UNIVERSITY OF CALIFORNIA
RIVERSIDE

Multi-Robot Cooperative Localization and Target Tracking

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Electrical Engineering

by

Pengxiang Zhu

June 2022

Dissertation Committee:

Dr. Wei Ren, Chairperson
Dr. Jay A.Farrell
Dr. Konstantinos Karydis

The Dissertation of Pengxiang Zhu is approved:

Committee Chairperson

University of California, Riverside

Acknowledgments

First and foremost I am grateful to my advisor Dr. Wei Ren for his invaluable mentoring and continue support over the past five years. Without his help, this work would never have been here. His broad knowledge and insightful thinking have inspired me throughout my PhD study and helped me learn a lot of skills. Also, his diligence and humble personality encourage me to stay hungry and stay foolish. In addition, I would like to thank Dr. Guoquan Huang from the University of Delaware for giving me the opportunity to collaborate with his group. The experience help me dig into the visual-inertial navigation system and make progress in my research. Thanks for all the efforts they made.

I am grateful to the members in my oral and defense committees, Dr. Jay Farrell, Dr. Konstantinos Karydis, Dr. Fabio Pasqualetti, Dr. Matt Barth and Dr. Jiasi Chen for their valuable advice and insightful comments on my research.

My appreciation also goes out to my friends and labmates, who make my PhD life a wonderful experience. Especially, I appreciate the work from Patrick Geneva and Yulin Yang who come from RPNG lab. The experience working with them enhanced my knowledge and promote my research work. My friends, Hongsheng Yu, Xing Zheng and Shaocheng Wang have offered me their assistance in my research project. I also want to thank Yong Ding, Shan Sun, Peng Wang, Shaoshu Su, Runze Li, Zhichao Liu and many other friends at Riverside. They make my stressful PhD life not that boring and help me maintain optimistic and positive through the hard time.

Most importantly, I would like to express my gratitude to my parents. They devoted everything they have to provide me the study opportunities. Without their unwa-

vering support, it would be impossible for me to complete my study.

Finally, I want to acknowledge the financial support from National Science Foundation (Grant no. CMMI-1537729, CMMI-2027139) and the Department of Electrical and Computer Engineering at UCR.

To my parents Shuping Zhu and Fenying Tang for all the support.

ABSTRACT OF THE DISSERTATION

Multi-Robot Cooperative Localization and Target Tracking

by

Pengxiang Zhu

Doctor of Philosophy, Graduate Program in Electrical Engineering
University of California, Riverside, June 2022
Dr. Wei Ren, Chairperson

Sensor networks with the ability of communication and perception has a wide range of applications. They can be utilized to estimate a target's pose even if some sensors are blind to the target. This problem is termed as distributed state estimation (DSE) which has been widely studied. However, existing works are limited to 2-D scenarios with the assumption of *fixed* and *known* sensor states. This manuscript addresses the limitations and extends DSE to the mobile robot case where the robots use onboard sensors to track the target's state.

In particular, in Chapter 2 we study the problem of joint localization and target tracking (JLATT) in 2-D situations. A team of robots simultaneously localize themselves and track multiple targets. Instead of treating localization and target tracking as two separate problems, we explicitly account for the influence of one to the other and exploit it to improve performance in a distributed context. We introduce a fully distributed algorithm that is applicable to generic robot motion, target process and measurement models, and is robust to time-varying sensing and communication typologies.

In the following three chapters, we work on the 3-D scenarios and use the most popular sensor rig – the visual-inertial sensor. Specifically, we first focus on the target tracking and robot localization separately and then work on the visual-inertial JLATT. In chapter 3, a *static* camera network is used to cooperatively estimate the six degree-of-freedom (6-DoF) pose of a moving object. A novel distributed Kalman filter (DKF) is introduced for a general nonlinear system. In chapter 4, we present a multi-robot visual-inertial navigation system (VINS) which achieves cooperative localization (CL) by efficiently fuses environmental features. The algorithm enables drift-free estimation through the use of loop-closure constraints to other robots' historical poses without a significant increase in computational cost. Finally, in chapter 5, we present an algorithm to track a target's state by utilizing a heterogeneous robot network. Rather than assuming a known common global frame for all the robots, we allow each robot to perform motion estimation locally. For localization, 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. The novel DKF is employed to track the pose of the object which is represented as a point cloud.

The research presented in this dissertation aims at extending the application of multi-robots by improving the performance in self-localization and target tracking. The proposed algorithms are demonstrated in Monte-Carlo simulations and experiments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Description

Studies on distributed target estimation using sensor networks have recently become attractive. Each sensor aims to have a good pose (orientation and position) estimator of the target using information among communication neighborhood, even its own cannot observe the target directly. However, most of the existing results address this problem by using a network of static sensors and simply assume that sensors' positions are known. To address this limitation, we utilize a mobile robot network where each robot is equipped with proprioceptive sensors, exteroceptive sensors and communication units. Proprioceptive sensors like wheel encoders and inertial measurement unit (IMU) can measure the self-motion of the robot, while exteroceptive like cameras and lidars can capture the information of surroundings.

As we do not assume known robot poses, high-precision localization especially in GPS-denied environments is a prerequisite for successful target state estimation. We first

consider 2-D cases that the robots use robot-to-robot relative measurements to achieve localization and robot-to-target relative measurements to achieve target tracking. We assume the target follows a known motion model and then jointly estimate the robots' and the targets' poses instead of treating localization and tracking independently. By doing this, the well-estimated target can work as an object reference in the localization part to improve the performance. Stability is studied under very mild conditions.

Next, we focus on 3-D cases. One promising solution to localization is the VINS, which is to fuse both the measurements from cameras and IMUs. These sensors are cheap and light-weight but they are complementary and able to provide rich environmental information, hence enabling highly-accurate motion estimation. The visual-inertial sensor rig has been widely used in robotics, autonomous driving, virtual reality (VR), and augmented reality (AR). Hence, we employ the camera and IMU sensors in the 3-D cases.

Target can be represented as a point cloud in 3-D cases. We can select one point as the representative point and treat the others as non-representative points. The origin of the target's coordinate is chosen as the representative point. Non-representative points also provide constrains of targets' poses. Orientations are represented as unit quaternion and then a distributed Kalman filter which is applicable in SO3 is introduced to track the target's state cooperatively.

By considering that robot-to-robot measurements are difficult and inefficient to obtain when cameras are used, we employ environmental features to perform localization and additional geometric like the common features (features observed by multiple robots) to improve the accuracy. As the goal of the VINS here is to estimate robots' poses and

not to generate a map, VIO is prefer as an efficient solution. We incorporate loop-closure constraints from the historical common features (e.g., a robot can gain information if another robot had previously explored the same location). As a result, the proposed estimator does not require simultaneous viewing of the same features.

Lastly, we study the problem of joint visual-inertial localization and target tracking. To be more realistic, we do not assume the target follows a known motion model. Additionally, a pre-designed common global frame is not required and each robot can perform motion tracking locally. In this part, we adopt a heterogeneous mobile robot network. One robot build a map that is shared in the group while other robots run VIO. Common features in the map are utilized to enable a map-based localization that can bound the navigation drift of VIO. Note that in this system, we separate the localization and tracking parts. This is because the mismatch between the adopt target model and the true unknown model can introduce inconsistency in the localization part which will degrade the performance.

1.2 Key Contributions

1.2.1 Distributed JLATT for generic models

To the best of our knowledge, it is the first time that a distributed algorithm with consistency guarantee is proposed for JLATT in mobile robot networks. Each robot only needs to exchange information with its one-hop communicating neighbors. The algorithm can handle multiple measurements simultaneously, including multiple relative measurements (i.e., robot-to-robot and robot-to-target measurements) as well as absolute measurements if

available. Furthermore, it supports generic robot motion, target process and measurement models. In the case of linearized time-varying systems, it is proved that the estimated error covariances of both robots' poses and targets' states are bounded under certain mild conditions on the sensing and communication graphs and system observability. To the best of our knowledge, it is the first time that the stability for CL or JLATT in mobile robot networks is analyzed in a distributed setting, even for the linearized system.

1.2.2 Distributed 3-D Target State Estimation

The existing DKF can only work on vector space which means it can only use Euler angle representation when estimating the orientation. However, Euler angle expression suffers from singularities. So the main contribution of this chapter is that a novel DKF suitable for the 3-D DSE over sensor networks is proposed and further applied to camera networks. The proposed approach is fully distributed and applicable for generic target motion and measurement models. It can also handle time-varying communication typologies and changing blind agents.

1.2.3 Distributed Visual-Inertial Cooperative Localization

We propose a fully distributed multi-robot visual-inertial CL estimator by delicately exploiting information contained in both environmental landmarks and loop-closures across robots and time. Specifically, We develop a fully distributed CI-based visual-inertial CL estimation algorithm, which allows for accurate, efficient and consistent estimation of all robot states. We propose two different SLAM feature measurement models that allow for cooperative estimation of common long-lived environmental features, and validate their

relative accuracy and computational complexity through a series of simulations. We introduce a computationally efficient method for long-term loop-closure to reduce localization drift, which enables multi-robot constraints between historical poses and features, allowing for robots to gain additional constraints even in the case when other robots are not actively in the same location. We thoroughly validate the proposed approach in Monte Carlo simulations and real world experiments by comparing to centralized CL algorithms.

1.2.4 Distributed Visual-Inertial JLATT

The proposed distributed algorithm consists of multi-robot VINS and cooperative target state tracking. We formulate the JLATT problem in a more realistic scenario with the monocular visual-inertial setting. Specifically, we consider changing communication typologies and dynamic blind robots (the robots losing sight of the whole 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. The novel DKF is used to achieve 6-DoF target tracking of a moving object whose motion model is unknown. 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 *unknown* actual one.

1.3 Organization of the Manuscript

The Manuscript is organized as follows. In the following chapter, we study the JLATT problem under generic motion and measurement models. A distributed estimator is proposed and its stability is studied. The effectiveness is validated using simulations and experiments in 2-D environments. In Chapter 3, we focus on estimating the 6-Dof pose of a rigid body target using fixed sensor networks whose states are assumed known. Simulations shows the performance of the proposed DKF. In Chapter 4, we study the visual-inertial CL problem. A distributed cooperative SLAM algorithm is proposed. The efficiency and accuracy are demonstrated in extensive simulations and experiments. Next, in Chapter 5, we introduce a distributed algorithm to achieve JLATT in 3-D scenarios by using visual-inertial sensors. We validate the proposed algorithm in synthesized datasets. Finally, Chapter 6 summarizes the main results of this manuscript.

Chapter 2

Distributed Joint Localization and Target Tracking (Generic Models)

2.1 Introduction and Related Works

Sensor networks with the ability to communicate, sense, and interact with surroundings have a wide range of applications such as region monitoring, area surveillance, and search and rescue. When mobile robots equipped with sensors are employed, a large area can be covered without the need to increase the number of sensors in the network. Also, the robots can actively pursue targets and prevent them escaping from the sensing regions of their onboard sensors. In this chapter, a team of, possibly heterogeneous, mobile robots is employed to track multiple targets in a fully or intermittently absolute measurement (e.g., GPS data) denied environment. To perform this task, distributed strategies outperform centralized approaches in scalability, energy (e.g., processing and communica-

tion) efficiency and robustness against failures or attacks. In particular, we aim to propose a fully distributed algorithm with only local information and local communication in the absence of global parameters and multi-hop information propagation or flooding. As we do not assume *a prior* known information about the robots' poses, to successfully track the targets, it is necessary for the robots to determine their poses precisely. Cooperative Localization (CL) is a widely used technique to achieve multi-robot localization in the absence of absolute measurements. In CL, by cooperating with other robots, each robot can estimate its own pose using relative measurements (e.g., relative distance, bearing, relative pose or any combination of them) between robots. In particular, distributed CL has gathered significant attention in robotics. However, distributed centralized-equivalent algorithms presented in [109, 62, 6, 121, 79] are not fully distributed. At each occurrence of the measurements, some variables need to be shared among the team through information propagation rather than purely one-hop neighbor-to-neighbor communication. For example, a distributed algorithm equivalent to the centralized *Extended Kalman Filter* (EKF) is presented in [109]. But the measurements obtained by one robot are required to be transmitted to all teammates. For another instance, [62] introduces new intermediate local variables to decouple the propagation stage of the EKF. However, the communication graph is required to have a spanning tree rooted at the interim master so as to propagate the intermediate local variables to the rest of the team through multiple hops in one time step. To relax the communication limitations in centralized-equivalent approaches, [100] presents an EKF-based distributed algorithm to handle asynchronous communication. But the cross-correlations between robots are ignored, which leads to inconsistent estimates.

In contrast, the distributed algorithm proposed in [75] is able to approximate the cross-correlations between robots. Nevertheless, the estimate is not guaranteed to be consistent. The *Covariance Intersection* (CI) technique is used in [20] to compute a consistent estimate. However, the estimate requires a particular measurement model, specifically, the relative poses of neighbors. The *Interleaved Update* (IU) algorithm in [5] can handle generic models and compute consistent estimates. Nevertheless each robot in a team of M robots has to maintain 2^M filters and keep tracking the origin of the measurements. Besides the above mentioned limitations, all aforementioned approaches do not consider robots working in a dynamic environment where moving targets exist and hence ignore the effect resulting from jointly estimating the targets' states. Also, there exist some cases where robots need to co-work with targets (e.g., humans) and then it is essential to estimate the poses of targets in addition to localizing themselves.

In another aspect, many algorithms have been proposed to address the distributed target estimation problem with sensor networks. Each sensor fuses local information with information from its neighbors to estimate the state of a common target. Current approaches, either consensus-based or diffusion-based algorithms, solve the tracking problem using a static sensor network where the sensors' positions are assumed to be known explicitly or implicitly [97, 57, 108, 10, 98, 15, 124, 22, 46].

However, there exist several approaches for solving the problem of joint localization and target tracking (JLATT). Mobile robots are adopted in [49, 1, 90]. A consistent *Unscented Incremental Smoothing* algorithm is introduced in [49] by enforcing the observability constraint on the unscented transformation. In [1], the problem is modeled under

a least square minimization framework, where the states of the robots, the targets and static landmarks are jointly estimated. To mitigate, not avoid, the risk of using the measurements more than once, a common reference is defined by using static landmarks which might be unavailable. By assuming that robots have access to the measurements of absolute orientations, an EKF-based approach is presented in [90]. Furthermore, it is analytically shown that jointly estimating the robot and target positions results in better accuracy of the robots' position estimates in the steady state, in comparison to the CL. It is worth noting that the algorithms mentioned above are all *centralized*.

A distributed algorithm for JLATT is presented in [4], where static sensors are used. The sensors are localized via a Jacobi algorithm that computes the *Best Linear Unbiased estimates* in a distributed matter. In order to use the Jacobi algorithm, the measurements between sensors are required to have a particular linear model. Also, each sensor has to maintain a history of the average measurements. As the number of sensors increases, the storage and computational costs increase dramatically. In addition, a distributed Kalman filter is designed to estimate the target's state. Here, only the prior estimates from neighbors are used and the neighbors' relative measurements to the target are neglected. As a result, some useful information might be lost. Although this approach is distributed, it is limited to static sensor networks where each sensor's state is a static parameter to be estimated. When a mobile robot network is employed, each robot propagates its pose according to a noisy motion model. The state estimates of two robots or one robot and one target become correlated after updating the estimates using the relative measurements between them. Note that directly fusing these two estimates would yield an inconsistent estimate. Then,

there exist significant challenges to avoid information double-counting between robots and account for the coupling between localization and target tracking.

The above observations motivate us to derive a fully distributed algorithm for JLATT with mobile robot networks. We explicitly account for the mutual influence between localization and target tracking and exploit it to improve performance in a fully distributed way. In terms of stability analysis, it is worth pointing out that few works analyze the stability in CL while all the works on target tracking are limited to static sensor networks. We aim to jointly analyze the stability in both the localization and tracking parts. Our approach is based on two fully distributed estimates and able to track multiple targets by using mobile robots whose poses are unknown.

2.2 Preliminaries

2.2.1 Notations and Definitions

Let the vector $\mathbf{x}^k$ represent the actual pose of a robot or the actual state of a target at time k. Given a real-valued $\mathbf{x}^k$, the prior estimate is $\bar{\mathbf{x}}^k$ and the posterior estimate is $\hat{\mathbf{x}}^k$. Denote, respectively, $\bar{\mathbf{e}}^k = \mathbf{x}^k - \bar{\mathbf{x}}^k$ and $\mathbf{e}^k = \mathbf{x}^k - \hat{\mathbf{x}}^k$, the prior and posterior estimation errors. Then, we use $\bar{\mathbf{p}}^k$ and $\mathbf{p}^k$ to represent, respectively, the estimated covariance of $\bar{\mathbf{e}}^k$ and $\mathbf{e}^k$. We distinguish the variables associated with robot i's self estimate by the subscript R_i, e.g., $\bar{\mathbf{x}}^k_{R_i}$ representing robot i's prior estimate of its own actual pose $\mathbf{x}^k_{R_i}$ and $\bar{\mathbf{p}}^k_{R_i}$ representing the estimated covariance of $\bar{\mathbf{e}}^k_{R_i}$ with $\bar{\mathbf{e}}^k_{R_i} = \mathbf{x}^k_{R_i} - \bar{\mathbf{x}}^k_{R_i}$. Further, we distinguish the variables associated with robot i's estimate of target j by the subscript T_{ij}, e.g., $\bar{\mathbf{x}}^k_{T_{ij}}$ denoting robot i's prior estimate of target j's actual state $\mathbf{x}^k_{T_j}$ and $\bar{\mathbf{p}}^k_{T_{ij}}$ denoting the estimated covariance

of $\bar{\mathbf{e}}^k_{T_{ij}}$ with $\bar{\mathbf{e}}^k_{T_{ij}} = \mathbf{x}^k_{T_j} - \bar{\mathbf{x}}^k_{T_{ij}}$. We denote by $\mathbf{I}_n$ the identity matrix of dimension $n \times n$. The superscript T denotes transpose and superscript -1 represents inverse.

$\mathbb{E}\{\cdot\}$ computes the expectation of a random variable. $\mathrm{Diag}\{\cdot\}$ and $\mathrm{Max}\{\cdot\}$ denote, respectively, the block-diagonal matrix constructed from the elements and the maximum of the elements. We let $\mathrm{Tr}\{\cdot\}$ denote the trace of a matrix. The interval of time instants $\mathcal{T}^{k_n}_{k_0}$ is defined as $[k_0, \ldots, k_n]$, where $0 \leq k_0 < k_n < \infty$. For symmetric matrices $\mathbf{A}$ and $\mathbf{B}$, the notation $\mathbf{A} \geq \mathbf{B}$ (or $\mathbf{A} > \mathbf{B}$) means that $\mathbf{A}-\mathbf{B}$ is positive semidefinite (or definite). For finite sets $\mathcal{A}$ and $\mathcal{B}$, we denote by $\mathcal{A}\backslash\mathcal{B}$ the set whose elements include all elements in $\mathcal{A}$ that are not in $\mathcal{B}$. The transition matrix on $\mathcal{T}^{\tau}_{k_0}$, $\mathbf{\Phi}(\tau, k_0)$, is defined as $\mathbf{\Phi}(\tau, k_0) = \mathbf{\Phi}^{\tau-1}_{R_i}, \ldots, \mathbf{\Phi}^{k_0+1}_{R_i}\mathbf{\Phi}^{k_0}_{R_i}$ and $\mathbf{\Phi}(k_0, k_0)$ is the identity matrix.

Definition 1 *[54] Suppose that $\mathbf{x}^k$ is a random variable. Let $\hat{\mathbf{x}}^k$ and $\mathbf{p}^k$ be, respectively, the estimate of $\mathbf{x}^k$ and the estimated error covariance. The pair ($\hat{\mathbf{x}}^k$,$\mathbf{p}^k$) is said to be consistent if the actual error covariance $\mathbb{E}\{\mathbf{e}^k(\mathbf{e}^k)^{\mathsf{T}}\} \leq \mathbf{p}^k$.*

The consistency is a critical property of estimates that the estimated error covariances realistically expresses the covariance of actual errors. In contrast, an inconsistent estimate that underestimates the actual errors might diverge as a result [86, 3].

2.2.2 Graphs

In the network of M robots, we define a directed communication graph $G^k_c = (\mathcal{V}, \mathcal{E}^k_c)$, where $\mathcal{V} = \{R_1, \ldots, R_M\}$ is the robot set and $\mathcal{E}^k_c \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set, which stands for the communication links between robots at time k. We assume that self edge $(i, i) \in \mathcal{E}^k_c$, $\forall i \in \mathcal{V}$, exists in the communication graph. If there exists an edge $(j, i) \in \mathcal{E}^k_c$,

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. At time k, the communicating neighbor set of robot i is defined as $\mathcal{N}_{c,i}^k = \{i|(l,i) \in \mathcal{E}_c^k, \forall l \neq i, l \in \mathcal{V}\}$; The inclusive communicating neighbor set of robot i is $\mathcal{J}_{c,i}^k = \mathcal{N}_{c,i}^k \cup \{i\}$.

