}_{G_1}\bar{q}$ (deg)	38.598	12.799	5.542	2.167
	${}^{G_1}\mathbf{p}_T$ (m)	3.560	0.208	0.122	0.072
Robot3	${}^{T}_{G_1}\bar{q}$ (deg)	29.908	12.273	5.516	2.150
	${}^{G_1}\mathbf{p}_T$ (m)	1.831	0.201	0.124	0.074
Robot4	${}^{T}_{G_1}\bar{q}$ (deg)	32.355	11.867	5.463	2.158
	${}^{G_1}\mathbf{p}_T$ (m)	1.965	0.204	0.126	0.076

VI. Conclusion

In this paper, we propose a distributed filtering algorithm that cooperatively estimates the 6-DoF poses of a moving object and networked robots with onboard visual-inertial sensors. By using the information from neighboring robots, each robot performs a more accurate and robust tracking of the target object even if it fails to see the target. Common environmental features are exploited to provide prior information which is used to bound the long-term errors of the VIO. Further, we get rid of the pre-designed common global frame which is widely used in the literature regarding multi-robot applications. The communication graph can be time varying with the only requirement that robot 1 should have a directed path to the other robots in the union graph over a time period for transmitting the prior map. When robot 1 stops renewing the prior map, the communication graph can be fully distributed that each robot only needs to communicate with its one-hop neighbors that might be changing over time in the estimators' update steps. The performance of the proposed algorithm has been evaluated by Monte-Carlo simulations.

References

[1] C. X. Guo, K. Sartipi, R. C. DuToit, G. A. Georgiou, R. Li, J. O'Leary, E. D. Nerurkar, J. A. Hesch, and S. I. Roumeliotis, "Large-scale cooperative 3d visual-inertial mapping in a manhattan world," in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, 2016, pp. 1071–1078.

[2] C. X. Guo, K. Sartipi, R. C. DuToit, G. A. Georgiou, R. Li, J. O'Leary, E. D. Nerurkar, J. A. Hesch, and S. I. Roumeliotis, "Resource-aware large-scale cooperative three-dimensional mapping using multiple mobile devices," *IEEE Transactions on Robotics*, vol. 34, no. 5, pp. 1349–1369, 2018.

[3] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 3565–3572.

[4] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige, and R. Siegwart, "Keyframe-based visual-inertial slam using nonlinear optimization," *Proceedings of Robotis Science and Systems*, 2013.

[5] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[6] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular slam with map reuse," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 796–803, 2017.

[7] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2016.

[8] I. V. Melnyk, J. A. Hesch, and S. I. Roumeliotis, "Cooperative vision-aided inertial navigation using overlapping views," in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 936–943.

[9] A. Martinelli, A. Oliva, and B. Mourrain, "Cooperative visual-inertial sensor fusion: the analytic solution," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 453–460, 2019.

[10] A. Martinelli, A. Renzaglia, and A. Oliva, "Cooperative visual-inertial sensor fusion: fundamental equations and state determination in closed-form," *Autonomous Robots*, pp. 1–19, 2019.

[11] M. Karrer, P. Schmuck, and M. Chli, "Cvi-slam—collaborative visual-inertial slam," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2762–2769, 2018.

[12] J. Chen, T. Liu, and S. Shen, "Tracking a moving target in cluttered environments using a quadrotor," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2016, pp. 446–453.

[13] K. Qiu, T. Qin, W. Gao, and S. Shen, "Tracking 3-d motion of dynamic objects using monocular visual-inertial sensing," *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 799–816, 2019.

[14] T. Qin and S. Shen, "Robust initialization of monocular visual-inertial estimation on aerial robots," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2017, pp. 4225–4232.

[15] K. Eckenhoff, Y. Yang, P. Geneva, and G. Huang, "Tightly-coupled visual-inertial localization and 3-d rigid-body target tracking," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1541–1548, 2019.

[16] A. Rozantsev, S. N. Sinha, D. Dey, and P. Fua, "Flight dynamics-based recovery of a uav trajectory using ground cameras," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6030–6039.

[17] M. Vo, S. G. Narasimhan, and Y. Sheikh, "Spatiotemporal bundle adjustment for dynamic 3d reconstruction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1710–1718.

[18] R. Olfati-Saber, "Distributed kalman filtering for sensor networks," in *Proceedings of the IEEE Conference on Decision and Control*. IEEE, 2007, pp. 5492–5498.

[19] A. T. Kamal, J. A. Farrell, and A. K. Roy-Chowdhury, "Information weighted consensus filters and their application in distributed camera networks," *IEEE Transactions on Automatic Control*, vol. 58, no. 12, pp. 3112–3125, 2013.

[20] G. Battistelli, L. Chisci, G. Mugnai, A. Farina, and A. Graziano, "Consensus-based linear and nonlinear filtering," *IEEE Transactions on Automatic Control*, vol. 60, no. 5, pp. 1410–1415, 2014.

[21] J. Hu, L. Xie, and C. Zhang, "Diffusion Kalman filtering based on covariance intersection," *IEEE Transactions on Signal Processing*, vol. 60, no. 2, pp. 891–902, 2011.

[22] G. Huang, M. Kaess, and J. J. Leonard, "Consistent unscented incremental smoothing for multi-robot cooperative target tracking," *Robotics and Autonomous Systems*, vol. 69, pp. 52–67, 2015.

[23] A. Ahmad, G. D. Tipaldi, P. Lima, and W. Burgard, "Cooperative robot localization and target tracking based on least squares minimization," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2013, pp. 5696–5701.

[24] F. M. Mirzaei, A. I. Mourikis, and S. I. Roumeliotis, "On the performance of multi-robot target tracking," in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 3482–3489.

[25] P. Zhu and W. Ren, "Multi-robot joint localization and target tracking with local sensing and communication," in *Proceedings of the American Control Conference*. IEEE, 2019, pp. 3261–3266.

[26] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3. IEEE, 2004, pp. 2149–2154.

[27] W. Breckenridge, "Quaternions proposed standard conventions," *Jet Propulsion Laboratory, Pasadena, CA, Interoffice Memorandum IOM*, pp. 343–79, 1999.

[28] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1280–1286.

[29] M. Li and A. I. Mourikis, "High-precision, consistent ekf-based visual-inertial odometry," *The International Journal of Robotics Research*, vol. 32, no. 6, pp. 690–711, 2013.

[30] A. B. Chatfield, *Fundamentals of high accuracy inertial navigation*. American Institute of Aeronautics and Astronautics, 1997.

[31] P. Geneva, J. Maley, and G. Huang, "An efficient schmidt-ekf for 3d visual-inertial slam," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 105–12 115.

[32] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America*, vol. 4, no. 4, pp. 629–642, 1987.

[33] F. L. Markley, Y. Cheng, J. L. Crassidis, and Y. Oshman, "Averaging quaternions," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 4, pp. 1193–1197, 2007.

[34] P. Zhu and W. Ren, "Distributed kalman filter for 3-d moving object tracking over sensor networks," in *Proceedings of the IEEE Conference on Decision and Control*. IEEE, 2020.

[35] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, "Rotors—a modular gazebo mav simulator framework," in *Robot Operating System (ROS)*. Springer, 2016, pp. 595–625.