Similarly, we define a directed sensing graph $G_s^k = (\mathcal{V}, \mathcal{E}_s^k)$ to describe robot-to-robot measurements, where $\mathcal{E}_s^k \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set, which stands for the detection links between robots at time k. For example, when robot i detects robot j at time k, there exists an edge (j,i) directed from robot j to robot i in $\mathcal{E}_s^k$. At time k, we denote the sensing neighbor set of robot i by $\mathcal{N}_{s,i}^k = \{i|(l,i) \in \mathcal{E}_s^k, \forall l \neq i, l \in \mathcal{V}\}$ (i.e., all robots detected by robot i). 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 $i_j \in \mathcal{V}$. Besides, the set of N targets is denoted by $\mathcal{U} = \{T_1, \ldots, T_N\}$ and the subset of targets detected by robot i at time k is denoted by $\mathcal{U}_i^k$. We assume that for each robot, the communication radius is larger than the sensing radii of all robots. Then when robot i detects robot j, robot i can receive the information broadcast by robot j.

2.2.3 Track-to-Track Fusion

Track-to-track fusion is the problem of combining multiple estimates of a state into a single and more accurate estimate. At time k, consider two consistent estimation pairs $(\mathbf{p}_{a_1}^k, \hat{\mathbf{x}}_{a_1}^k)$ and $(\mathbf{p}_{a_2}^k, \hat{\mathbf{x}}_{a_2}^k)$ of $\mathbf{x}^k$, we seek to compute an improved consistent estimate $(\mathbf{p}_c^k, \hat{\mathbf{x}}_c^k)$. The cross-correlation between $\hat{\mathbf{x}}_{a_1}^k$ and $\hat{\mathbf{x}}_{a_2}^k$ is denoted as $\mathbf{p}_{a_1 a_2}^k$. If $\mathbf{p}_{a_1 a_2}^k$ is known,

the consistent fused estimate with minimum covariance is given by [7]

$$\mathbf{p}_c^k = \mathbf{p}_{a_1}^k - (\mathbf{p}_{a_1}^k - \mathbf{p}_{a_1a_2}^k)[\mathbf{p}_{a_1}^k + \mathbf{p}_{a_2}^k - \mathbf{p}_{a_1a_2}^k - (\mathbf{p}_{a_1a_2}^k)^{\mathsf{T}}]^{-1}[\mathbf{p}_{a_1}^k - (\mathbf{p}_{a_1a_2}^k)^{\mathsf{T}}],$$

$$\hat{\mathbf{x}}_c^k = \hat{\mathbf{x}}_{a_1}^k + (\mathbf{p}_{a_1}^k - \mathbf{p}_{a_1a_2}^k)[\mathbf{p}_{a_1}^k + \mathbf{p}_{a_2}^k - \mathbf{p}_{a_1a_2}^k - (\mathbf{p}_{a_1a_2}^k)^{\mathsf{T}}]^{-1}(\hat{\mathbf{x}}_{a_2}^k - \hat{\mathbf{x}}_{a_1}^k).$$

Further, if $\hat{\mathbf{x}}_{a_1}^k$ and $\hat{\mathbf{x}}_{a_2}^k$ are independent, by setting $\mathbf{p}_{a_1a_2}^k = 0$, we have

$$\begin{aligned} (\mathbf{p}_c^k)^{-1} &= (\mathbf{p}_{a_1}^k)^{-1} + (\mathbf{p}_{a_2}^k)^{-1}, \\ \hat{\mathbf{x}}_c^k &= \mathbf{p}_c^k[(\mathbf{p}_{a_1}^k)^{-1}\hat{\mathbf{x}}_{a_1}^k + (\mathbf{p}_{a_2}^k)^{-1}\hat{\mathbf{x}}_{a_2}^k]. \end{aligned} \tag{2.1}$$

On the other hand, if $\mathbf{p}_{a_1a_2}^k$ is unknown, CI, a well-known conservative fusion scheme that yields a consistent fused estimate, is given as follows [54]

$$\begin{aligned} [\mathbf{p}_c^k]_{\mathrm{CI}} &= \left[\alpha_1^k(\mathbf{p}_{a_1}^k)^{-1} + (1-\alpha_1^k)(\mathbf{p}_{a_2}^k)^{-1}\right]^{-1}, \\ [\hat{\mathbf{x}}_c^k]_{\mathrm{CI}} &= [\mathbf{p}_c^k]_{\mathrm{CI}}\left[\alpha_1^k(\mathbf{p}_{a_1}^k)^{-1}\hat{\mathbf{x}}_{a_1}^k + (1-\alpha_1^k)(\mathbf{p}_{a_2}^k)^{-1}\hat{\mathbf{x}}_{a_2}^k\right], \end{aligned} \tag{2.2}$$

where $\alpha_1^k \in [0,1]$. Compared with CI, the recently proposed *Inverse Covariance Intersection* (ICI) [96] provides an optimal consistent and tight solution and therefore is more accurate. The ICI is given as [96]

$$\begin{aligned} [\mathbf{p}_c^k]_{\mathrm{ICI}} &= \left[(\mathbf{p}_{a_1}^k)^{-1} + (\mathbf{p}_{a_2}^k)^{-1} - (\mathbf{\Gamma}_c^k)^{-1}\right]^{-1}, \\ [\hat{\mathbf{x}}_c^k]_{\mathrm{ICI}} &= [\mathbf{p}_c^k]_{\mathrm{ICI}}\left(\mathbf{K}_c^k\hat{\mathbf{x}}_{a_1}^k + \mathbf{L}_c^k\hat{\mathbf{x}}_{a_2}^k\right). \end{aligned} \tag{2.3}$$

where

$$\mathbf{\Gamma}_c^k = \alpha_2^k\mathbf{p}_{a_1}^k + (1-\alpha_2^k)\mathbf{p}_{a_2}^k,$$

$$\mathbf{K}_c^k = (\mathbf{p}_{a_1}^k)^{-1} - \alpha_2^k(\mathbf{\Gamma}_c^k)^{-1},$$

$$\mathbf{L}_c^k = (\mathbf{p}_{a_2}^k)^{-1} - (1-\alpha_2^k)(\mathbf{\Gamma}_c^k)^{-1},$$

for any $\alpha_2^k \in [0,1]$. The time-varying parameters α_1^k and α_2^k can be chosen to minimize an optimality criterion such as the traces of $[\mathbf{p}_c^k]_{\mathrm{CI}}$ and $[\mathbf{p}_c^k]_{\mathrm{ICI}}$, respectively. $(\mathbf{\Gamma}_c^k)^{-1}$ can be considered a tight outer bound of the common information.

Lemma 2 *[96] Let $[\mathbf{p}_c^k]_{CI}^*$ and $[\mathbf{p}_c^k]_{ICI}^*$ be, respectively, the fused covariances with minimal traces by using CI and ICI at time k. Then $[\mathbf{p}_c^k]_{ICI}^* \leq [\mathbf{p}_c^k]_{CI}^*$.*

CI is generalized to fuse an arbitrary number of estimation pairs $(\mathbf{p}_{a_i}^k, \hat{\mathbf{x}}_{a_i}^k)$, $i = 1, \ldots, n$, according to [53]

$$\begin{aligned} [\mathbf{p}_c^k]_{\mathrm{CI}} &= \left[\sum_{i=1}^{n} \alpha_i^k (\mathbf{p}_{a_i}^k)^{-1}\right]^{-1}, \\ [\hat{\mathbf{x}}_c^k]_{\mathrm{CI}} &= [\mathbf{p}_c^k]_{\mathrm{CI}} \left[\sum_{i=1}^{n} \alpha_i^k (\mathbf{p}_{a_i}^k)^{-1} \hat{\mathbf{x}}_{a_i}^k\right], \end{aligned} \tag{2.4}$$

where $\alpha_i^k \in [0, 1]$ and $\sum_{i=1}^{n} \alpha_i^k = 1$. For the sake of computational simplicity, we use the simplified algorithm in [95] to calculate α_i^k as

$$\alpha_i^k = \frac{1/\mathrm{Tr}\{\mathbf{p}_{a_i}^k\}}{\sum_{i=1}^{n} 1/\mathrm{Tr}\{\mathbf{p}_{a_i}^k\}}. \tag{2.5}$$

2.2.4 Problem Formulation

Consider a group of M heterogeneous mobile robots and N targets moving within the same space. Each robot carries proprioceptive sensors (e.g., odometries) to measure its self-motion and exteroceptive sensors (e.g., cameras or laser scanners) to generate relative measurements to other robots as well as multiple targets. Besides, some robots might have access to the absolute measurements intermittently. The motion of robot i is described by a nonlinear model

$$\mathbf{x}_{R_i}^k = \mathbf{f}_i(\mathbf{x}_{R_i}^{k-1}, \mathbf{u}_{R_i}^{k-1} - \mathbf{w}_{R_i}^{k-1}), \tag{2.6}$$

where $\mathbf{x}_{R_i}^k$, $\mathbf{u}_{R_i}^k$, and $\mathbf{w}_{R_i}^k$ are, respectively, the ith robot's pose (position and orientation), the measured input, and the process noise at time k. We assume that the noise $\mathbf{w}_{R_i}$ is zero-mean white Gaussian.

The state of target j at time k is represented by $\mathbf{x}_{T_i}^k$, which might contain the target's pose or velocity components. The process model of target j is given as

$$\mathbf{x}_{T_j}^k = \mathbf{g}_j(\mathbf{x}_{T_j}^{k-1}, \mathbf{w}_{T_j}^{k-1}), \tag{2.7}$$

where $\mathbf{w}_{T_j}$ is the process noise, assumed to be zero-mean white Gaussian.

At time k, if robot j (respectively, target j) is within the sensing region of robot i, robot i obtains the robot-to-robot measurement $\mathbf{z}_{R_{ij}}^k$ (respectively, robot-to-target measurement $\mathbf{z}_{T_{ij}}^k$). If accessible, robot i receives the absolute measurement $\mathbf{z}_{R_i}^k$. We model the collected measurements as

$$\begin{aligned} \mathbf{z}_{R_{ij}}^k &= \mathbf{h}_{ij}^r(\mathbf{x}_{R_i}^k, \mathbf{x}_{R_j}^k) + \mathbf{v}_{R_{ij}}^k, \\ \mathbf{z}_{T_{ij}}^k &= \mathbf{h}_{ij}^t(\mathbf{x}_{R_i}^k, \mathbf{x}_{T_j}^k) + \mathbf{v}_{T_{ij}}^k, \\ \mathbf{z}_{R_i}^k &= \mathbf{h}_i^a(\mathbf{x}_{R_i}^k) + \mathbf{v}_{R_i}^k, \end{aligned} \tag{2.8}$$

where $\mathbf{v}_{R_{ij}}^k$, $\mathbf{v}_{T_{ij}}^k$, and $\mathbf{v}_{R_i}^k$ are the corresponding measurement noises, assumed to be zero-mean white Gaussian. The covariance matrices are, respectively, represented as $\mathbf{R}_{R_{ij}}^k = \mathbf{E}[\mathbf{v}_{R_{ij}}^k(\mathbf{v}_{R_{ij}}^k)^\mathsf{T}]$, $\mathbf{R}_{T_{ij}}^k = \mathbf{E}[\mathbf{v}_{T_{ij}}^k(\mathbf{v}_{T_{ij}}^k)^\mathsf{T}]$ and $\mathbf{R}_{R_i}^k = \mathbf{E}[\mathbf{v}_{R_i}^k(\mathbf{v}_{R_i}^k)^\mathsf{T}]$. Note that at any time, some robots might not be able to obtain any relative or absolute measurement. Further, we assume that the measurement noises are mutually uncorrelated across robots and are uncorrelated with the process noises.

The objective of our work is for each robot i to construct estimates of its own pose and of each target's state by using its local measurements if available and the information received from its one-hop communicating neighbors at the current time.

2.3 Proposed Fully Distributed Algorithm

In this section, we derive a fully distributed scheme for JLATT from the perspective of *Extended Information Filter* (EIF), the information form of Kalman filter.

2.3.1 Distributed Extended Information Filtering

Localization

Unlike the existing works on distributed target estimation with static sensor networks where the pose of each sensor is deterministic and known, we consider the general scenario where the poses of the mobile robots (serving as mobile sensors) are states to be estimated. Robot i estimates $\mathbf{x}_{R_i}^k$ by using its available relative measurements to other robots and targets. At time k, when robot i detects another robot $l \in \mathcal{V}$, robot i obtains the relative measurement $\mathbf{z}_{R_{il}}^k$ and receives the information broadcast by robot l. The broadcast information contains robot l's current pose estimate $\bar{\mathbf{x}}_{R_l}^k$ with estimated covariance $\bar{\mathbf{p}}_{R_l}^k$. After linearization of the measurement $\mathbf{z}_{R_{il}}^k$ at $\bar{\mathbf{x}}_{R_i}^k$ and $\bar{\mathbf{x}}_{R_l}^k$, we compute the measurement residual

$$\bar{\mathbf{z}}_{R_{il}}^k = \mathbf{H}_{R_{il}}^k \bar{\mathbf{e}}_{R_i}^k + \widetilde{\mathbf{H}}_{R_{il}}^k \bar{\mathbf{e}}_{R_l}^k + \mathbf{v}_{R_{il}}^k, \tag{2.9}$$

where $\bar{\mathbf{z}}_{R_{il}}^k = \mathbf{z}_{R_{il}}^k - \mathbf{h}_{il}^r(\bar{\mathbf{x}}_{R_i}^k, \bar{\mathbf{x}}_{R_l}^k)$ with $\mathbf{H}_{R_{il}}^k = \frac{\partial \mathbf{h}_{il}^r}{\partial \mathbf{x}_{R_i}}(\bar{\mathbf{x}}_{R_i}^k, \bar{\mathbf{x}}_{R_l}^k)$ and $\widetilde{\mathbf{H}}_{R_{il}}^k = \frac{\partial \mathbf{h}_{il}^r}{\partial \mathbf{x}_{R_l}}(\bar{\mathbf{x}}_{R_i}^k, \bar{\mathbf{x}}_{R_l}^k)$. By defining $\bar{\mathbf{v}}_{R_{il}}^k = \widetilde{\mathbf{H}}_{R_{il}}^k \bar{\mathbf{e}}_{R_l}^k + \mathbf{v}_{R_{il}}^k$, we get

$$\bar{\mathbf{z}}_{R_{il}}^k = \mathbf{H}_{R_{il}}^k \bar{\mathbf{e}}_{R_i}^k + \bar{\mathbf{v}}_{R_{il}}^k.$$

The corresponding covariance for $\bar{\mathbf{v}}_{R_{il}}^k$ is given by

$$\bar{\mathbf{R}}_{R_{il}}^k = \mathbf{R}_{R_{il}}^k + \widetilde{\mathbf{H}}_{R_{il}}^k \bar{\mathbf{p}}_{R_l}^k (\widetilde{\mathbf{H}}_{R_{il}}^k)^\mathsf{T}, \tag{2.10}$$

which has included the uncertainty of robot l's pose estimate. Then, define the relative correction pair $(\mathbf{s}^k_{R_{il}}, \mathbf{y}^k_{R_{il}})$ as

$$\begin{aligned} \mathbf{s}^k_{R_{il}} &= (\mathbf{H}^k_{R_{il}})^{\mathsf{T}}(\bar{\mathbf{R}}^k_{R_{il}})^{-1}\mathbf{H}^k_{R_{il}}, \\ \mathbf{y}^k_{R_{il}} &= (\mathbf{H}^k_{R_{il}})^{\mathsf{T}}(\bar{\mathbf{R}}^k_{R_{il}})^{-1}(\bar{\mathbf{z}}^k_{R_{il}} + \mathbf{H}^k_{R_{il}}\bar{\mathbf{x}}^k_{R_i}). \end{aligned} \tag{2.11}$$

Similarly, when robot i detects target $j \in \mathcal{U}$, robot i obtains the relative measurement $\mathbf{z}^k_{T_{ij}}$. After linearization of the measurement $\mathbf{z}^k_{T_{ij}}$ at $\bar{\mathbf{x}}^k_{R_i}$ and $\bar{\mathbf{x}}^k_{T_{ij}}$, we compute the measurement residual

$$\bar{\mathbf{z}}^k_{T_{ij}} = \mathbf{H}^k_{T_{ij}}\bar{\mathbf{e}}^k_{R_i} + \widetilde{\mathbf{H}}^k_{T_{ij}}\bar{\mathbf{e}}^k_{T_{ij}} + \mathbf{v}^k_{T_{ij}}, \tag{2.12}$$

where $\bar{\mathbf{z}}^k_{T_{ij}} = \mathbf{z}^k_{T_{ij}} - \mathbf{h}^t_{ij}(\bar{\mathbf{x}}^k_{R_i}, \bar{\mathbf{x}}^k_{T_{ij}})$ with $\mathbf{H}^k_{T_{ij}} = \frac{\partial \mathbf{h}^t_{ij}}{\partial \mathbf{x}_{R_i}}(\bar{\mathbf{x}}^k_{R_i}, \bar{\mathbf{x}}^k_{T_{ij}})$ and $\widetilde{\mathbf{H}}^k_{T_{ij}} = \frac{\partial \mathbf{h}^t_{ij}}{\partial \mathbf{x}_{T_{ij}}}(\bar{\mathbf{x}}^k_{R_i}, \bar{\mathbf{x}}^k_{T_{ij}})$. By Defining $\bar{\mathbf{v}}^k_{T_{ij}} = \widetilde{\mathbf{H}}^k_{T_{ij}}\bar{\mathbf{e}}^k_{T_{ij}} + \mathbf{v}^k_{T_{ij}}$, we get

$$\bar{\mathbf{z}}^k_{T_{ij}} = \mathbf{H}^k_{T_{ij}}\bar{\mathbf{e}}^k_{R_i} + \bar{\mathbf{v}}^k_{T_{ij}}.$$

The corresponding covariance of $\bar{\mathbf{v}}^k_{T_{ij}}$ is given by

$$\bar{\mathbf{R}}^k_{T_{ij}} = \mathbf{R}^k_{T_{ij}} + \widetilde{\mathbf{H}}^k_{T_{ij}}\bar{\mathbf{p}}^k_{T_{ij}}(\widetilde{\mathbf{H}}^k_{T_{ij}})^{\mathsf{T}}. \tag{2.13}$$

Then, define the relative correction pair $(\mathbf{s}^k_{T_{ij}}, \mathbf{y}^k_{T_{ij}})$ as

$$\begin{aligned} \mathbf{s}^k_{T_{ij}} &= (\mathbf{H}^k_{T_{ij}})^{\mathsf{T}}(\bar{\mathbf{R}}^k_{T_{ij}})^{-1}\mathbf{H}^k_{T_{ij}}, \\ \mathbf{y}^k_{T_{ij}} &= (\mathbf{H}^k_{T_{ij}})^{\mathsf{T}}(\bar{\mathbf{R}}^k_{T_{ij}})^{-1}(\bar{\mathbf{z}}^k_{T_{ij}} + \mathbf{H}^k_{T_{ij}}\bar{\mathbf{x}}^k_{R_i}). \end{aligned} \tag{2.14}$$

Note that unlike (2.11), no communication is needed to compute (2.14) as $(\bar{\mathbf{p}}^k_{T_{ij}}, \bar{\mathbf{x}}^k_{T_{ij}})$, $\bar{\mathbf{x}}^k_{R_i}$ and $\mathbf{z}^k_{T_{ij}}$ are all available at robot i.

Remark 3 *As shown in* (2.10) *and* (2.13)*, the noise covariances* $\mathbf{R}^k_{R_{il}}$ *and* $\mathbf{R}^k_{T_{ij}}$ *are, respectively, suitably increased by a positive semidefinite quantity* $\widetilde{\mathbf{H}}^k_{R_{il}}\bar{\mathbf{p}}^k_{R_l}(\widetilde{\mathbf{H}}^k_{R_{il}})^{\mathsf{T}}$ *and* $\widetilde{\mathbf{H}}^k_{T_{ij}}\bar{\mathbf{p}}^k_{T_{ij}}(\widetilde{\mathbf{H}}^k_{T_{ij}})^{\mathsf{T}}$.

As a result, a large uncertainty in robot l's pose or target j's state leads to a large $\bar{\mathbf{R}}^k_{R_{il}}$ or $\bar{\mathbf{R}}^k_{T_{ij}}$, which makes $(\mathbf{s}^k_{R_{il}}, \mathbf{y}^k_{R_{il}})$ or $(\mathbf{s}^k_{T_{ij}}, \mathbf{y}^k_{T_{ij}})$ small. Then the influence caused by the corresponding inaccurate measurements will be alleviated.

At time k, if robot i has access to its absolute measurement $\mathbf{z}^k_{R_i}$, the measurement residual after linearization at $\bar{\mathbf{x}}^k_{R_i}$ is given by

$$\bar{\mathbf{z}}^k_{R_i} = \mathbf{C}^k_{R_i} \bar{\mathbf{e}}^k_{R_i} + \mathbf{v}^k_{R_i}, \tag{2.15}$$

where $\bar{\mathbf{z}}^k_{R_i} = \mathbf{z}^k_{R_i} - \mathbf{h}^a_i(\bar{\mathbf{x}}^k_{R_i})$ with $\mathbf{C}^k_{R_i} = \frac{\partial \mathbf{h}^a_i}{\partial \mathbf{x}_{R_i}}(\bar{\mathbf{x}}^k_{R_i})$. For notation convenience, if robot i's absolute measurement is not accessible, we let $\mathbf{R}^k_{R_i} = \infty$, which assumes infinite uncertainty about $\mathbf{z}^k_{R_i}$. Then we denote the absolute correction pair $(\mathbf{s}^k_{R_{ii}}, \mathbf{y}^k_{R_{ii}})$ as

$$\begin{aligned} \mathbf{s}^k_{R_{ii}} &= (\mathbf{C}^k_{R_i})^\mathsf{T} (\mathbf{R}^k_{R_i})^{-1} \mathbf{C}^k_{R_i}, \\ \mathbf{y}^k_{R_{ii}} &= (\mathbf{C}^k_{R_i})^\mathsf{T} (\mathbf{R}^k_{R_i})^{-1} (\bar{\mathbf{z}}^k_{R_i} + \mathbf{C}^k_{R_i} \bar{\mathbf{x}}^k_{R_i}). \end{aligned} \tag{2.16}$$

Next, the task is to compute the posterior estimate $\hat{\mathbf{x}}^k_{R_i}$ with the estimated covariance $\mathbf{p}^k_{R_i}$ from the available correction pairs and the prior estimate $\bar{\mathbf{x}}^k_{R_i}$ with the estimated covariance $\bar{\mathbf{p}}^k_{R_i}$. Although the relative measurement noises are mutually uncorrelated, the defined $\bar{\mathbf{v}}^k_{R_{il}}$ and $\bar{\mathbf{v}}^k_{T_{ij}}$ are correlated for different l and j due to the correlations between the estimates of the robots' poses as well as the targets' states.

Accordingly, the corresponding relative correction pairs are correlated. We apply the CI algorithm (2.4) on the relative correction pairs to guarantee consistency with the simplified weight selection strategy (2.5). The absolute correction pair can be directly incorporated, as it is uncorrelated with the relative correction pairs. Therefore, at time k,

we can compute an estimate of $\mathbf{x}_{R_i}^k$ by using all available correction pairs. We have

$$\breve{\mathbf{p}}_{R_i}^k = (\sum_{l\in\mathcal{N}_{s,i}^k} \eta_{il}^k \mathbf{s}_{R_{il}}^k + \sum_{j\in\mathcal{U}_i^k} \eta_{ij}^k \mathbf{s}_{T_{ij}}^k + \mathbf{s}_{R_{ii}}^k)^{-1}, \tag{2.17a}$$

$$\breve{\mathbf{x}}_{R_i}^k = \breve{\mathbf{p}}_{R_i}^k(\sum_{l\in\mathcal{N}_{s,i}^k} \eta_{il}^k \mathbf{y}_{R_{il}}^k + \sum_{j\in\mathcal{U}_i^k} \eta_{ij}^k \mathbf{y}_{T_{ij}}^k + \mathbf{y}_{R_{ii}}^k), \tag{2.17b}$$

where $\eta_{il}^k \in (0,1]$ and $\eta_{ij}^k \in (0,1]$ subject to $\sum_{l\in\mathcal{N}_{s,i}^k} \eta_{il}^k + \sum_{j\in\mathcal{U}_i^k} \eta_{ij}^k = 1$.

Recall that another estimate of $\mathbf{x}_{R_i}^k$ is the prior estimate $\bar{\mathbf{x}}_{R_i}^k$. Here, one might be tempted to directly fuse them using (2.1) as

$$\begin{aligned} \mathbf{p}_{R_i}^k &= [(\bar{\mathbf{p}}_{R_i}^k)^{-1} + (\breve{\mathbf{p}}_{R_i}^k)]^{-1}, \\ \hat{\mathbf{x}}_{R_i}^k &= \mathbf{p}_{R_i}^k[(\bar{\mathbf{p}}_{R_i}^k)^{-1}\bar{\mathbf{x}}_{R_i}^k + (\breve{\mathbf{p}}_{R_i}^k)^{-1}\breve{\mathbf{x}}_{R_i}^k], \end{aligned} \tag{2.18}$$

which implicitly assumes that $\bar{\mathbf{x}}_{R_i}^k$ and $\breve{\mathbf{x}}_{R_i}^k$ are uncorrelated. However, this is not the case. For example, when robot i uses robot l's pose estimate to update its own, their estimates become correlated. If we use (2.18) directly, when there exists a chain of updates back to robot l, robot l's pose estimate will be overconfident, since we incorporate the common information twice. In fact, the posterior estimation process becomes the problem of track-to-track fusion under unknown correlations. In order to guarantee the consistency while improving the accuracy, we adopt an ICI based update approach to fuse $\bar{\mathbf{x}}_{R_i}^k$ and $\breve{\mathbf{x}}_{R_i}^k$ in this step. The proposed *Distributed EIF* (DEIF) algorithm for localization is summarized in Tab. 2.1.

Remark 4 *We incorporate robot-to-target measurements $\mathbf{z}_{T_{ij}}^k$ in the localization part. Intuitively, this can result in more accurate estimates of the robots' poses, since the targets can be treated as moving references to the robots. This is one of the advantages resulting*

Table 2.1: DEIF algorithm for localization

Propagation:

$\mathbf{\Phi}_{R_i}^{k-1} = \frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_{R_i}}(\hat{\mathbf{x}}_{R_i}^{k-1}, \mathbf{u}_{R_i}^{k-1})$,

$\mathbf{G}_{R_i}^{k-1} = \frac{\partial \mathbf{f}_i}{\partial \mathbf{w}_{R_i}}(\hat{\mathbf{x}}_{R_i}^{k-1}, \mathbf{u}_{R_i}^{k-1})$,

$\mathbf{Q}_{R_i}^{k-1} = \mathbf{G}_{R_i}^{k-1}\mathbf{E}[\mathbf{w}_{R_i}^{k-1}(\mathbf{w}_{R_i}^{k-1})^\mathsf{T}](\mathbf{G}_{R_i}^{k-1})^\mathsf{T}$,

$\bar{\mathbf{p}}_{R_i}^{k} = \mathbf{\Phi}_{R_i}^{k-1}\mathbf{p}_{R_i}^{k-1}(\mathbf{\Phi}_{R_i}^{k-1})^\mathsf{T} + \mathbf{Q}_{R_i}^{k-1}$,

$\bar{\mathbf{x}}_{R_i}^{k} = \mathbf{f}_i(\hat{\mathbf{x}}_{R_i}^{k-1}, \mathbf{u}_{R_i}^{k-1})$.

Compute the update terms:

Obtain $(\breve{\mathbf{p}}_{R_i}^{k}, \breve{\mathbf{x}}_{R_i}^{k})$ using (2.10)-(2.17).

$\mathbf{\Omega}_{R_i}^{k} = (\bar{\mathbf{p}}_{R_i}^{k})^{-1}, \quad \mathbf{q}_{R_i}^{k} = (\bar{\mathbf{p}}_{R_i}^{k})^{-1}\bar{\mathbf{x}}_{R_i}^{k}$,

$\mathbf{\Gamma}_{R_i}^{k} = (\breve{\mathbf{p}}_{R_i}^{k})^{-1}[\alpha_i^k(\breve{\mathbf{p}}_{R_i}^{k})^{-1} + (1-\alpha_i^k)\mathbf{\Omega}_{R_i}^{k}]^{-1}\mathbf{\Omega}_{R_i}^{k}$,

$\mathbf{K}_{R_i}^{k} = \mathbf{\Omega}_{R_i}^{k} - \alpha_i^k\mathbf{\Gamma}_{R_i}^{k}$,

$\mathbf{L}_{R_i}^{k} = (\breve{\mathbf{p}}_{R_i}^{k})^{-1} - (1-\alpha_i^k)\mathbf{\Gamma}_{R_i}^{k}$.

Update:

$\mathbf{p}_{R_i}^{k} = [\mathbf{\Omega}_{R_i}^{k} + (\breve{\mathbf{p}}_{R_i}^{k})^{-1} - \mathbf{\Gamma}_{R_i}^{k}]^{-1}$

$\hat{\mathbf{x}}_{R_i}^{k} = \mathbf{p}_{R_i}^{k}[\mathbf{K}_{R_i}^{k}(\mathbf{\Omega}_{R_i}^{k})^{-1}\mathbf{q}_{R_i}^{k} + \mathbf{L}_{R_i}^{k}\breve{\mathbf{x}}_{R_i}^{k}]$.

The time-varying weight α_i^k subject to $\alpha_i^k \in [0,1]$ is selected to minimize $\mathrm{Tr}\{\mathbf{p}_{R_i}^{k}\}$.

from jointly estimating the states of both robots and targets. The algorithm in Tab. 2.1 is still applicable in the case of CL without targets involved by simply letting $\mathcal{U}_i^k = \varnothing$ *in* (2.17)*. In this case, we refer the algorithm as CL-DEIF. Compared with the existing works on CL, our approach is a fully distributed solution that is consistent, amenable to general models, and computationally simple while accounting for the possible existence of targets.*

Target Tracking

Recall that when robot i detects target $j \in \mathcal{U}$ at time k, linearization of $\mathbf{z}_{T_{ij}}^k$ at $\bar{\mathbf{x}}_{T_{ij}}^k$ and $\bar{\mathbf{x}}_{R_i}^k$ yields the measurement residual (2.12). By defining $\tilde{\mathbf{v}}_{T_{ij}}^k = \mathbf{H}_{T_{ij}}^k \bar{\mathbf{e}}_{R_i}^k + \mathbf{v}_{T_{ij}}^k$, we get $\bar{\mathbf{z}}_{T_{ij}}^k = \widetilde{\mathbf{H}}_{T_{ij}}^k \bar{\mathbf{e}}_{T_{ij}}^k + \tilde{\mathbf{v}}_{T_{ij}}^k$. The corresponding covariance of $\tilde{\mathbf{v}}_{T_{ij}}^k$ is given by

$$\widetilde{\mathbf{R}}_{T_{ij}}^k = \mathbf{R}_{T_{ij}}^k + \mathbf{H}_{T_{ij}}^k \bar{\mathbf{p}}_{R_i}^k (\mathbf{H}_{T_{ij}}^k)^\mathsf{T}. \tag{2.19}$$

Then, define the relative correction pair $(\tilde{\mathbf{s}}_{T_{ij}}^k, \tilde{\mathbf{y}}_{T_{ij}}^k)$ as

$$\begin{aligned} \tilde{\mathbf{s}}_{T_{ij}}^k &= (\widetilde{\mathbf{H}}_{T_{ij}}^k)^\mathsf{T} (\widetilde{\mathbf{R}}_{T_{ij}}^k)^{-1} \widetilde{\mathbf{H}}_{T_{ij}}^k, \\ \tilde{\mathbf{y}}_{T_{ij}}^k &= (\widetilde{\mathbf{H}}_{T_{ij}}^k)^\mathsf{T} (\widetilde{\mathbf{R}}_{T_{ij}}^k)^{-1} (\bar{\mathbf{z}}_{T_{ij}}^k + \widetilde{\mathbf{H}}_{T_{ij}}^k \bar{\mathbf{x}}_{T_{ij}}^k). \end{aligned} \tag{2.20}$$

Next, all available prior estimation pairs and correction pairs from the inclusive communicating neighborhood, i.e., $(\bar{\mathbf{p}}_{T_{lj}}^k, \bar{\mathbf{x}}_{T_{lj}}^k)$ and $(\tilde{\mathbf{s}}_{T_{lj}}^k, \tilde{\mathbf{y}}_{T_{lj}}^k)$, $\forall l \in \mathcal{J}_{c,i}^k$, are incorporated to compute the posterior estimate $\hat{\mathbf{x}}_{T_{ij}}^k$ with the estimated covariance $\mathbf{p}_{T_{ij}}^k$. It is possible that a certain robot, say robot l, cannot directly detect target j. Then for notation convenience, we let $\mathbf{R}_{T_{lj}}^k = \infty$, which assumes infinite uncertainty about the corresponding measurement $\mathbf{z}_{T_{lj}}^k$. As a result, the received $\tilde{\mathbf{y}}_{T_{lj}}^k = 0$ and $\tilde{\mathbf{s}}_{T_{lj}}^k = 0$.

The first step is to use all available correction pairs to compute an estimate of $\mathbf{x}_{T_j}^k$. Similar to the localization part, due to the correlations between the robot pose estimates, the defined $\tilde{\mathbf{v}}_{T_{ij}}^k$ are correlated for different i. Accordingly, we apply the CI algorithm (2.4) on the relative correction pairs to guarantee consistency with the simplified weight selection strategy (2.5). Then, at time k, by using all available relative correction pairs, we have

$$\breve{\mathbf{p}}_{T_{ij}}^k = \Big(\sum_{l \in \mathcal{J}_{c,i}^k} \tilde{\eta}_{lj}^k \tilde{\mathbf{s}}_{T_{lj}}^k \Big)^{-1}, \quad \breve{\mathbf{x}}_{T_{ij}}^k = \breve{\mathbf{p}}_{T_{ij}}^k \Big(\sum_{l \in \mathcal{J}_{c,i}^k} \tilde{\eta}_{lj}^k \tilde{\mathbf{y}}_{T_{lj}}^k \Big), \tag{2.21}$$

where $\tilde{\eta}_{lj}^k = 0$ if robot l cannot directly detect target j; otherwise $\tilde{\eta}_{lj}^k \in (0, 1]$ is the weight subject to $\sum_{l \in \mathcal{J}_{c,i}^k} \tilde{\eta}_{lj}^k = 1$. The second step is to fuse all available prior estimation pairs. Due to the common process model of target j, the local prior estimates $(\bar{\mathbf{p}}_{T_{lj}}^k, \bar{\mathbf{x}}_{T_{lj}}^k)$, $\forall l \in \mathcal{J}_{c,i}^k$, are highly correlated after propagations. Therefore, the CI algorithm (2.4) is used to guarantee consistency with the simplified weight selection strategy (2.5). We have

$$\mathbf{\Omega}_{T_{ij}}^k = \sum_{l \in \mathcal{J}_{c,i}^k} \pi_{il}^k (\bar{\mathbf{p}}_{T_{lj}}^k)^{-1}, \quad \mathbf{q}_{T_{ij}}^k = \sum_{l \in \mathcal{J}_{c,i}^k} \pi_{il}^k (\bar{\mathbf{p}}_{T_{lj}}^k)^{-1} \bar{\mathbf{x}}_{T_{lj}}^k, \tag{2.22}$$

where $\pi_{il}^k \geq \underline{\pi} > 0$ is the weight subject to $\sum_{l \in \mathcal{J}_{c,i}^k} \pi_{il}^k = 1$ with $\underline{\pi}$ being the uniform lower bound of all the weights at all time steps.

The third step is to fuse $(\breve{\mathbf{p}}_{T_{ij}}^k, \breve{\mathbf{x}}_{T_{ij}}^k)$ with $(\mathbf{\Omega}_{T_{ij}}^k, \mathbf{q}_{T_{ij}}^k)$ to compute the posterior estimate of target j. One might consider that the relative correction pairs and the prior estimation pairs are uncorrelated and directly fuse $(\breve{\mathbf{p}}_{T_{ij}}^k, \breve{\mathbf{x}}_{T_{ij}}^k)$ with $(\mathbf{\Omega}_{T_{ij}}^k, \mathbf{q}_{T_{ij}}^k)$ using (2.1) as

$$\mathbf{p}_{T_{ij}}^k = [\mathbf{\Omega}_{T_{ij}}^k + (\breve{\mathbf{p}}_{T_{ij}}^k)^{-1}]^{-1}, \quad \hat{\mathbf{x}}_{T_{ij}}^k = \mathbf{p}_{T_{ij}}^k [\mathbf{q}_{T_{ij}}^k + (\breve{\mathbf{p}}_{T_{ij}}^k)^{-1} \breve{\mathbf{x}}_{T_{ij}}^k]. \tag{2.23}$$

This is the case when static sensor networks with known positions are employed. However, as target j's state estimate has been used to update the robots' pose estimates in the localization part, its state estimate becomes correlated with the robots' pose estimates. Hence, here we adopt the ICI (2.3) to fuse $(\breve{\mathbf{p}}_{T_{ij}}^k, \breve{\mathbf{x}}_{T_{ij}}^k)$ with $(\mathbf{\Omega}_{T_{ij}}^k, \mathbf{q}_{T_{ij}}^k)$ to avoid information double-counting when the robots' pose estimates are in turn used in the relative correction pairs to compute the posterior estimate of target j. The proposed *Distributed EIF* (DEIF) algorithm for target tracking is summarized in Tab. 2.2.

Table 2.2: DEIF algorithm for target tracking

Propagation:

$\mathbf{\Phi}_{T_{ij}}^{k-1} = \frac{\partial \mathbf{g}_j}{\partial \mathbf{x}_{T_{ij}}}(\hat{\mathbf{x}}_{T_{ij}}^{k-1}), \qquad \mathbf{G}_{T_{ij}}^{k-1} = \frac{\partial \mathbf{g}_j}{\partial \mathbf{w}_{T_j}}(\hat{\mathbf{x}}_{T_{ij}}^{k-1}),$

$\mathbf{Q}_{T_{ij}}^{k-1} = \mathbf{G}_{T_{ij}}^{k-1}\mathbf{E}[\mathbf{w}_{T_j}^{k-1}(\mathbf{w}_{T_j}^{k-1})^\mathsf{T}](\mathbf{G}_{T_{ij}}^{k-1})^\mathsf{T},$

$\bar{\mathbf{p}}_{T_{ij}}^{k} = \mathbf{\Phi}_{T_{ij}}^{k}\mathbf{p}_{T_{ij}}^{k-1}(\mathbf{\Phi}_{T_{ij}}^{k-1})^\mathsf{T} + \mathbf{Q}_{T_{ij}}^{k-1},$

$\bar{\mathbf{x}}_{T_{ij}}^{k} = \mathbf{g}_j(\hat{\mathbf{x}}_{T_{ij}}^{k-1}).$

Compute the update terms:

Obtain $(\breve{\mathbf{p}}_{T_{ij}}^{k}, \breve{\mathbf{x}}_{T_{ij}}^{k})$ and $(\mathbf{\Omega}_{T_{ij}}^{k}, \mathbf{q}_{T_{ij}}^{k})$ using (2.19)-(2.22).

$\mathbf{\Gamma}_{T_{ij}}^{k} = (\breve{\mathbf{p}}_{T_{ij}}^{k})^{-1}[\alpha_{ij}^{k}(\breve{\mathbf{p}}_{T_{ij}}^{k})^{-1} + (1-\alpha_{ij}^{k})\mathbf{\Omega}_{T_{ij}}^{k}]^{-1}\mathbf{\Omega}_{T_{ij}}^{k},$

$\mathbf{K}_{T_{ij}}^{k} = \mathbf{\Omega}_{T_{ij}}^{k} - \alpha_{ij}^{k}\mathbf{\Gamma}_{T_{lj}}^{k},$

$\mathbf{L}_{T_{ij}}^{k} = (\breve{\mathbf{p}}_{T_{ij}}^{k})^{-1} - (1-\alpha_{ij}^{k})\mathbf{\Gamma}_{T_{ij}}^{k}.$

Update:

$\mathbf{p}_{T_{ij}}^{k} = [\mathbf{\Omega}_{T_{ij}}^{k} + (\breve{\mathbf{p}}_{T_{ij}}^{k})^{-1} - \mathbf{\Gamma}_{T_{ij}}^{k}]^{-1}$

$\hat{\mathbf{x}}_{T_{ij}}^{k} = \mathbf{p}_{T_{ij}}^{k}[\mathbf{K}_{T_{ij}}^{k}(\mathbf{\Omega}_{T_{ij}}^{k})^{-1}\mathbf{q}_{T_{ij}}^{k} + \mathbf{L}_{T_{ij}}^{k}\breve{\mathbf{x}}_{T_{ij}}^{k}].$

The time-varying weight α_{ij}^{k} subject to $\alpha_{ij}^{k} \in [0,1]$ is selected to minimize $\mathrm{Tr}\{\mathbf{p}_{T_{ij}}^{k}\}$.

Remark 5 *In Tables 2.2, the prior estimates are weighted averaged over the communicating neighborhood. Therefore, a robot directly sensing target j can either directly or indirectly influence the other robots through the communication topology. Hence, target j's state is cooperatively estimated by each robot even if some robots cannot detect target j at a certain time. In addition, data association is required for multi-target tracking. However, it is out of the scope in this chapter. We assume that each robot knows exactly which measurement belongs to which target.*

2.3.2 Joint Localization and Target Tracking

Based on the previous section, we propose the JLATT (JLATT) algorithm from the DEIF perspective in a mobile robot network, where multiple relative measurements

might take place at one robot and each robot can communicate with its nearby neighbors within the communication range. We refer the algorithm as JLATT-DEIF. It is worth noting that the communication and sensing topologies are subject to change with time as well as the robots not directly sensing the targets.

Initialization: For $i \in \mathcal{V}$, initialize the DEIF estimates: $\mathbf{p}^0_{R_i}, \hat{\mathbf{x}}^0_{R_i}$ and $\mathbf{p}^0_{T_{ij}}, \hat{\mathbf{x}}^0_{T_{ij}}$, $\forall j \in \mathcal{U}$.

Propagation: as in Tables 2.1 and 2.2.

Update:

1. Robot i obtains available relative measurements $\mathbf{z}^k_{R_{il}}$ to the other robots in $\mathcal{N}^k_{s,i}$ and $\mathbf{z}^k_{T_{ij}}$ to the targets. Recall that if target j is out of the sensing region of robot i, $(\bar{\mathbf{R}}^k_{T_{ij}})^{-1} = 0$.

2. Robot i obtains its absolute measurement $\mathbf{z}^k_{R_i}$ if available and otherwise $(\mathbf{R}^k_{R_i})^{-1} = 0$.

3. Receive $\{\bar{\mathbf{p}}^k_{T_{lj}}, \bar{\mathbf{x}}^k_{T_{lj}}, \tilde{\mathbf{s}}^k_{T_{lj}}, \tilde{\mathbf{y}}^k_{T_{lj}}\}$ from robot l, $\forall l \in \mathcal{N}^k_{c,i}$ and $\forall j \in \mathcal{U}$.

4. Compute localization correction pairs $\{\mathbf{s}^k_{R_{il}}, \mathbf{y}^k_{R_{il}}\}$, $\forall l \in \mathcal{N}^k_{s,i}$ as in (2.10)-(2.11), $\{\mathbf{s}^k_{T_{ij}}, \mathbf{y}^k_{T_{ij}}\}$, $\forall j \in \mathcal{U}^k_i$ as in (2.13)-(2.14) and $\{\mathbf{s}^k_{R_{ii}}, \mathbf{y}^k_{R_{ii}}\}$ as in (2.16).

5. Update the pose estimate of robot i as in Tab. 2.1.

6. Compute tracking correction pairs $\{\tilde{\mathbf{s}}^k_{T_{ij}}, \tilde{\mathbf{y}}^k_{T_{ij}}\}$, $\forall j \in \mathcal{U}$ as in (2.19)-(2.20).

7. Update the state estimate of target $j \in \mathcal{U}$ as in Tab. 2.2.

Remark 6 *The state and covariance propagations and updates described in Tables 2.1 and 2.2 allow for a fully distributed JLATT-DEIF algorithm which uses one-hop communication and requires no global parameter.*

2.4 Stability Analysis

In this section, the stability of the proposed algorithm is analyzed in the linearized time-varying systems. By linearizing (2.6), the error propagation equation for robot i is given by

$$\bar{\mathbf{e}}_{R_i}^k = \mathbf{\Phi}_{R_i}^{k-1}\mathbf{e}_{R_i}^{k-1} + \mathbf{G}_{R_i}^{k-1}\mathbf{w}_{R_i}^{k-1}, \tag{2.24}$$

where $\mathbf{\Phi}_{R_i}^{k-1}$ and $\mathbf{G}_{R_i}^{k-1}$ are defined in Tab. 2.1, with the measurement error equations given by (2.9), (2.12), and (2.15). By linearizing (2.7), the error propagation equation for target j is given by

$$\bar{\mathbf{e}}_{T_{ij}}^k = \mathbf{\Phi}_{T_{ij}}^{k-1}\mathbf{e}_{T_{ij}}^{k-1} + \mathbf{G}_{T_{ij}}^{k-1}\mathbf{w}_{T_j}^{k-1}, \tag{2.25}$$

where $\mathbf{\Phi}_{T_{ij}}^{k-1}$ and $\mathbf{G}_{T_{ij}}^{k-1}$ are defined in Tab. 2.2, with the measurement error equation given by (2.12). We refer to (2.24), (2.9), (2.12), and (2.15) as the localization system and (2.25) and (2.12) as the tracking system, respectively. Further, the motion, process and measurement noise covariances are assumed to be time invariant for simplicity (i.e., $\mathbf{Q}_{R_i}^k = \mathbf{Q}_{R_i} > 0$, $\mathbf{Q}_{T_{ij}}^k = \mathbf{Q}_{T_{ij}} > 0$, $\mathbf{R}_{R_{ij}}^k = \mathbf{R}_{R_{ij}} > 0$, $\mathbf{R}_{R_i}^k = \mathbf{R}_{R_i} > 0$, and $\mathbf{R}_{T_{ij}}^k = \mathbf{R}_{T_{ij}} > 0$). Next, we focus on the localization part. We first give the definition of observable pairs.

Definition 7 *The pair $(\mathbf{A}^\tau, \mathbf{C}^\tau)$, where τ is the time index, is observable on $\mathcal{T}_{j_0}^{j_1}$, if and only if the observability grammian*

$$\sum_{\tau=j_0}^{j_1} [\mathbf{A}(\tau, j_0)]^\mathsf{T} (\mathbf{C}^\tau)^\mathsf{T} \mathbf{C}^\tau \mathbf{A}(\tau, j_0) > 0$$

where $\mathbf{A}(\tau, j_0)$ is the transition matrix on $\mathcal{T}_{j_0}^{\tau}$.

In order to evaluate the stability of the algorithm in Tab. 2.1, we make the following assumptions.

Assumption 2.4.1 *There exists a positive integer $\bar{k}$ such that at each time $k \geq \bar{k}$, the following statements hold.*

1. *There exists a nonempty subset $\mathcal{V}^k \subseteq \mathcal{V}$, such that for each robot $i \in \mathcal{V}^k$, the pair $(\mathbf{\Phi}^{\tau}_{R_i}, \mathbf{C}^{\tau}_{R_i})$ is observable on $\mathcal{T}^{k-\bar{k}+\bar{n}}_{k-\bar{k}}$, where $0 < \bar{n} < \bar{k}$.*

2. *For each robot $j \in \mathcal{V} \backslash \mathcal{V}^k$, there exists a directed path from a certain robot $i \in \mathcal{V}^k$ to j in the form of $(i_0, i_1), (i_1, i_2), \ldots, (i_{l-1}, i_l)$, where $i_0 = i$ and $i_l = j$, and l consecutive intervals of the form $\mathcal{T}^{m_1}_{m_0}, \ldots, \mathcal{T}^{m_l}_{m_{l-1}}$, where $m_0 = k - \bar{k} + \bar{n}$ and $m_l = k$, such that $(\mathbf{\Phi}^{\tau}_{R_{i_s}}, \mathbf{H}^{\tau}_{R_{i_s i_{s-1}}})$ is observable on $\mathcal{T}^{m_s}_{m_{s-1}}$, where $s = 1, \ldots, l$.*

Assumption 2.4.2 *For each $k \geq 0$ and robot $i \in \mathcal{V}$, the system matrix $\mathbf{\Phi}^k_{R_i}$ is invertible.*

Assumption 2.4.3 *For each robot $i \in \mathcal{V}$, the initialized estimation pair $\left(\hat{\mathbf{x}}^0_{R_i}, \mathbf{p}^0_{R_i}\right)$ is consistent. That is,*

$$\mathbb{E}\{\mathbf{e}^0_{R_i}(\mathbf{e}^0_{R_i})^{\mathsf{T}}\} \leq \mathbf{p}^0_{R_i}.$$

Remark 8 *As for Assumption 2.4.1, $\mathcal{V}^k$ can be changing over time. For example, $\mathcal{V}^k$ might just contain one robot at times. Further, none of the robots needs to receive absolute measurements on $\mathcal{T}^k_{m_0}$. In other words, for either absolute or relative measurements, only a sparse possibly changing subset of the team needs to have access to and those measurements can be intermittent. Assumption 2.4.2 is automatically satisfied as $\mathbf{\Phi}^k_{R_i}$ is obtained by discretization of a continuous-time system before linearization. Finally, Assumption 2.4.3 can be guaranteed by initializing $\mathbf{p}^0_{R_i}$ with a sufficiently large value.*

We next give the main stability result of the localization part and then prove it step by step.

Theorem 9 *Suppose that Assumptions 2.4.1-2.4.3 hold. Then the pose estimate of each robot is stable under the algorithm in Tab. 2.1. That is, for each robot $i \in \mathcal{V}$, there exists a positive definite matrix $\mathbf{p}_i$ such that*

$$\mathbb{E}\{\mathbf{e}_{R_i}^k (\mathbf{e}_{R_i}^k)^\mathsf{T}\} \leq \mathbf{p}_i$$

for any $k \geq \bar{k}$.

In order to prove the above stability result, we first study the consistency of the estimates.

Lemma 10 *Let Assumption 2.4.3 hold. For each robot $i \in \mathcal{V}$, the estimation pair $\left(\hat{\mathbf{x}}_{R_i}^k, \mathbf{p}_{R_i}^k\right)$ obtained from the proposed DEIF in Tab. 2.1 is consistent, that is,*

$$\mathbb{E}\{\mathbf{e}_{R_i}^k (\mathbf{e}_{R_i}^k)^\mathsf{T}\} \leq \mathbf{p}_{R_i}^k, \ \forall k \geq 0.$$

Proof. The proof is shown by induction. When $k = 0$, Assumption 2.4.3 implies that $\mathbb{E}\{\mathbf{e}_{R_i}^0 (\mathbf{e}_{R_i}^0)^\mathsf{T}\} \leq \mathbf{p}_{R_i}^0$. Then it is assumed that at time $k-1$, $\mathbb{E}\{\mathbf{e}_{R_i}^{k-1} (\mathbf{e}_{R_i}^{k-1})^\mathsf{T}\} \leq \mathbf{p}_{R_i}^{k-1}$. Notice that the propagation error satisfies

$$\bar{\mathbf{e}}_{R_i}^k = \mathbf{\Phi}_{R_i}^{k-1} \mathbf{e}_{R_i}^{k-1} + \mathbf{G}_{R_i}^{k-1} \mathbf{w}_{R_i}^{k-1}.$$

Because $\mathbb{E}\{\bar{\mathbf{e}}_{R_i}^k (\mathbf{w}_{R_i}^k)^\mathsf{T}\} = 0$, it follows that

$$\begin{aligned}\mathbb{E}\{\bar{\mathbf{e}}_{R_i}^k (\bar{\mathbf{e}}_{R_i}^k)^\mathsf{T}\} &= \mathbf{\Phi}_{R_i}^{k-1} \mathbb{E}\{\mathbf{e}_{R_i}^{k-1} (\mathbf{e}_{R_i}^{k-1})^\mathsf{T}\} (\mathbf{\Phi}_{R_i}^{k-1})^\mathsf{T} + \mathbf{Q}_{R_i} \\ &\leq \mathbf{\Phi}_{R_i}^{k-1} \mathbf{p}_{R_i}^{k-1} (\mathbf{\Phi}_{R_i}^{k-1})^\mathsf{T} + \mathbf{Q}_{R_i} = \bar{\mathbf{p}}_{R_i}^k.\end{aligned}$$

Next, as analyzed in Section 2.3.1, by exploiting the consistency property of ICI, the update step in Tab. 2.1 is guaranteed to be consistent. It follows that $\mathbb{E}\{\mathbf{e}_{R_i}^k (\mathbf{e}_{R_i}^k)^\mathsf{T}\} \leq \mathbf{p}_{R_i}^k$. ■

Lemma 10 points out that, in order to prove the boundedness of the actual error covariance, it is sufficient to show that the estimated covariance $\mathbf{p}_{R_i}^k$ is upper bounded by a certain constant matrix.

Lemma 11 *[10] Let $\mathbf{\Phi}$ be a nonsingular matrix. For any $\mathbf{Q} > 0$ and $\breve{\mathbf{p}} > 0$, there exists a parameter $\beta \in (0, 1]$ such that $(\mathbf{\Phi}\mathbf{p}\mathbf{\Phi}^{\mathsf{T}} + \mathbf{Q})^{-1} \geq \beta \mathbf{\Phi}^{-\mathsf{T}}\mathbf{p}^{-1}\mathbf{\Phi}^{-1}$ for any $\mathbf{p} \geq \breve{\mathbf{p}}$.*

Lemma 12 *Suppose that Assumptions 2.4.1-2.4.3 hold. Then for each $i \in \mathcal{V}$, there exists a positive-definite matrix $\mathbf{p}_i$ such that*

$$\mathbf{p}_{R_i}^k \leq \mathbf{p}_i, \qquad \forall k \geq \bar{k}.$$

Proof. To simplify the notation, for certain time τ and τ_0, we define $\tilde{\tau}_{\tau_0} = \tau - \tau_0$. First, we focus on robot $i \in \mathcal{V}^k$. At any time $a \in \mathcal{T}_{m_0}^k$, the inverse of the updated covariance can be written as

$$(\mathbf{p}_{R_i}^a)^{-1} = \mathbf{\Omega}_{R_i}^a + \sum_{l \in \mathcal{N}_{s,i}^a} \eta_{il}^a \mathbf{s}_{R_{il}}^a + \sum_{j \in \mathcal{U}_i^a} \eta_{ij}^a \mathbf{s}_{T_{ij}}^a + \mathbf{s}_{R_{ii}}^a - \mathbf{\Gamma}_{R_i}^a.$$

Invoking Lemma 2, one can get the lower bound

$$\begin{aligned}(\mathbf{p}_{R_i}^a)^{-1} &\geq \alpha_a \mathbf{\Omega}_{R_i}^a + (1 - \alpha_a)\Big(\sum_{l \in \mathcal{N}_{s,i}^a} \eta_{il}^a \mathbf{s}_{R_{il}}^a + \mathbf{s}_{R_{ii}}^a\Big) \\ &= \alpha_a \Psi\{(\mathbf{p}_{R_i}^{a-1})^{-1}\} + (1 - \alpha_a)\Big(\sum_{l \in \mathcal{N}_{s,i}^a} \eta_{il}^a \mathbf{s}_{R_{il}}^a + \mathbf{s}_{R_{ii}}^a\Big)\end{aligned}$$

where $\Psi\{(\mathbf{p}_{R_i}^{a-1})^{-1}\} = [\mathbf{\Phi}_{R_i}^{a-1}\mathbf{p}_{R_i}^{a-1}(\mathbf{\Phi}_{R_i}^{a-1})^{\mathsf{T}} + \mathbf{Q}_{R_i}]^{-1}$ and $\alpha_a \in (0, 1)$. Further, under Assumption 2.4.2, it follows from Lemma 11 that

$$\Psi\{(\mathbf{p}_{R_i}^{a-1})^{-1}\} \geq \beta_a (\mathbf{\Phi}_{R_i}^{a-1})^{-\mathsf{T}} (\mathbf{p}_{R_i}^{a-1})^{-1} (\mathbf{\Phi}_{R_i}^{a-1})^{-1},$$

with $\beta_a \in (0,1]$, for any $\mathbf{p}_{R_i}^{a-1} \geq \mathbb{E}\{\mathbf{e}_{R_i}^{a-1}(\mathbf{e}_{R_i}^{a-1})^{\mathsf{T}}\}$. Then, one can obtain

$$\begin{aligned}(\mathbf{p}_{R_i}^{a})^{-1} \geq & \alpha_a\beta_a(\mathbf{\Phi}_{R_i}^{a-1})^{-\mathsf{T}}(\mathbf{p}_{R_i}^{a-1})^{-1}(\mathbf{\Phi}_{R_i}^{a-1})^{-1} \\ & + (1-\alpha_a)(\sum_{l\in\mathcal{N}_{s,i}^{a}} \eta_{il}^{a}\mathbf{s}_{R_{il}}^{a} + \mathbf{s}_{R_{ii}}^{a}).\end{aligned} \tag{2.26}$$

where we can further write

$$\begin{aligned}(\mathbf{p}_{R_i}^{a-1})^{-1} \geq & \alpha_{a-1}\beta_{a-1}(\mathbf{\Phi}_{R_i}^{a-2})^{-\mathsf{T}}(\mathbf{p}_{R_i}^{a-2})^{-1}(\mathbf{\Phi}_{R_i}^{a-2})^{-1} \\ & + (1-\alpha_{a-1})(\sum_{l\in\mathcal{N}_{s,i}^{a-1}} \eta_{il}^{a-1}\mathbf{s}_{R_{il}}^{a-1} + \mathbf{s}_{R_{ii}}^{a-1}).\end{aligned} \tag{2.27}$$

By recursively substituting (2.27) into (2.26) for $\bar{a} = a - \tilde{k}_{\bar{k}}$ times, one can write

$$(\mathbf{p}_{R_i}^{a})^{-1} \geq \sum_{\tau=0}^{\bar{a}} \check{\alpha}_\tau\check{\beta}_\tau\mathbf{\Phi}_{R_i}^{-\mathsf{T}}(a,\tilde{a}_\tau)\triangle\mathbf{\Phi}_{R_i}^{-1}(a,\tilde{a}_\tau), \tag{2.28}$$

where $\mathbf{\Phi}_{R_i}(a,\tilde{a}_\tau)$ is the transition matrix on $\mathcal{T}_a^{\tilde{a}_\tau}$; $\triangle = \sum_{l\in\mathcal{N}_{s,i}^{\tilde{a}_\tau}} \eta_{il}^{\tilde{a}_\tau}\mathbf{s}_{R_{il}}^{\tilde{a}_\tau} + \mathbf{s}_{R_{ii}}^{\tilde{a}_\tau}$; $\check{\alpha}_\tau = (1-\alpha_{\tilde{a}_\tau})\prod_{j=0}^{\tau-1}\alpha_{\tilde{a}_j}$ and $\check{\beta}_\tau = \prod_{j=0}^{\tau-1}\beta_{\tilde{a}_j}$, for $\tau > 0$; $\check{\alpha}_\tau = (1-\alpha_a)$, and $\check{\beta}_\tau = 1$, for $\tau = 0$. Note that the right hand side of (2.28) can be equivalently written as $\mathbf{\Phi}_{R_i}^{-\mathsf{T}}(a,\tilde{a}_{\bar{a}})\mathbf{\Psi}\mathbf{\Phi}_{R_i}^{-1}(a,\tilde{a}_{\bar{a}})$, where

$$\begin{aligned}\mathbf{\Psi} = & \sum_{\tau=0}^{\bar{a}} \check{\alpha}_\tau\check{\beta}_\tau\mathbf{\Phi}_{R_i}^{\mathsf{T}}(\tilde{a}_\tau,\tilde{a}_{\bar{a}})\triangle\mathbf{\Phi}_{R_i}(\tilde{a}_\tau,\tilde{a}_{\bar{a}}) = \sum_{\tau=0}^{\bar{a}-\bar{n}-1} \check{\alpha}_\tau\check{\beta}_\tau\mathbf{\Phi}_{R_i}^{\mathsf{T}}(\tilde{a}_\tau,\tilde{a}_{\bar{a}})\triangle\mathbf{\Phi}_{R_i}(\tilde{a}_\tau,\tilde{a}_{\bar{a}}) \\ & + \sum_{\tau=\bar{a}-\bar{n}}^{\bar{a}} \check{\alpha}_\tau\check{\beta}_\tau\mathbf{\Phi}_{R_i}^{\mathsf{T}}(\tilde{a}_\tau,\tilde{a}_{\bar{a}})\triangle\mathbf{\Phi}_{R_i}(\tilde{a}_\tau,\tilde{a}_{\bar{a}}).\end{aligned}$$

Invoking part (1) of Assumption 2.4.1, $(\mathbf{\Phi}_{R_i}^{\tau}, \mathbf{C}_{R_i}^{\tau})$ is observable on $\mathcal{T}_{a-\bar{a}}^{a-\bar{a}+\bar{n}}$. We have

$$\sum_{\tau=\bar{a}-\bar{n}}^{\bar{a}} \mathbf{\Phi}_{R_i}^{\mathsf{T}}(\tilde{a}_\tau,\tilde{a}_{\bar{a}})(\mathbf{C}_{R_i}^{\tilde{a}_\tau})^{\mathsf{T}}\mathbf{C}_{R_i}^{\tilde{a}_\tau}\mathbf{\Phi}_{R_i}(\tilde{a}_\tau,\tilde{a}_{\bar{a}}) > 0.$$

Recall that $\mathbf{s}_{R_{ii}}^{\tau} = (\mathbf{C}_{R_i}^{k})^{\mathsf{T}}\mathbf{R}_{R_i}^{-1}\mathbf{C}_{R_i}^{k}$. Then, by noticing that $\check{\alpha}_\tau \in (0,1)$, $\check{\beta}_\tau \in (0,1]$ and $\mathbf{R}_{R_i}^{-1} > 0$, it can be seen that

$$\mathbf{\Psi} \geq \sum_{\tau=\bar{a}-\bar{n}}^{\bar{a}} \check{\alpha}_\tau\check{\beta}_\tau\mathbf{\Phi}_{R_i}^{\mathsf{T}}(\tilde{a}_\tau,\tilde{a}_{\bar{a}})\mathbf{s}_{R_{ii}}^{\tilde{a}_\tau}\mathbf{\Phi}_{R_i}(\tilde{a}_\tau,\tilde{a}_{\bar{a}}) > 0.$$

Hence, we can obtain a positive-definite matrix

$$(\mathbf{p}_i^a)^{-1} = \sum_{\tau=\bar{a}-\bar{n}}^{\bar{a}} \check{\alpha}_\tau \check{\beta}_\tau \mathbf{\Phi}_{R_i}^{-\mathsf{T}}(a, \tilde{a}_\tau) \mathbf{s}_{R_{ii}}^{\tilde{a}_\tau} \mathbf{\Phi}_{R_i}^{-1}(a, \tilde{a}_\tau),$$

such that $\mathbf{p}_{R_i}^a \leq \mathbf{p}_i^a$. Further, let $\bar{\mathbf{p}}_i = \mathbf{\Phi}_{R_i} \mathbf{p}_i (\mathbf{\Phi}_{R_i})^\mathsf{T} + \mathbf{Q}_{R_i}$, where $\mathbf{\Phi}_{R_i} \geq \mathbf{\Phi}_{R_i}^\tau$, $\forall \tau \in \mathcal{T}_{k-\bar{k}}^a$. We have $\bar{\mathbf{p}}_{R_i}^k \leq \bar{\mathbf{p}}_i^a$. As a is arbitrary in $\mathcal{T}_{m_0}^k$, it follows that there exist, respectively, positive matrices $\mathbf{p}_i$ and $\bar{\mathbf{p}}_i$ such that $\mathbf{p}_{R_i}^\tau \leq \mathbf{p}_i$, $\bar{\mathbf{p}}_{R_i}^\tau \leq \bar{\mathbf{p}}_i$ and $\forall \tau \in \mathcal{T}_{m_0}^k$.

Next, consider the robots in $\mathcal{V} \backslash \mathcal{V}^k$. Starting from robot j to which there exists an edge from one robot $i \in \mathcal{V}^k$ on $\mathcal{T}_{m_0}^{m_1}$. Such robot j exists due to part (2) of Assumption 2.4.1. Following a similar process, at any time $b \in \mathcal{T}_{m_1}^k$, after $\bar{b} = b - m_0$ iterations, one can obtain

$$(\mathbf{p}_{R_j}^b)^{-1} \geq \sum_{\tau=\bar{b}-\bar{m}_1}^{\bar{b}} \check{\alpha}_\tau \check{\beta}_\tau \mathbf{\Phi}_{R_j}^{-\mathsf{T}}(b, \tilde{b}_\tau) \Big(\sum_{l \in \mathcal{N}_{s,j}^{\tilde{b}_\tau}} \eta_{jl}^{\tilde{b}_\tau} \mathbf{s}_{R_{jl}}^{\tilde{b}_\tau} \Big) \mathbf{\Phi}_{R_j}^{-1}(b, \tilde{b}_\tau),$$

where $\mathbf{\Phi}_{R_j}(b, \tilde{b}_\tau)$ is the transition matrix on $\mathcal{T}_b^{\tilde{b}_\tau}$; $\bar{m}_1 = m_1 - m_0$. Note that it is possible that $\mathbf{s}_{R_{jj}}^\tau = 0$, $\forall \tau \in \mathcal{T}_{k-\bar{k}}^k$ but $\mathcal{N}_{s,j}^\tau$ contains i on $\mathcal{T}_{m_0}^{m_1}$. Then one can write

$$(\mathbf{p}_{R_j}^b)^{-1} \geq \sum_{\tau=\bar{b}-\bar{m}_1}^{\bar{b}} \check{\alpha}_\tau \check{\beta}_\tau \mathbf{\Phi}_{R_j}^{-\mathsf{T}}(b, \tilde{b}_\tau) \eta_{ji}^{\tilde{b}_\tau} \mathbf{s}_{R_{ji}}^{\tilde{b}_\tau} \mathbf{\Phi}_{R_j}^{-1}(b, \tilde{b}_\tau).$$

Recall from (2.10) and (2.11) that

$$\mathbf{s}_{R_{ji}}^\tau = (\mathbf{H}_{R_{ji}}^\tau)^\mathsf{T} (\bar{\mathbf{R}}_{R_{ji}}^\tau)^{-1} \mathbf{H}_{R_{ji}}^\tau = (\mathbf{H}_{R_{ji}}^\tau)^\mathsf{T} \{ \mathbf{R}_{R_{ji}} + \widetilde{\mathbf{H}}_{R_{ji}}^\tau \bar{\mathbf{p}}_{R_i}^\tau (\widetilde{\mathbf{H}}_{R_{ji}}^\tau)^\mathsf{T} \}^{-1} \mathbf{H}_{R_{ji}}^\tau.$$

It can be immediately seen that the boundedness of $\mathbf{p}_{R_j}^\tau$ is related to $\bar{\mathbf{p}}_{R_i}^\tau$. As $\bar{\mathbf{p}}_{R_i}^\tau \leq \bar{\mathbf{p}}_i$, there exists $\widetilde{\mathbf{H}}_{R_{ji}} \in \{ \widetilde{\mathbf{H}}_{R_{ji}}^\tau \mid \tau \in \mathcal{T}_{m_0}^{m_1} \}$ such that $\bar{\mathbf{R}}_{R_{ji}}^\tau \leq \bar{\mathbf{R}}_{R_{ji}} = \mathbf{R}_{R_{ji}} + \widetilde{\mathbf{H}}_{R_{ji}} \bar{\mathbf{p}}_i (\widetilde{\mathbf{H}}_{R_{ji}})^\mathsf{T}$, $\forall \tau \in \mathcal{T}_{m_0}^{m_1}$. Then, we can write $\mathbf{s}_{R_{ji}}^\tau \geq \underline{\mathbf{s}}_{R_{ji}}^\tau > 0$, where $\underline{\mathbf{s}}_{R_{ji}}^\tau = (\mathbf{H}_{R_{ji}}^\tau)^\mathsf{T} (\bar{\mathbf{R}}_{R_{ji}})^{-1} \mathbf{H}_{R_{ji}}^\tau$, $\forall \tau \in \mathcal{T}_{m_0}^{m_1}$. Invoking part (2) in Assumption 2.4.1, $(\mathbf{\Phi}_{R_j}^\tau, \mathbf{H}_{R_{ji}}^\tau)$ is observable on $\mathcal{T}_{b-\bar{b}}^{b-\bar{b}+\bar{m}_1}$. We have

$$\sum_{\tau=\bar{b}-\bar{m}_1}^{\bar{b}} \mathbf{\Phi}_{R_j}^\mathsf{T}(\tilde{b}_\tau, \tilde{b}_{\bar{b}}) (\mathbf{H}_{R_{ji}}^{\tilde{b}_\tau})^\mathsf{T} \mathbf{H}_{R_{ji}}^{\tilde{b}_\tau} \mathbf{\Phi}_{R_j}(\tilde{b}_\tau, \tilde{b}_{\bar{b}}) > 0.$$

Then, by noticing that $\check{\alpha}_\tau \in (0,1)$, $\check{\beta}_\tau \in (0,1]$, $\eta_{ji}^{\tilde{b}_\tau} \in (0,1]$ and $\bar{\mathbf{R}}_{R_{ji}}^{-1} > 0$, we can obtain a positive-definite matrix

$$(\mathbf{p}_j^b)^{-1} = \sum_{\tau=\bar{b}-\bar{m}_1}^{\bar{b}} \check{\alpha}_\tau \check{\beta}_\tau \eta_{ji}^{\tilde{b}_\tau} \mathbf{\Phi}_{R_j}^{-\mathsf{T}}(b,\tilde{b}_\tau) \underline{\mathbf{s}}_{R_{ji}}^{\tilde{b}_\tau} \mathbf{\Phi}_{R_j}^{-1}(b,\tilde{b}_\tau),$$

such that $\mathbf{p}_{R_j}^b \leq \mathbf{p}_j^b$. Hence, we can claim that there exists a matrix $\mathbf{p}_j$ such that $\mathbf{p}_{R_j}^\tau \leq \mathbf{p}_j$, $\forall \tau \in \mathcal{T}_{m_1}^k$. Also, we can find an upper bound $\bar{\mathbf{p}}_j$ of $\bar{\mathbf{p}}_{R_j}^\tau$, where $\tau \in \mathcal{T}_{m_1}^k$.

Similarly, for another robot $l \in \mathcal{V}\backslash\mathcal{V}^k$ to which there exists an edge from robot j on $\mathcal{T}_{m_1}^{m_2}$. We can obtain a positive-definite matrix $\mathbf{p}_l$ which is associated with $\bar{\mathbf{p}}_j$, such that $\mathbf{p}_{R_l}^\tau \leq \mathbf{p}_l$, $\forall \tau \in \mathcal{T}_{m_2}^k$.

Part (2) of Assumption 2.4.1 says that for each robot in $\mathcal{V}\backslash\mathcal{V}^k$, there exists a directed path from one robot in $\mathcal{V}^k$ to that robot. By applying the above approach orderly along that directed path, it takes $\bar{k}-\bar{n}$ time instants to make the estimated pose covariance of the farthest robot upper bounded, where $\bar{k}-\bar{n}$ is the total length of l consecutive intervals. As this is the case for any $k \geq \bar{k}$, we can conclude the proof. ■

Hence, the statement of Theorem 9 follows directly from Lemmas 10 and 12.

Since the stability analysis is the same for each target, we focus on one target $j \in \mathcal{U}$. Further, as the proof follows a similar approach to that of Theorem 9, only the different parts are shown in detail in the following. We first give the definition of the joint observable set and orderly appearing path as follows.

Definition 13 *Let $\mathcal{V}'$ be a nonempty subset of $\mathcal{V}$. The tracking system* (2.12) *and* (2.25) *of target j is jointly observable to the robots in $\mathcal{V}'$ on $\mathcal{T}_{j_0}^{j_1}$ if and only if the joint observability*

grammian

$$\sum_{l\in\mathcal{V}'}\sum_{\tau=j_0}^{j_1}[\mathbf{\Phi}_{T_{lj}}(\tau,j_0)]^{\mathsf{T}}(\widetilde{\mathbf{H}}^{\tau}_{T_{lj}})^{\mathsf{T}}\widetilde{\mathbf{H}}^{\tau}_{T_{lj}}\mathbf{\Phi}_{T_{lj}}(\tau,j_0)>0,$$

where $\mathbf{\Phi}_{T_{lj}}(\tau,j_0)$ *is the transition matrix on* $\mathcal{T}^{\tau}_{j_0}$.

Definition 14 *[122] Let* $\mathcal{B}=\{e_1,\ldots,e_p\}$, *where* $e_j=(i_{j-1},i_j)$, $\forall j=1,\ldots,p$, *be a direct path in a graph. Then* $\mathcal{B}$ *is an orderly appearing path on* $\mathcal{T}^{\tau_1}_{\tau_0}$, *if there exist* p *time instants* $\tau_{l_1}<\tau_{l_2}<,\ldots,<\tau_{l_p}$ *on* $\mathcal{T}^{\tau_1}_{\tau_0}$ *such that* e_j *is an edge (including the self edge) of that graph at time* τ_{l_i}, *where* $i=1,\ldots,p$.[1]

In order to derive the stability result of the algorithm in Tab. 2.2, the following assumptions and a lemma are needed.

Assumption 2.4.4 *There exists a positive integer* $\bar{l}$, *such that for each robot* $i\in\mathcal{V}$ *at each time* $k\geq\bar{k}+\bar{l}$, *where* $\bar{k}$ *is from Assumption 2.4.1, one can find a nonempty robot subset* $\mathcal{V}^k_i\subseteq\mathcal{V}$ *that satisfies the following statements.*

1. $\mathcal{V}^k_i$ *has joint observability about target* j *on* $\mathcal{T}^{k-\bar{l}+\bar{m}}_{k-\bar{l}}$, *where* $0<\bar{m}<\bar{l}$.
2. *Every robot in* $\mathcal{V}^k_i$ *has an orderly appearing path in the communication graph* G^{τ}_c *to* i *on* $\mathcal{T}^{k}_{k-\bar{l}+\bar{m}}$.

Assumption 2.4.5 *For any* $k\geq 0$, *the system matrix* $\mathbf{\Phi}^k_{T_{lj}}$ *is invertible.*

Assumption 2.4.6 *For each robot* $i\in\mathcal{V}$, *the initialized estimation pair* $\left(\hat{\mathbf{x}}^0_{T_{ij}},\mathbf{p}^0_{T_{ij}}\right)$ *is consistent. That is* $\mathbb{E}\{\mathbf{e}^0_{T_{ij}}(\mathbf{e}^0_{T_{ij}})^{\mathsf{T}}\}\leq\mathbf{p}^0_{T_j}$.

[1] By default (i,i) itself can be an orderly appearing path.

Let $\mathbf{D}^k = [\mathbf{D}^k_{il}]$ be the stochastic adjacently matrix associated with G^k_c at time k, where $\mathbf{D}^k_{il} = \pi^k_{il}$ with $\pi^k_{il} \in (0, 1]$ being the weights from the algorithm in Tab. 2.2 if $l \in \mathcal{J}^k_{c,i}$ and $\mathbf{D}^k_{il} = 0$ otherwise.

Lemma 15 *[122] Given a finite time interval $\mathcal{T}^{j_1}_{j_0}$, let $\mathcal{D}^{j_1}_{j_0} = \mathbf{D}^{j_1}, \ldots, \mathbf{D}^{j_0+1}\mathbf{D}^{j_0}$. Then $\{\mathcal{D}^{j_1}_{j_0}\}_{il} > 0$ if and only if there exists an orderly appearing path from i to l on $\mathcal{T}^{j_1}_{j_0}$.*

The following theorem shows the stability result of the tracking part.

Theorem 16 *Suppose that Assumptions 2.4.1-2.4.6 hold. Then target j's state estimate obtained by each robot is stable under the algorithm in Tab. 2.2. That is, for each $i \in \mathcal{V}$, there exists a positive-definite matrix $\mathbf{p}_{i,j}$ such that,*

$$\mathbb{E}\{\mathbf{e}^k_{T_{ij}}(\mathbf{e}^k_{T_{ij}})^\mathsf{T}\} \leq \mathbf{p}_{i,j}$$

for each $k \geq \bar{k} + \bar{l}$.

Proof. Notice that the consistency of the update step is preserved by ICI as shown in Section 2.3.1. Then, under Assumption 2.4.6, following the same process as in Lemma 10, we can get the consistency result. That is, for each robot $i \in \mathcal{V}$,

$$\mathbb{E}\{\mathbf{e}^\tau_{T_{ij}}(\mathbf{e}^\tau_{T_{ij}})^\mathsf{T}\} \leq \mathbf{p}^\tau_{T_{ij}}, \ \forall \tau \geq 0.$$

As Assumption 2.4.5 holds, for each robot $i \in \mathcal{V}$ at time k, where $k \geq \bar{k} + \bar{l}$, by following a similar approach to that in Lemma 12, we have

$$(\mathbf{p}^k_{T_{ij}})^{-1} \geq \sum_{l \in \mathcal{V}} \alpha_k \beta_k (\mathbf{\Phi}^{k-1}_{T_{lj}})^{-\mathsf{T}} \mathbf{D}^k_{il} (\mathbf{p}^{k-1}_{T_{lj}})^{-1} (\mathbf{\Phi}^{k-1}_{T_{lj}})^{-1} + (1 - \alpha_k) \sum_{l \in \mathcal{J}^k_{c,i}} \tilde{\eta}^k_{lj} \tilde{\mathbf{s}}^k_{T_{lj}},$$

where $\alpha_k \in (0,1)$ and $\beta_k \in (0,1]$. Then, by noticing that $\sum_{l\in\mathcal{J}^k_{c,i}} \tilde{\mathbf{s}}^k_{T_{lj}} \geq \sum_{l\in\mathcal{V}} \mathbf{D}^k_{il}\tilde{\mathbf{s}}^k_{T_{lj}}$, after $\bar{l}$ iterations, one can write

$$(\mathbf{p}^k_{T_{ij}})^{-1} \geq \sum_{\tau=\bar{l}-\bar{m}}^{\bar{l}} \check{\alpha}_\tau\check{\beta}_\tau \sum_{l\in\mathcal{V}} \mathbf{\Phi}^{-\mathsf{T}}_{T_{lj}}(k,\tilde{k}_\tau)\{\mathcal{D}^k_{k-\tau}\}_{il}\tilde{\eta}^{\tilde{k}_\tau}_{lj}\tilde{\mathbf{s}}^{\tilde{k}_\tau}_{T_{lj}}\mathbf{\Phi}^{-1}_{T_{lj}}(k,\tilde{k}_\tau). \tag{2.29}$$

where $\check{\alpha}_\tau = (1-\alpha_{\tilde{a}_\tau})\prod_{j=0}^{\tau-1}\alpha_{\tilde{a}_j}$ and $\check{\beta}_\tau = \prod_{j=0}^{\tau-1}\beta_{\tilde{a}_j}$, for $\tau > 0$; $\check{\alpha}_\tau = 1-\alpha_a$ and $\check{\beta}_\tau = 1$, for $\tau = 0$. Equation (2.29) can be further written as

$$(\mathbf{p}^k_{T_{ij}})^{-1} \geq \sum_{l\in\mathcal{V}} \mathbf{\Phi}^{-\mathsf{T}}_{T_{lj}}(k,\tilde{k}_{\bar{k}})[\sum_{\tau=\bar{l}-\bar{m}}^{\bar{l}} \check{\alpha}_\tau\check{\beta}_\tau\mathbf{\Phi}^{\mathsf{T}}_{T_{lj}}(\tilde{k}_\tau,\tilde{k}_{\bar{k}})\{\mathcal{D}^k_{\tilde{k}_\tau}\}_{il}\tilde{\eta}^{\tilde{k}_\tau}_{lj}\tilde{\mathbf{s}}^{\tilde{k}_\tau}_{T_{lj}}\mathbf{\Phi}_{T_{lj}}(\tilde{k}_\tau,\tilde{k}_{\bar{k}})]\mathbf{\Phi}^{-1}_{T_{lj}}(k,\tilde{k}_{\bar{k}}).$$

As part (2) of Assumption 2.4.4 is satisfied, it follows from Lemma 15 that $\{\mathcal{D}^k_{\tilde{k}_\tau}\}_{il} > 0$, $\forall l \in \mathcal{V}^k_i$ and $\forall \tau \in \mathcal{T}^{\bar{l}}_{\bar{l}-\bar{m}}$. Then, from part (1) of Assumption 2.4.4, we can claim that

$$\sum_{l\in\mathcal{V}}\sum_{\tau=\bar{l}-\bar{m}}^{\bar{l}} \mathbf{\Phi}^{\mathsf{T}}_{T_{lj}}(\tilde{k}_\tau,\tilde{k}_{\bar{k}})\{\mathcal{D}^k_{\tilde{k}_\tau}\}_{il}(\widetilde{\mathbf{H}}^{\tilde{k}_\tau}_{T_{lj}})^{\mathsf{T}}\widetilde{\mathbf{H}}^{\tilde{k}_\tau}_{T_{lj}}\mathbf{\Phi}_{T_{lj}}(\tilde{k}_\tau,\tilde{k}_{\bar{k}})$$

is positive definite. Recall from (2.13) and (2.14) that

$$\tilde{\mathbf{s}}^\tau_{T_{lj}} = (\widetilde{\mathbf{H}}^\tau_{T_{lj}})^{\mathsf{T}}(\bar{\mathbf{R}}^\tau_{T_{lj}})^{-1}\widetilde{\mathbf{H}}^\tau_{T_{lj}} = (\widetilde{\mathbf{H}}^\tau_{T_{lj}})^{\mathsf{T}}\{\mathbf{R}_{T_{lj}} + \mathbf{H}^\tau_{T_{lj}}\bar{\mathbf{p}}^\tau_{R_l}(\mathbf{H}^\tau_{T_{lj}})^{\mathsf{T}}\}^{-1}\widetilde{\mathbf{H}}^\tau_{T_{lj}}.$$

Invoking Theorem 9, for any $\tau \geq \bar{k}$, there exists an upper bound $\bar{\mathbf{p}}_l$ for $\bar{\mathbf{p}}^\tau_{R_l}$, $\forall l \in \mathcal{V}$. Then, there exists $\mathbf{H}_{T_{lj}} \in \{\mathbf{H}^\tau_{T_{lj}} \mid \tau \in \mathcal{T}^{k-\bar{l}+\bar{m}}_{k-\bar{l}}\}$ such that $\bar{\mathbf{R}}^\tau_{T_{lj}} \leq \bar{\mathbf{R}}_{T_{lj}} = \mathbf{R}_{T_{lj}} + \mathbf{H}_{T_{lj}}\bar{\mathbf{p}}_l(\mathbf{H}_{T_{lj}})^{\mathsf{T}}$, $\forall\tau \in \mathcal{T}^{k-\bar{l}+\bar{m}}_{k-\bar{l}}$. Then, we can write $\tilde{\mathbf{s}}^\tau_{T_{lj}} \geq \underline{\tilde{\mathbf{s}}}^\tau_{T_{lj}}$, where $\underline{\tilde{\mathbf{s}}}^\tau_{T_{lj}} = (\widetilde{\mathbf{H}}^\tau_{T_{lj}})^{\mathsf{T}}\bar{\mathbf{R}}^{-1}_{T_{lj}}\widetilde{\mathbf{H}}^\tau_{T_{lj}}$, $\forall\tau \in \mathcal{T}^{k-\bar{l}+\bar{m}}_{k-\bar{l}}$. Let $\mathcal{B}^\tau \subseteq \mathcal{V}$ be the set of blind robots at time τ. On $\mathcal{T}^{k-\bar{l}+\bar{m}}_{k-\bar{l}}$, as $\mathcal{V}^\tau_i$ is nonempty, $\mathcal{V}\backslash\mathcal{B}^\tau$ (the set of robots directly detects target j) is nonempty. For any $l \in \mathcal{V}\backslash\mathcal{B}^\tau$, we have $\tilde{\eta}^\tau_{lj} \in (0,1]$ and $\mathbf{R}^{-1}_{T_{lj}} > 0$, $\forall\tau \in \mathcal{T}^{k-\bar{l}+\bar{m}}_{k-\bar{l}}$. Further, by noticing that $\check{\alpha}_\tau \in (0,1)$, $\check{\beta}_\tau \in (0,1]$ and $\mathcal{V}^\tau_i \subseteq \mathcal{V}\backslash\mathcal{B}^\tau$, we can obtain a positive-definite matrix $(\mathbf{p}_{i,j})^{-1}$ as

$$\sum_{l\in\mathcal{V}}\sum_{\tau=\bar{l}-\bar{m}}^{\bar{l}} \check{\alpha}_\tau\check{\beta}_\tau\tilde{\eta}^{\tilde{k}_\tau}_{lj}\mathbf{\Phi}^{-\mathsf{T}}_{T_{lj}}(k,\tilde{k}_\tau)\{\mathcal{D}^k_{\tilde{k}_\tau}\}_{il}\underline{\tilde{\mathbf{s}}}^{\tilde{k}_\tau}_{T_{lj}}\mathbf{\Phi}^{-1}_{T_{lj}}(k,\tilde{k}_\tau),$$

such that $\mathbf{p}_{T_{ij}}^k \leq \mathbf{p}_{i,j}$.

Hence, the proof can be concluded by noticing that in the above proof, i is arbitrary chosen from $\mathcal{V}$ for each time $k \geq \bar{k} + \bar{l}$. ■

2.5 Simulations

In this section, the performance of the proposed JLATT-DEIF algorithm is tested via a series of Monte Carlo simulations. We consider the scenario, where a team of $M = 4$ robots randomly move on a surface and track multiple targets with $N = 2$. While any type of motion and process model is applicable for the proposed algorithm, we adopt the unicycle model for both robots and targets, to be consistent with the ensuing experimental case. The robot pose $\mathbf{x}_{R_i}^k$ is described with the position $[x_{R_i}^k, y_{R_i}^k]$ and the orientation $\phi_{R_i}^k$ in the global frame. Then the motion model is expressed as

$$\begin{aligned} x_{R_i}^k &= x_{R_i}^{k-1} + v_{R_i}^{k-1}\delta t \cos(\phi_{R_i}^{k-1}) \\ y_{R_i}^k &= y_{R_i}^{k-1} + v_{R_i}^{k-1}\delta t \sin(\phi_{R_i}^{k-1}) \\ \phi_{R_i}^k &= \phi_{R_i}^{k-1} + \omega_{R_i}^{k-1}\delta t, \end{aligned} \tag{2.30}$$

where δt is the length of the sampling time interval, and v_{R_i}, ω_{R_i} represent, respectively, the linear and rotational velocity of robot i. These velocities are measured by the odometries equipped on the drive wheels and the associated noise is assumed to be white Gaussian with the standard deviation of 0.02 m for position and 2° for orientation. Each robot moves with a constant linear velocity of $v_{R_i} = 0.5$ m/s, and the rotational velocity ω_{R_i} is chosen from a uniform distribution over $[-\frac{\pi}{6}, \frac{\pi}{6}]$ rad/s. Similarly, the targets move in the same area following the process of (2.30) with $v_{T_i} = 0.6$ m/s and $\omega_{T_i} \in [-\frac{\pi}{5}, \frac{\pi}{5}]$ rad/s, subject to the

same noise of the robot odometry measurements. The state $\mathbf{x}_{T_i}$ to be estimated contains the position and orientation of target i also in the global frame.

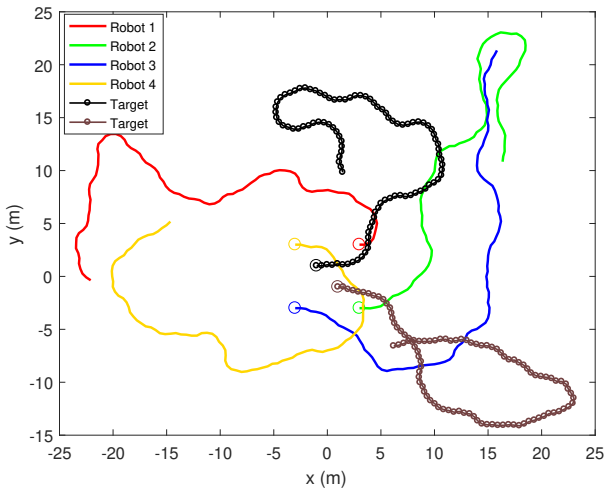

Figure 2.1: A team of four robots move randomly and track two targets. Their starting positions are marked by cicles.

In the test, the robots and the targets start from different locations and follow the real trajectories depicted in Fig. 2.1. Although our approach can deal with generic measurement models. To be consistent with the experiment, each robot records the relative distance and bearing to other robots and targets within its sensing region. In order to fully validate our algorithm, in the simulation case no absolute measurements exist and the relative measurements are generated randomly in time, while in the following experimental case, the landmarks provide absolute measurements and the relative measurements obtained are related to the pose of each robot. For robot i, if robot j is detected, the relative

measurement is given by

$$\mathbf{z}_{R_{ij}}^k = \begin{bmatrix} \sqrt{(x_{R_j}^k - x_{R_i}^k)^2 + (y_{R_j}^k - y_{R_i}^k)^2} \\ \text{atan2}((y_{R_j}^k - y_{R_i}^k), (x_{R_j}^k - x_{R_i}^k)) - \phi_{R_i}^k \end{bmatrix} + \mathbf{v}_{R_{ij}}^k,$$

where $\mathbf{v}_{R_{ij}}$ is a zero-mean white Gaussian noise. The distance noise is set to be 3% of the actual value and the standard deviation of the bearing noise equals to 3°. The same model is used for the robot-to-target measurement $\mathbf{z}_{T_{ij}}$.

Consider a general case in which each robot performs relative measurements to the other robots with the probability of 20%, while the probability of detecting the targets is 40%. We consider a weak communication link with the failure probability of 30% between each pair. Since the absolute measurement is not accessible, we assume that each robot knows its initial global pose. The initial estimates of the targets' states obtained by each robot are set to $\hat{\mathbf{x}}_{T_{ij}}^0 \sim \mathcal{N}(\mathbf{x}_{T_j}(0), \mathbf{p}_{T_{ij}}^0)$, where $\mathbf{x}_{T_j}(0)$ is the initial true state of the target, and the initial covariance $\mathbf{p}_{T_{ij}}^0 = \mathbf{I}_3$, for $j = 1, 2$. We run 50 Monte Carlo simulations and compare the following four cases under the same setup.

- Dead reckoning (DR): No relative measurements exist. The robots propagate their estimates by integrating the measured velocities. Target tracking is not considered here, since good knowledge of the robots' poses is a prerequisite for tracking.

- CL-DEIF: To show the strength of jointly estimating the states of robots and targets, we purposely neglect the existence of targets and perform CL using the novel DEIF in Tab. 2.1 without incorporating robot-to-target measurements.

- JLATT-DEIF: Based on the algorithms of Tab. 2.1 and 2.2, we achieve localization and target tracking simultaneously.

- CEKF: To the author's knowledge, none of the existing works can address the same problem in a fully distributed way. We hence use CEKF as the benchmark. The centralized state vector contains all the robots' and targets' states. Whenever a relative measurement occurs, the EKF-based update invokes.

We employ the root mean square error (RMSE) to quantify the accuracy. Figures 2.2 show the average RMSE over 50 Monte Carlo runs in positions and orientations for the four robots. As expected, without relative measurements, the estimation errors of DR increase quickly as time goes on. When relative measurements take place in the other three cases, due to the collected information regarding the relative motion to the other robots and targets, the pose uncertainties are significantly reduced. It is evident that JLATT-DEIF results in better accuracy for the estimates of both robot positions and orientations, compared with CL-DEIF in which robot-to-target measurements are ignored. Figures 2.3 depict the position and orientation RMSE for the state estimates of two targets obtained by four robots using JLATT-DEIF and the benchmark CEKF. It becomes clear that four robots can track the targets with performance close to CEKF through communicating only with one-hop neighbors. In comparison to CEKF which achieves the best accuracy, the errors of JLATT-DEIF are slightly larger in both localization and tracking. This is due to the fact that each robot only uses the information from itself and one-hop communicating neighbors. However, it allows for a fully distributed implementation with less computational and communication cost while preserving consistency.

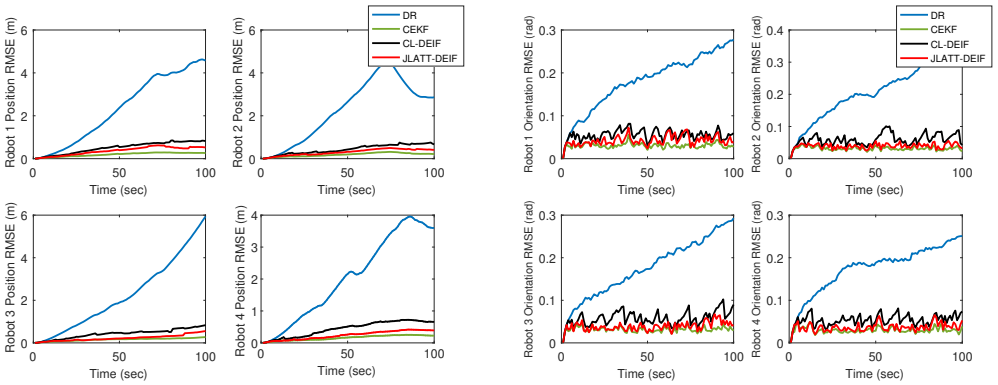

Figure 2.2: Position (left) and Orientation (right) RMSE for four robots averaged over 50 Monte Carlo runs.

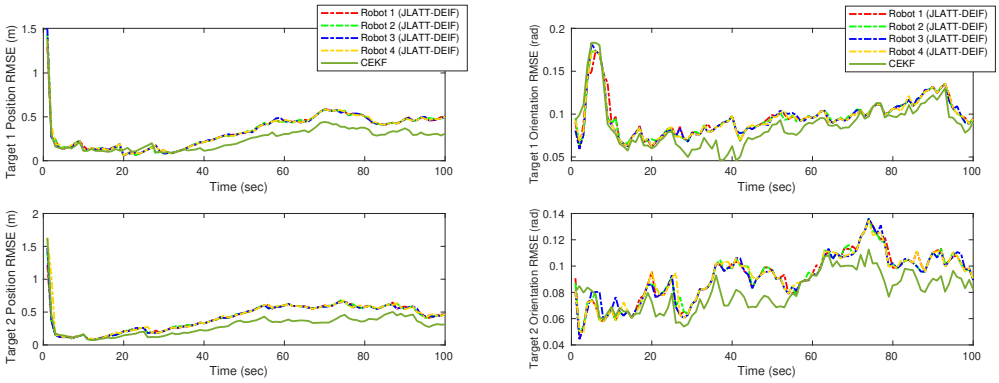

Figure 2.3: Position (left) and Orientation (right) RMSE for two targets averaged over 50 Monte Carlo runs.

To illustrate the consistency issue considered in Section 2.3, we show how the algorithm would perform if the update steps in Tab. 2.1 and Tab. 2.2 are replaced with, respectively, (2.18) and (2.23). The resulting algorithm is denoted as the *inconsistent JLATT-DEIF* (iJLATT-DEIF) and is then compared with the JLATT-DEIF. The normalized estimation error squared (NEES)[8] is used to evaluate the filter consistency. Specifically, if a filer is consistent, it is expected that the average NEES over all Monte Carlo runs for both robots' and targets' states will be close to 3 (i.e., should be close to the dimension of

the state errors). A larger NEES value indicates inconsistency. Fig. 2.4 shows the average NEES for the estimates of one robot and one target. We note that the average NEES of the JLATT-DEIF is close to that of the benchmark CEKF as well as the ideal value 3 in both the localization and tracking parts. While the average NEES of the iJLATT-DEIF in these two parts is gradually increasing over time, indicating that the estimates become inconsistent quickly.

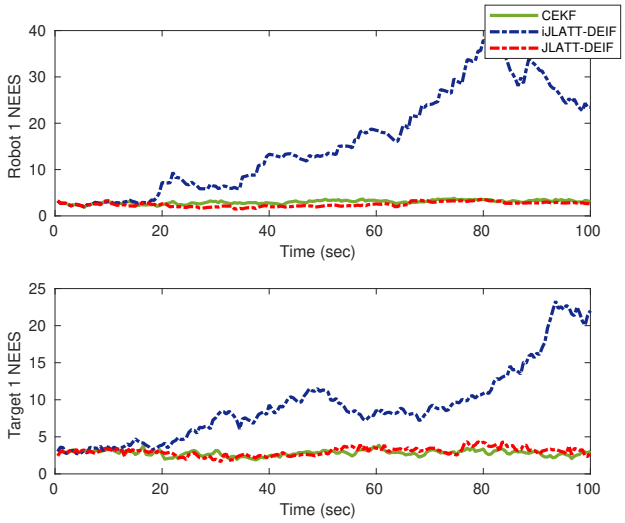

Figure 2.4: Average NEES for Robot 1 and Target 1 (obtained by Robot 1) averaged over 50 Monte Carlo runs.

2.6 Experiments

We further evaluate the performance of our approach on the publically available *UTIAS multi-robot cooperative localization and mapping dataset* [67], where a fleet of five ground robots move in an indoor area of 15 m $\times$ 8 m with 15 static landmarks. Each robot is equipped with a monocular camera with the field of view (FOV) of about 60 degrees. The

robot makes range and bearing measurements when another robot or landmark is inside its FOV. In the meanwhile, a Vicon system is used to monitor the robots' poses and the positions of the landmarks, serving as the groundtruth in the global frame. Note that the original intention of the dataset is not for target tracking. In order to test our approach, we treat one of the robots as the target whose exteroceptive measurements are dropped and the other four form a robot network. We sample the logged data at 50 Hz. In the dataset, there are numerous occlusions between the robots and the target assigned for our purpose, which does not allow the recovering of the target trajectory. To test the target tracking scenario, the groundtruth is used to synthesize robot-to-target measurements with the accuracy of 1% of the actual value for position and 1° for bearing. Note that the synthesized data is only incorporated in the tracking part. Tab. 2.3 gives an overview on the number of actual measurements (including odometry data, and relative and absolute measurements obtained by cameras), and the synthesized robot-to-target measurements for each robot within the first 30000 time instants. The values in parentheses are the actual robot-to-target measurements.

Table 2.3: Overview of how many measurements are used.

_	Actual Measurement		Synthesized Measurement
	Odometry	Camera	(robot-to-target)
Robot 1	30000	2184 (124)	297
Robot 2	30000	2424 (136)	272
Robot 3	30000	2874 (153)	259
Robot 4	30000	3530 (155)	267

The initial estimates for the robot poses are obtained by adding (or subtracting) 0.5 m offset to (or from) the true positions and 5° to (or from) the true orientation, rather

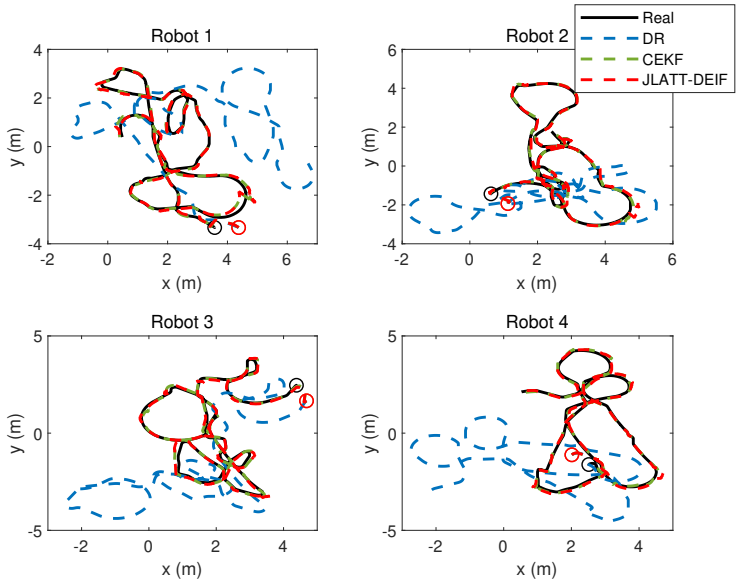

Figure 2.5: Trajectories of four robots. In these lines, the black solid lines correspond to the real value, the blue dashed lines to DR, the green dashed lines to CEKF, and the red dashed lines to JLATT-DEIF. The initial true and estimated positions are marked by circles with the corresponding colors. Circles of DR, CEKF and JLATT-DEIF are overlapped for each robot.

than the true poses in the preceding simulation. For each robot, the target state estimate is initialized at $\hat{\mathbf{x}}^0_{R_i,1} \sim \mathcal{N}(\mathbf{x}_{T_1}(0), \mathbf{p}^0_{R_i,1})$, where $\mathbf{p}^0_{R_i,1} = 0.5\mathbf{I}_3$. As in the preceding simulation test, we compare the performance of our approach JLATT-DEIF with the benchmark CEKF and the DR. Fig. 2.5 depicts the real and estimated trajectories for the robots and Fig. 2.6 shows the results of the target estimates obtained by four robots over the first 30000 time instants. It becomes clear that each robot's estimate of its own position (respectively, the target's position) well tracks the real trajectory of its own (respectively, the target) without knowing the initial true pose. Further, as shown in Figs. 2.7 and 2.8, each robot can also well estimate the real orientations of itself and the target. This is due to the existence of the landmarks which provide the robots with intermittent absolute information in addition

to the relative measurements. Further, the proposed JLATT-DEIF performs comparably to CEKF and the groundtruth.

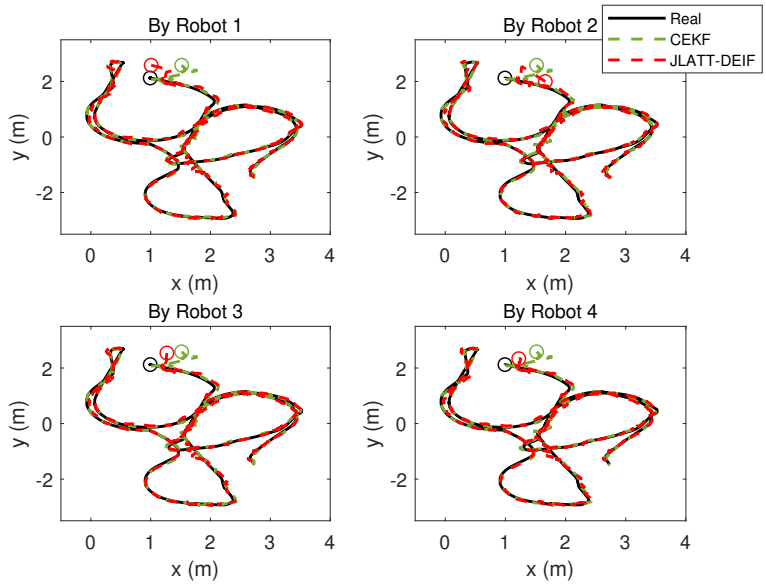

Figure 2.6: Trajectories of the target obtained by four robots. In these lines, the black solid lines correspond to the real value, the green dashed lines to CEKF, and the red dashed lines to JLATT-DEIF. The initial true and estimated positions are marked by circles with the corresponding colors.

Fig. 2.9 presents the recorded absolute value of x errors for one of the robots and the 3σ bound for these errors over a time interval of 10000 instants. The top plot shows the result for iJLATT-DEIF. There are some time intervals in which the absolute x error is outside the 3σ bound, a clear indication that the iJLATT-DEIF estimate is overconfident, which may cause the estimate to diverge. Unlike that, in the bottom one the resulting absolute x error of JLATT-DEIF is well enveloped by the 3σ bound, which agrees with the previous simulation result that JLATT-DEIF is consistent in the localization part. Fig. 2.10 illustrates the comparative result along x direction for one target. Note that robot-to-

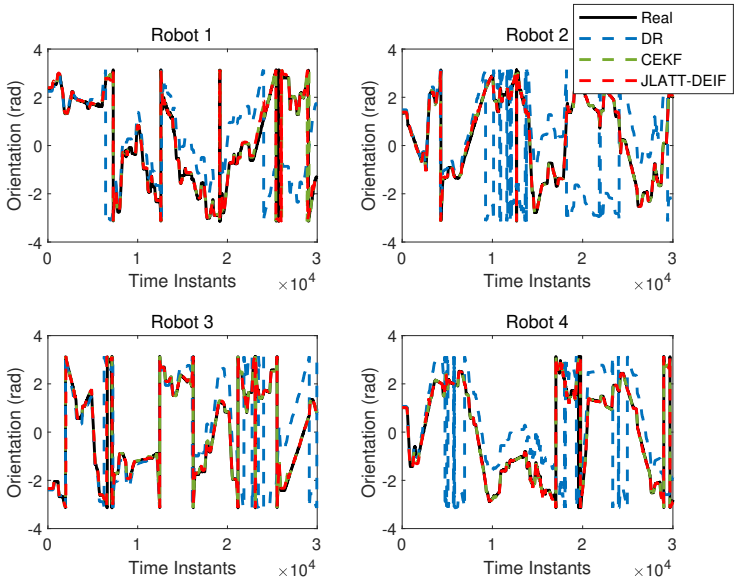

Figure 2.7: Orientations for four robots. In these lines, the black solid lines correspond to the real value, the blue dashed lines to DR, the green dashed lines to CEKF, and the red dashed lines to JLATT-DEIF.

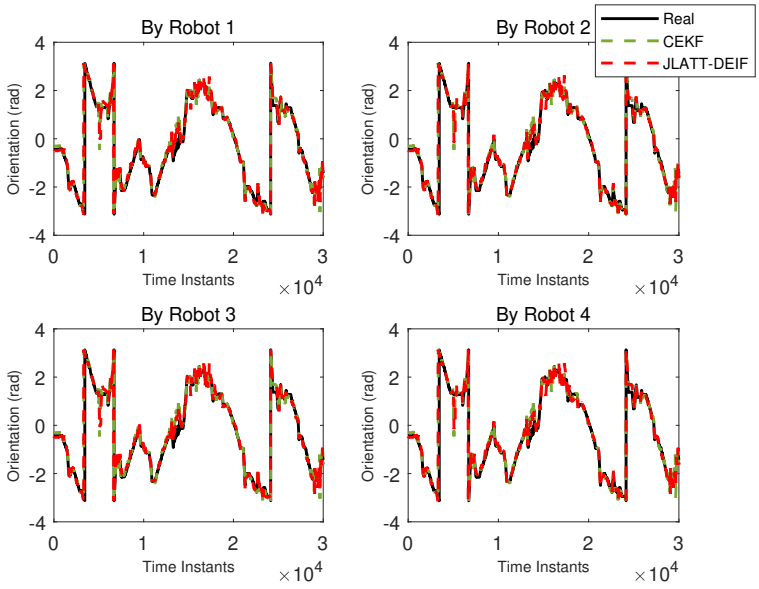

Figure 2.8: Orientations for the target obtained by four robots. In these lines, the black solid lines correspond to the real value, the green dashed lines to CEKF, and the red dashed lines to JLATT-DEIF.

target measurements have been incorporated in the localization part. Hence, directly using (2.23) leads to an inconsistent estimate as shown in the top plot. While as shown in the bottom one, JLATT-DEIF also computes consistent estimates in the tracking part.

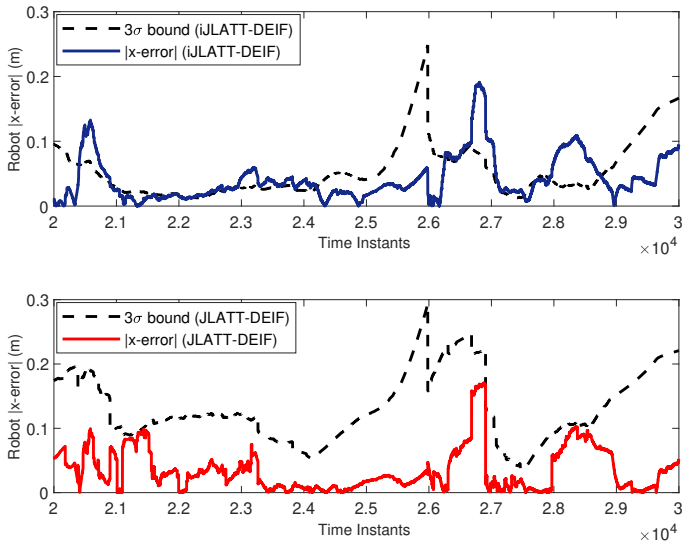

Figure 2.9: Position error in x-direction for one of the robots by using iJLATT-DEIF (top) and JLATT-DEIF (bottom). The solid lines correspond to the absolute value of x errors and the dashed lines to the 3σ bounds

2.7 Conclusions

In this chapter, we have introduced a fully distributed algorithm for the problem of JLATT when both the sensor network and the targets are mobile. Each robot maintains only the latest estimates of its own pose and the states of the targets. The proposed algorithm only requires one communication iteration with the nearby neighbors at one time instant. Further, our approach supports generic robot motion, target process and measurement models, changing communication topologies and dynamic blind robots. These properties

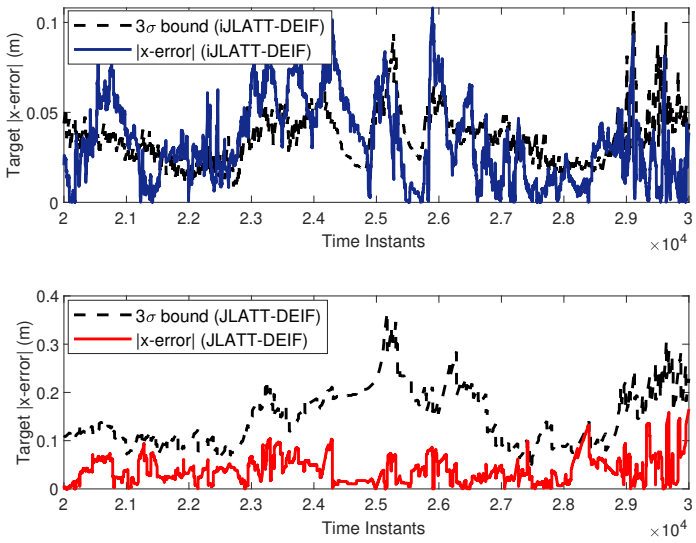

Figure 2.10: Position error in x-direction for the target by using iJLATT-DEIF (top) and JLATT-DEIF (bottom). The solid lines correspond to the absolute value of x errors and the dashed lines to the 3σ bounds

ensure that our approach is applicable in a wide range of multi-robot scenarios. We have also theoretically justified that the proposed estimates are consistent and stable with the errors being bounded in the linearized case. The effectiveness of our approach has been validated by using Monte Carlo simulations and the real-world dataset in different scenarios.

Chapter 3

Distributed 3-D Target State Estimation

3.1 Introduction and Related Works

State estimation in sensor networks has a wide range of applications such as target tracking, environmental monitoring and surveillance. It is assumed that a state of interest is evolving according to noisy dynamics, and each agent may or may not get measurements that are related to the state. The objective is to obtain an accurate estimator of this state on every agent. In conventional centralized algorithms, all the agents send their measurements to a fusion center that runs a centralized Kalman filter (CKF) to get an optimal estimator. This estimator is then sent back to every agent. This approach requires expensive communication and computational resources. Moreover, it has the potential for the failure on the fusion center. In contrast, distributed approaches that have the

advantages of effectiveness, scalability and robustness have drawn more attentions in the research community. In 3-D environments, quaternions have been introduced to express orientations due to its unambiguity and computational efficiency. Furthermore, compared to the Euler angle expression, quaternions avoid singularity when calculating rotations [59]. However, quaternions are not valid vector quantities, which makes the existing distributed Kalman filter (DKF) algorithms not suitable for the quaternion-based 3-D motion tracking in sensor networks.

Due to the aptitude for distributed computing, most of the existing DKF algorithms are derived from the *information filter* (IF) (information form of the Kalman filter) which propagates and updates, instead of the state estimate and the covariance, the information pair that contains the information matrix and the information vector The consensus algorithm as a tool of distributed averaging has been exploited in DSE. Three kind of consensus filters are proposed in [14] where the consensus-on-information algorithm performs the consensus on the prior information pair and the consensus-on-measurements algorithm performs the consensus on the measurements. These two algorithms are then combined to provide a hybrid consensus filter. Ref. [10] develops a Kullback–Leibler average consensus filter where the local measurement is first used to update the local prior information pairs and then the consensus is exploited on the resulting posterior information pairs. Some other consensus filtering algorithms derived from the IF can be found in [57, 128, 11, 41].

Apart from the consensus-based algorithms that require several communication iterations for each measurement, a more efficient kind of DKF is based on the covariance intersection (CI) algorithm presented in [54, 53]. The CI algorithm is proposed to obtain

an improved and consistent estimator from the fusion of multiple estimators with unknown correlations by using a convex combination of the local information pairs. The weights are chosen to minimize the trace or determinant of the fused covariance. Refs. [3] and [42] let each agent compute an estimator by using its own measurements independently and then fuse the resulting posterior information pairs among the neighborhood with CI to obtain an improved estimator. Only the posterior information pairs are transmitted. Another typical CI-based approach is proposed in [45, 123, 122] where the prior information pairs are first fused with CI and then the resulting prior estimator at every agent is further updated with all the measurements among the neighborhood. Here, the prior information pairs and the local measurements are transmitted. Some other CI-based DKF can be found in [16, 126].

Both the IF-derived consensus filters and the CI-based DKF algorithms need to compute the information vector. However, we cannot calculate the information vector for a quaternion due to the mismatching dimension between a quaternion and the corresponding covariance [119]. We further explore the existing DKF without the need of computing the information vector. The Kalman consensus filter (KCF) in [97] and the Generalized KCF (GKCF) in [58] perform an average consensus on the prior estimates in the update step. KCF use equal weights, which causes large estimation errors with blind agents. This issue is avoided by using GKCF that weights the prior estimates by their covariances. Ref. [22] presents the diffusion Kalman filter where each agent first updates the local estimates using its own and the neighbors' measurements, and then computes a convex combination of the resulting estimates. Nevertheless, quaternions are not in the vector space. Then, the arithmetic mean (average consensus) or a convex combination computed is no longer a valid

quaternion and has no physical meaning, which renders the approaches in [97, 58, 22] not applicable.

Indirect Kalman filter has been proposed in [119, 116] to address the problem of *single* robot 3-D localization where the quaternion is used to represent the orientation. But how to fuse the information especially the quaternions from other sensors in a sensor network has not been addressed. From the above observations, it is clear that the existing DKF algorithms are not applicable for the quaternion-based 3-D tracking, which limits their applications in many real-world scenarios where the target exhibits 3-D motion.

3.2 Preliminaries

3.2.1 Quaternion

A quaternion consists of a vector and scalar portion as

$$\bar{q} = q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k} + q_4.$$

For notation simplicity, $\bar{q}$ can be further written as a four-dimensional column matrix given by

$$\bar{q} = \begin{bmatrix} \mathbf{q} \\ q_4 \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 \end{bmatrix}^{\mathsf{T}}. \tag{3.1}$$

Orientation is represented as a unit quaternion [18] which satisfies $|\bar{q}| = \sqrt{|\mathbf{q}|^2 + q_4^2} = 1$. A rotation can be represented by a unit quaternion

$$\bar{q} = \begin{bmatrix} \mathbf{q} \\ q_4 \end{bmatrix} = \begin{bmatrix} \mathbf{m} \cdot \sin(\theta/2) \\ \cos(\theta/2) \end{bmatrix}, \tag{3.2}$$

where $\mathbf{m}$ is the unit vector defining the rotation axis and θ is the angle of rotation. A rotation can also be described by a rotation matrix $\mathbf{R}$ which is related to the unit quaternion $\bar{q}$ by

$$\mathbf{R} = \left(2q_4^2 - 1\right)\mathbf{I}_3 - 2q_4\lfloor \mathbf{q} \times \rfloor + 2\mathbf{q}\mathbf{q}^\mathsf{T}$$

where $\lfloor \cdot \times \rfloor$ denotes the skew symmetric matrix. Moreover, $\bar{q}$ and $-\bar{q}$ describe the same rotation [114].

The error vector and its covariance are usually expressed in terms of the arithmetic difference between the true and estimated values. However, using this representation for a quaternion would make the corresponding covariance singular [66]. Instead, the quaternion error $\delta\bar{q}$ is represented as a rotation between the estimated and true quaternion as $\bar{q} = \hat{\bar{q}} \otimes \delta\bar{q}$, where $\otimes$ denotes the quaternion multiplication. When representing the uncertainty of a quaternion error, a minimal representation of the 3-dimension vector is required [63]. Since the rotation associated with $\delta\bar{q}$ can be assumed to be very small, the mapping between $\delta\bar{q}$ and the minimal representation, the rotation angle error $\delta\boldsymbol{\theta} \in \mathcal{R}^3$, is obtained from (3.2) with small angle approximation as

$$\delta\bar{q} = \begin{bmatrix} \delta\mathbf{q} \\ \delta q_4 \end{bmatrix} = \begin{bmatrix} \mathbf{m} \cdot \sin(\delta\theta/2) \\ \cos(\delta\theta/2) \end{bmatrix} \approx \begin{bmatrix} \frac{1}{2}\delta\boldsymbol{\theta} \\ 1 \end{bmatrix},$$

where $\delta\boldsymbol{\theta} = \mathbf{m}\delta\theta$. Then, the uncertainty of $\delta\bar{q}$ is represented as the covariance of $\delta\boldsymbol{\theta}$. It is evident that for a quaternion estimate $\hat{\bar{q}}$, we cannot compute its information vector, as the dimension for the covariance of $\hat{\bar{q}}$ is 3×3 but $\hat{\bar{q}}$ is 4×1.

3.2.2 Notation and Definitions

$\mathbf{I}_{m\times n}$ ($\mathbf{0}_{m\times n}$) is the identity (zero) matrix of size $m \times n$. If $m = n$, for simplicity, we use $\mathbf{I}_m$ ($\mathbf{0}_m$) to denote the square identity (zero) matrix. We denote G, T and C_i, respectively, as the global frame, the target's body frame and the ith camera frame. ${}^T_G\bar{q}$, the target's orientation, describes the rotation from G to T. ${}^T_G\mathbf{R}$ is the rotation matrix associated with ${}^T_G\bar{q}$. ${}^G\mathbf{p}_T$, the target's global position, denotes the position of T in G. For vector quantities, the error $\delta\mathbf{x}$ is defined as the standard additive error $\delta\mathbf{x} = \mathbf{x} - \hat{\mathbf{x}}$. For a vector $\mathbf{x} = [x\ y\ z]^\mathsf{T}$, the projection function is defined as $\Pi(\mathbf{x}) = \frac{1}{z}[x\ y]^\mathsf{T}$ whose state Jacobian

$$\mathbf{H}_p(x) = \frac{1}{z}\begin{bmatrix} 1 & 0 & -\frac{x}{z} \\ 0 & 1 & -\frac{y}{z} \end{bmatrix}.$$

We define a directed communication graph $\mathcal{G}^k = (\mathcal{V}, \mathcal{E}^k)$, where $\mathcal{V}$ is the agent set and $\mathcal{E}^k$ is the edge set defined as $\mathcal{E}^k \subseteq \mathcal{V} \times \mathcal{V}$. $\mathcal{E}^k$ stands for the communication links between agents at timestep 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 agent i can receive information from agent j, then agent j is a communicating neighbor of agent i. The communicating neighbor set of agent i at timestep k can be defined as $\mathcal{N}_i^k = \{i|(l, i) \in \mathcal{E}^k, l \in \mathcal{V}\}$. Note that $i \in \mathcal{N}_i^k$.

3.3 3-D Distributed State Estimation Algorithm

3.3.1 Problem Formulation

Consider a network of agents, where each agent has the ability to communicate with its neighbors and sense the target with limited sensing region. The target is moving in a 3-D environment. Without loss of generality, we represent the 3-D motion of the target with the following state,

$$\mathbf{x} = \begin{bmatrix} {}^{T}_{G}\bar{q} \\ \\ \mathbf{x}_v \end{bmatrix} = \begin{bmatrix} {}^{T}_{G}\bar{q}^{\mathsf{T}} & {}^{G}\mathbf{p}_T{}^{\mathsf{T}} & {}^{G}\mathbf{v}_T{}^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}}, \tag{3.3}$$

where $\mathbf{x}$ includes the target's 6-DoF pose, ${}^{T}_{G}\bar{q}$ and ${}^{G}\mathbf{p}_T$ in addition to the target's global linear velocity ${}^{G}\mathbf{v}_T$; $\mathbf{x}_v$ contains all the vector quantities in $\mathbf{x}$. Consider the following nonlinear motion model as the dynamics of the target object:

$$\mathbf{x}^k = \mathbf{f}(\mathbf{x}^{k-1}, \mathbf{n}^{k-1}), \tag{3.4}$$

where $\mathbf{x}^k$ is the target's state at timestep k, $\mathbf{n}$ is the zero-mean white Gaussian noise with covariance $\mathbf{O}$. The local measurement $\mathbf{z}_i^k$ obtained by each agent i, $i \in \mathcal{V}$, is given by the following general nonlinear model:

$$\mathbf{z}_i^k = \mathbf{h}_i(\mathbf{x}^k, \mathbf{w}_i^k), \tag{3.5}$$

where $\mathbf{w}_i$ is the local measurement noise assumed to be zero-mean white Gaussian with covariance $\mathbf{R}_i$. We further suppose that the measurement and target process noises are mutually uncorrelated. The objective is to compute an accurate estimate of the target's state $\mathbf{x}$ on every agent by only using the information from itself and the one-hop communicating neighbors.

3.3.2 Proposed Distributed Kalman Filter

Suppose that at timestep k, each agent maintains a prior estimator $(\bar{\mathbf{x}}_i^k, \bar{\mathbf{p}}_i^k)$ after propagation. Now, agent i aims to update its local estimator $(\bar{\mathbf{x}}_i^k, \bar{\mathbf{p}}_i^k)$ by using its local information and the information from its one-hop communicating neighbors. The first step is to fuse all prior estimation pairs among the neighborhood, i.e., $(\bar{\mathbf{x}}_j^k, \bar{\mathbf{p}}_j^k)$, $\forall j \in \mathcal{N}_i^k$. Recall that we cannot directly compute the information vector of a quaternion and then use the consensus or CI algorithms to fuse the prior estimation pairs. Instead, we first *weighted synchronize* the prior estimation pairs to reduce its uncertainty. The weight π_j satisfies $\pi_j \in [0, 1]$ and $\sum_{j\in\mathcal{N}_i} \pi_j = 1$, which makes sure that we do not overuse the information among the neighborhood. For the estimates of the vector quantities $\mathbf{x}_v$ in $\mathbf{x}$, we compute

$$\check{\mathbf{x}}_{v_i}^k = \sum_{j\in\mathcal{N}_i^k} \pi_j^k \bar{\mathbf{x}}_{v_j}^k. \tag{3.6}$$

Note that for the quaternions, our objective is to average the orientations described by the quaternions, not the average of the quaternion. Simply taking the same form of $\mathbf{x}_{v_i}$ cannot even get a valid quaternion (e.g., change the sign of a quaternion should not change the described orientation). Here, we employ the method in [78] which provides a closed form solution of the averaged quaternion ${}^{T_i}_{G}\check{\bar{q}}$ by the following maximization procedure

$${}^{T_{i,k}}_{G}\check{\bar{q}} = \arg\max_{\bar{q}\in\mathbf{S}^3} \bar{q}^{\mathsf{T}}\mathbf{M}\bar{q}, \quad \mathbf{M} = \sum_{j\in\mathcal{N}_i^k} \pi_j^k ({}^{T_{j,k}}_{G}\bar{\bar{q}})^{\mathsf{T}} {}^{T_{j,k}}_{G}\bar{\bar{q}}, \tag{3.7}$$

where ${}^{T_{j,k}}_{G}\bar{\bar{q}}$ is agent j's prior estimate of ${}^{T_k}_{G}\bar{q}$; $\mathbf{S}^3$ denotes the unit 3-sphere. Solving (3.7) in fact gives a quaternion that minimizes the weighted sum of the orientation errors.

We define a compatible symbol $\boxtimes$ for computing the weighted average and then we obtain

$$\check{\mathbf{x}}_i^k = \begin{bmatrix} {}^{T_{i,k}}_{G}\check{q} \\ \\ \check{\mathbf{x}}_{v_i}^k \end{bmatrix} = \sum_{j \in \mathcal{N}_i^k} \pi_j^k \boxtimes \bar{\mathbf{x}}_j^k.$$

As for the synchronized covariance, we can directly compute $\check{\mathbf{p}}_i^k = \sum_{j \in \mathcal{N}_i^k} \pi_j^k \bar{\mathbf{p}}_j^k$, since the quaternion error is represented by the error of the rotational angle that is a vector quantity. The weight π_j^k is chosen to minimize the determinant or the trace of $\check{\mathbf{p}}_i^k$.

For the sake of computational simplicity, we use the simplified algorithm in [95] to calculate π_j^k as

$$\pi_j^k = \frac{1/\mathrm{Tr}(\bar{\mathbf{p}}_j)}{\sum_{j \in \mathcal{N}_i^k} 1/\mathrm{Tr}(\bar{\mathbf{p}}_j)},$$

where $\mathrm{Tr}(\cdot)$ computes the trace of a matrix. Clearly, more weights will be given to the prior estimation pairs with small covariances.

The second step is to fuse the intermediate estimation pair $(\check{\mathbf{x}}_i^k, \check{\mathbf{p}}_i^k)$ with all the local measurements $\mathbf{z}_j^k$, $\forall j \in \mathcal{N}_i^k$. If agent j cannot sense the target directly, we assume infinite uncertainties in $\mathbf{z}_j^k$, that is, $\mathbf{R}_j^k = \infty$. After linearization of $\mathbf{z}_i^k$ about the current estimated state, we compute

$$\mathbf{s}_i^k = (\mathbf{H}_i^k)^{\mathsf{T}} (\mathbf{R}_i^k)^{-1} \mathbf{H}_i^k, \quad \mathbf{y}_i^k = (\mathbf{H}_i^k)^{\mathsf{T}} (\mathbf{R}_i^k)^{-1} \tilde{\mathbf{z}}_i^k, \tag{3.8}$$

where $\tilde{\mathbf{z}}_i = \mathbf{z}_i - \mathbf{h}_i(\bar{\mathbf{x}}_i)$ and $\mathbf{H}_i = \frac{\partial \mathbf{h}_i}{\partial \mathbf{x}_i}(\bar{\mathbf{x}}_i)$. Then, we obtain the updated covariance $\mathbf{p}_i^k$ and the state correction $\delta \mathbf{x}_i^k$ according to

$$\mathbf{p}_i^k = \left[\left(\check{\mathbf{p}}_i^k \right)^{-1} + \sum_{j \in \mathcal{N}_i^k} \mathbf{s}_j^k \right]^{-1}, \quad \delta \mathbf{x}_i^k = \begin{bmatrix} \delta \boldsymbol{\theta}_i^k \\ \\ \delta \mathbf{x}_{v_i}^k \end{bmatrix} = \mathbf{p}_i^k \sum_{j \in \mathcal{N}_i^k} \mathbf{y}_j^k, \tag{3.9}$$

where $\delta\boldsymbol{\theta}_i$ is the orientation correction while $\delta\mathbf{x}_{v_i}$ is the corrections of the vector quantities. Next, we update $\check{\mathbf{x}}_i$ by using $\delta\mathbf{x}_i$.

For the vector quantities $\check{\mathbf{x}}_{v_i}$ in $\check{\mathbf{x}}_i$, we have $\hat{\mathbf{x}}_{v_i}^k = \check{\mathbf{x}}_{v_i}^k + \delta\mathbf{x}_{v_i}^k$. We update the quaternion ${}_G^{T_{i,k}}\check{q}$ according to

$$ {}_G^{T_{i,k}}\hat{\check{q}} = {}_G^{T_{i,k}}\check{q} \otimes \delta\bar{q}_i \tag{3.10} $$

where $\delta\bar{q}_i$ represents a rotation that is supposed to be a unit quaternion. Recall that $\delta\bar{q}_i$ is approximately equal to $[\frac{1}{2}\delta\theta_i^{\mathsf{T}}\ 1]^{\mathsf{T}}$, which is however not a unit quaternion. To obtain a unit quaternion, we let $\delta\bar{q}_i = \frac{1}{\sqrt{1+\frac{1}{4}\delta\boldsymbol{\theta}_i^{\mathsf{T}}\delta\boldsymbol{\theta}_i}} \begin{bmatrix} \frac{1}{2}\delta\boldsymbol{\theta}_i \\ 1 \end{bmatrix}$. We define a compatible symbol $\boxplus$ for updating $\check{\mathbf{x}}_i^k$. Then we have

$$ \hat{\mathbf{x}}_i^k = \begin{bmatrix} {}_G^{T_{i,k}}\hat{\check{q}} \\ \hat{\mathbf{x}}_{v_i}^k \end{bmatrix} = \check{\mathbf{x}}_i^k \boxplus \delta\mathbf{x}_i^k. \tag{3.11} $$

By adding the standard propagation step, the proposed 3-D DKF algorithm is summarised in Algorithm I.

3.4 Simulations

In this section, we apply the proposed 3-D DKF to address the DSE problem over a camera network where 10 cameras are employed to track a drone executing 3-D motion (see Fig. 3.1). Each camera has a limited field of view. The status of which cameras are directly sensing the target over the tracking period is shown in Fig. 3.2. Clearly, all of the cameras could turn into blind cameras for long time periods. Moreover, each camera's intrinsic parameters are known via prior calibration [30]. We perform extensive Monte-Carlo

Table 3.1: Algorithm I: 3-D DKF Algorithm Implemented by Agent i at Timestep k.

Propagation:

$$\mathbf{\Phi}_i^{k-1} = \tfrac{\partial \mathbf{f}}{\partial \mathbf{x}_i}(\hat{\mathbf{x}}_i^{k-1}), \qquad \mathbf{G}_i^{k-1} = \tfrac{\partial \mathbf{f}}{\partial \mathbf{n}}(\hat{\mathbf{x}}_i^{k-1}),$$

$$\mathbf{Q}_i^{k-1} = \mathbf{G}_i^{k-1}\mathbf{O}^{k-1}(\mathbf{G}_i^{k-1})^{\mathsf{T}},$$

$$\bar{\mathbf{p}}_i^k = \mathbf{\Phi}_i^{k-1}\mathbf{p}_i^{k-1}(\mathbf{\Phi}_i^{k-1})^{\mathsf{T}} + \mathbf{Q}_i^{k-1},$$

$$\bar{\mathbf{x}}_i^k = \mathbf{f}(\hat{\mathbf{x}}_i^{k-1}).$$

Update:

(1) compute the update terms $\mathbf{s}_i^k$, $\mathbf{y}_i^k$;
(2) receive $\mathbf{s}_j^k$, $\mathbf{y}_j^k$, $\bar{\mathbf{x}}_j^k$, $\bar{\mathbf{p}}_j^k$ from agent j, $\forall j \in \mathcal{N}_i^k$;
(3) update $\bar{\mathbf{x}}_i^k$, $\mathbf{p}_i^k$ according to

$$\mathbf{p}_i^k = \left[\left(\sum_{j\in\mathcal{N}_i^k} \pi_j^k \bar{\mathbf{p}}_j^k\right)^{-1} + \sum_{j\in\mathcal{N}_i^k} \mathbf{s}_j^k\right]^{-1}$$

$$\hat{\mathbf{x}}_i^k = \left(\sum_{j\in\mathcal{N}_i^k} \pi_j^k \boxtimes \bar{\mathbf{x}}_j^k\right) \boxplus \left(\mathbf{p}_i^k \sum_{j\in\mathcal{N}_i^k} \mathbf{y}_j^k\right)$$

The time-varying weight π_j^k subject to $\pi_j^k \in [0,1]$ is selected to minimize $\mathrm{Tr}\{\mathbf{p}_i^k\}$.

simulations to validate the effectiveness of the proposed algorithm.

3.4.1 State Vector and Models

As vision algorithms can yield many features on the target, like [27] we represent the 3-D rigid body target as the point cloud constructed by the tracked corner features. One of these features is chosen as the representative feature where the target's state is defined while all the other features are the non-representative features that can provide additional observations. These non-representative features' positions are also unknown. We include the non-representative features' relative position in the target's body frame in our estimation state to provide reobservation constraints. Therefore, the target state (3.3)

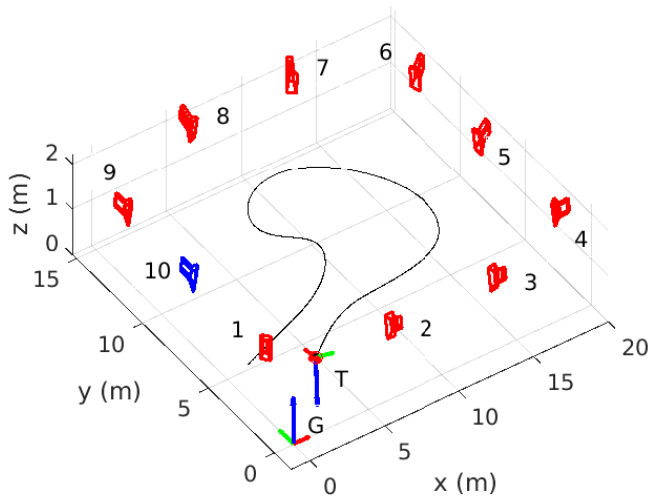

Figure 3.1: 3-D moving object tracking over camera networks. G and T are respectively, the global frame and the target's body frame. The Blue camera denotes the camera currently sensing the target directly while the red ones are the blind cameras. The 3-D trajectory followed by the target is the black line.

is extended to

$$\mathbf{x} = \begin{bmatrix} {}^{T}_{G}\bar{q} \\ \mathbf{x}_v \end{bmatrix} = \begin{bmatrix} {}^{T}_{G}\bar{q}^{\mathsf{T}} & {}^{G}\mathbf{p}_T{}^{\mathsf{T}} & {}^{G}\mathbf{v}_T{}^{\mathsf{T}} & {}^{T}\mathbf{p}_f{}^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}}, \tag{3.12}$$

$${}^{T}\mathbf{p}_f = \begin{bmatrix} {}^{T}\mathbf{p}_{f_1} & \cdots & {}^{T}\mathbf{p}_{f_n} \end{bmatrix}^{\mathsf{T}},$$

where ${}^{T}\mathbf{p}_f$ contains n non-representative features' relative positions in T.

At timestep k, suppose that the target moves according to the following dynamics [119]

$${}^{T_k}_{G}\dot{\bar{q}} = \frac{1}{2}\boldsymbol{\omega}^k \otimes {}^{T_k}_{G}\bar{q}, \quad {}^{G}\dot{\mathbf{p}}_{T_k} = {}^{G}\mathbf{v}_{T_k}, \quad {}^{G}\dot{\mathbf{v}}_{T_k} = {}^{T_k}_{G}\mathbf{R}^{\mathsf{T}}\mathbf{a}^k, \tag{3.13}$$

where $\boldsymbol{\omega}$ and $\mathbf{a}$ are the actual local angular velocity and linear acceleration. The corresponding noisy angular velocity and linear acceleration are given as $\boldsymbol{\omega}_m = \boldsymbol{\omega} + \mathbf{n}_\omega$ and $\mathbf{a}_m = \mathbf{a} + \mathbf{n}_a$, where $\mathbf{n}_\omega$ and $\mathbf{n}_a$ are zero-mean white Gaussian noise. $\boldsymbol{\omega}_m$ and $\mathbf{a}_m$ are known

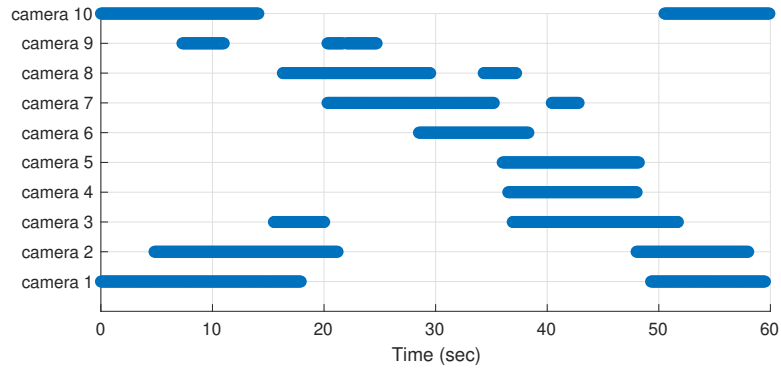

Figure 3.2: Status of cameras directly sensing the target. The bold blue lines indicate the time intervals when the cameras can directly sense the target.

to each agent. After linearizing (3.13), the corresponding error state obtained by camera i evolves according to

$$\begin{bmatrix} {}^{G}\dot{\delta\boldsymbol{\theta}}_{T_{i,k}} \\ {}^{G}\dot{\delta\mathbf{p}}_{T_{i,k}} \\ {}^{G}\dot{\delta\mathbf{v}}_{T_{i,k}} \end{bmatrix} = \mathbf{F}_i^k \begin{bmatrix} {}^{G}\delta\boldsymbol{\theta}_{T_{i,k}} \\ {}^{G}\delta\mathbf{p}_{T_{i,k}} \\ {}^{G}\delta\mathbf{v}_{T_{i,k}} \end{bmatrix} + \mathbf{L}_i^k \mathbf{n}^k \tag{3.14}$$

where $\mathbf{n} = \begin{bmatrix} \mathbf{n}_\omega^\mathsf{T} & \mathbf{n}_a^\mathsf{T} \end{bmatrix}^\mathsf{T}$ with the covariance $\mathbf{O}$,

$$\mathbf{F}_i = \begin{bmatrix} -\lfloor \boldsymbol{\omega}_{m\times} \rfloor & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \\ -\lfloor {}^{T_i}_{G}\bar{\mathbf{R}}^\mathsf{T} \mathbf{a}_{m\times} \rfloor & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix}, \ \mathbf{L}_i = \begin{bmatrix} -\mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & -{}^{T_i}_{G}\bar{\mathbf{R}}^\mathsf{T} \end{bmatrix}.$$

Then, we discretize (3.14) and obtain the first-order approximation. By noting that ${}^{T}\mathbf{p}_f$ does not evolve over time as we assume a rigid-body target, we obtain the discrete-time transition matrix and the noise covariance

$$\boldsymbol{\Phi}_i = \begin{bmatrix} \mathbf{F}_i \delta t + \mathbf{I}_9 & \mathbf{0}_{9\times 3n} \\ \mathbf{0}_{3n\times 9} & \mathbf{I}_{3n} \end{bmatrix}, \quad \mathbf{Q}_i = \begin{bmatrix} \mathbf{L}_i \mathbf{O} \mathbf{L}_i^\mathsf{T} \delta t & \mathbf{0}_{9\times 3n} \\ \mathbf{0}_{3n\times 9} & \mathbf{0}_{3n} \end{bmatrix},$$

where δt is the sampling time. With $\mathbf{\Phi}_i$ and $\mathbf{Q}_i$, we can perform the propagation step in Algorithm I.

As the target explores the environment, the target features are captured by the cameras. Each camera i is assumed to be static with the global pose $({}^{C_i}_{G}\bar{q}, {}^{G}\mathbf{p}_{C_i})$. At timestep k, the measurements of the representative features take the form

$$\mathbf{z}^k_{T_i} = \Pi({}^{C_i}\mathbf{p}_{T_k}) + \mathbf{w}^k_i, \tag{3.15}$$

$${}^{C_i}\mathbf{p}_{T_k} = {}^{C_i}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{T_k} - {}^{G}\mathbf{p}_{C_i}\right), \tag{3.16}$$

where ${}^{C_i}\mathbf{p}_T$ denotes the target's position in the ith camera frame; $\mathbf{w}_i$ is the zero-mean white Gaussian noise with covariance $\mathbf{R}_i$. By linearization of (3.15) and (3.16), we obtain the state Jacobian

$$\mathbf{H}_i = \mathbf{H}_p({}^{C_i}\bar{\mathbf{p}}_T){}^{C_i}_{G}\mathbf{R}\begin{bmatrix}\mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_{3\times 3(n+1)}\end{bmatrix}.$$

For a non-representative feature ${}^{T}\mathbf{p}_{f_1}$ (for notation simplicity, consider the first feature in ${}^{T}\mathbf{p}_f$), then (3.16) is replaced with

$${}^{C_i}\mathbf{p}_{T_k} = {}^{C_i}_{G}\mathbf{R}\left({}^{T_k}_{G}\mathbf{R}^{\mathsf{T}}{}^{T}\mathbf{p}_{f_1} + {}^{G}\mathbf{p}_{T_k} - {}^{G}\mathbf{p}_{C_i}\right). \tag{3.17}$$

Note that (3.17) puts constraints not only on the target's position ${}^{G}\mathbf{p}_{T_k}$ as (3.16) does, but also on the relative position ${}^{T}\mathbf{p}_{f_1}$ and the rotation matrix ${}^{T_k}_{G}\mathbf{R}$ associated with the target's orientation. By linearization of (3.15) and (3.17), we obtain the state Jacobian

$$\mathbf{H}_i = \mathbf{H}_p({}^{C_i}\bar{\mathbf{p}}_T){}^{C_i}_{G}\mathbf{R}[-\lfloor {}^{T_i}_{G}\bar{\mathbf{R}}^{\mathsf{T}}{}^{T}\bar{\mathbf{p}}_{f_1\times}\rfloor \quad \mathbf{I}_3 \quad \mathbf{0}_3 {}^{T_k}_{G}\bar{\mathbf{R}}^{\mathsf{T}} \quad \mathbf{0}_{3\times 3(n-1)}].$$

With $\mathbf{H}_i$, we can perform the update step in Algorithm I.

3.4.2 Results

The target is moving following a pre-designed 3-D trajectory. Further, the non-representative features are generated around the target's body frame. Each camera has the resolution of $[752, 480]$ and its maximum sensing distance is purposely set to 5 m. The linear acceleration and angular velocity noise are 0.4 m/s^2 and 0.03 rad/s, while the camera measurements are corrupted by 1 pixel noise. Then we perform 50 Monte-Carlo simulations and the results are quantified by the root mean squared error (RMSE).

To show the benefits of cooperative tracking, we assume that each camera can communicate with the other cameras with certain percentages. For example, 40% means that each camera can communicate with another camera with the probability of 40%. Hence, each camera's communicating neighbors are randomly chosen at every timestep and the communication graph is time varying. We compare the results of the proposed distributed algorithm (3-D DKF) against the one obtained by the benchmark (CKF) where all the cameras can communicate with the fusion center perfectly. Fig. 3.3 shows the averaged position RMSE (PRMSE) and the orientation RMSE (ORMSE) results for the CKF and the 3-D DKF over all trials and all cameras. It becomes clear that as the communication percentage increases, the estimation errors of the 3-D DKF reduce in both the positions and orientations. In particular, the performance of the 3-D DKF with 60% communication percentage is comparable to the CKF's performance.

Further, to show the performance of individual cameras, Tab.3.2 provides the averaged RMSE results with different communication percentages for the first four cameras (cams 1, 2, 3 and 4) over all trials and all timesteps. Obviously, none of the cameras can

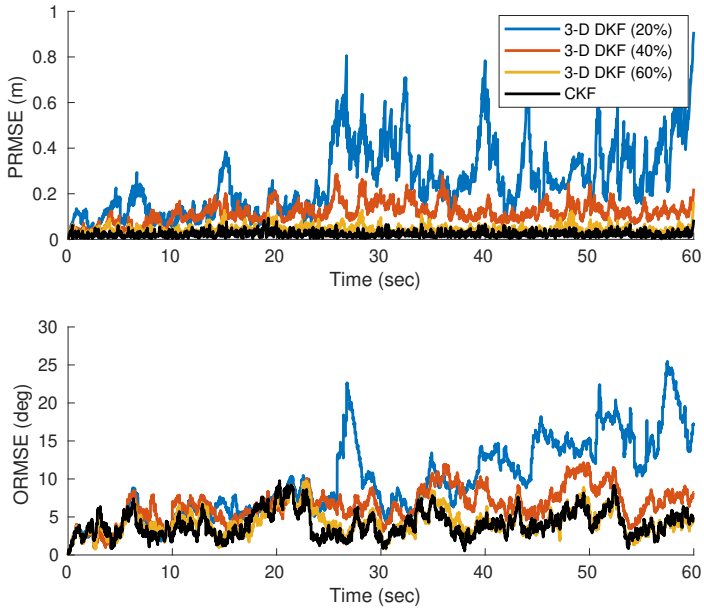

Figure 3.3: Averaged RMSE for the estimated target pose over 50 Monte-Carlo runs and ten cameras.

successfully track the target with 0% communication (no collaboration between cameras). While as the communication percentage increases, all the estimators maintained by each camera become more accurate. When the communication percentage is 40%, the estimated trajectories obtained by the first four cameras are plotted against the groundtruth in Fig. 3.4, which shows that our approach can well track the 3-D trajectory of the target.

3.5 Conclusion

In this chapter, we have introduced a new DKF that is applicable for tacking the 6-DoF motion of a target moving in 3-D environments over sensor networks. The proposed algorithm enjoys the property of being fully distributed as it only uses its own and one-hop neighbors' information. Moreover, it only requires a single communication iteration

Table 3.2: Averaged RMSE for the estimated target pose over 50 Monte-Carlo runs and all timesteps.

communication (3-D DKF)		0 %	20 %	40%	60%
Cam 1	PRMSE (m)	22.654	0.239	0.119	0.041
	ORMSE (deg)	22.246	9.902	6.992	4.320
Cam 2	PRMSE (m)	65.005	0.265	0.123	0.040
	ORMSE (deg)	36.935	10.306	6.917	4.285
Cam 3	PRMSE (m)	72.067	0.217	0.117	0.042
	ORMSE (deg)	26.246	9.970	6.900	4.337
Cam 4	PRMSE (m)	56.871	0.281	0.124	0.043
	ORMSE (deg)	38.271	10.245	6.925	4.314
CKF	PRMSE (m)	0.022			
	ORMSE (deg)	4.128			

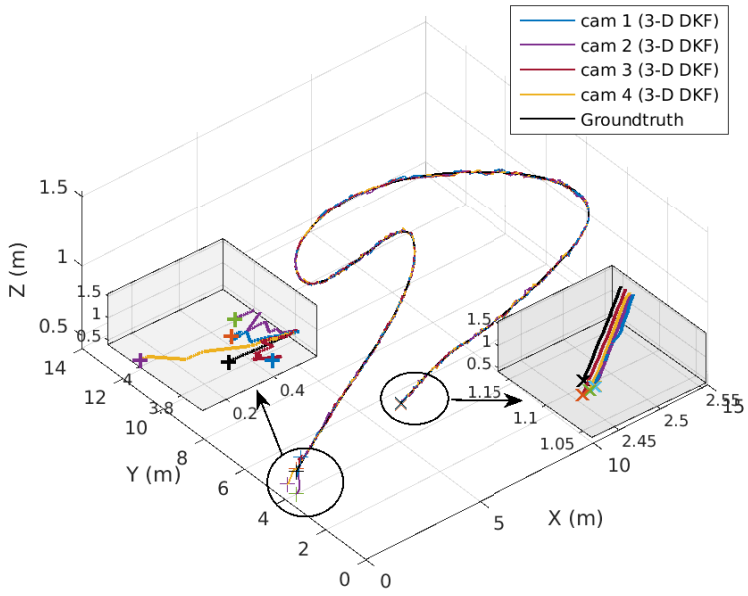

Figure 3.4: Estimated 3-D trajectories of the first four cameras. '+' denotes the start position while 'x' denotes the end point. The start and end areas are enlarged in the built-in figures.

in the update step and is robust to the time-varying changes in the network such as the communication topology, the blind agents, and the network size. It also deals with the generic target and measurement models. These properties ensure that our approach is

applicable in a wide range of cooperative target tacking scenarios. The performance is tested with the application to camera networks via Monte-Carlo simulations.

Chapter 4

Distributed Visual-Inertial Cooperative Localization

4.1 Introduction and Related Works

Camera and inertial measurement unit (IMU) pairs have been at the forefront of multi-robot (or mobile device) applications due to their complementary nature, low cost and small size. Accurate and efficient cooperative localization (CL) that enables multi-user augmented reality (AR) experiences, multi-device cooperative mapping, and multi-vehicle formation control, is a key barrier to overcome due to challenges of communication, distributed computation, and complexity of multi-robot asynchronous measurement constraints. One intuitive strategy to localize a group of robots is to let each member run a single-robot VINS algorithm independently. However, additional geometric constraints (e.g., common feature observations, relative robot-to-robot measurements) can be explored

in multi-robot systems to improve the localization performance, if robots communicate with each other. Then, it holds great potential to design cooperative VINS (C-VINS) algorithms for multi-robot systems.

Significant research efforts have recently been devoted to visual-inertial navigation system (VINS) [48], while primarily focusing on improving *single-robot* VINS accuracy, efficiency, and robustness [93, 103, 36]. The extension to the *multi-robot* case is not sufficiently explored as a naive approach would be prohibitively costly and non-realtime. For example, one could communicate all measurements generated from itself to each other (or fusion center), where all measurements could be optimally fused and all states can be refined jointly. While this does allow for accurate estimation, both the requirement for constant communication and the joint estimation of robot states requires cubic computational complexity in terms of the number of robots. As such, a multi-robot *distributed* estimator is needed to address these shortcomings by relaxing communication requirements and distributing the computation cost across all robots.

Efficient 2D CL has focused on the fusion of relative measurements between robots (e.g., relative robot-to-robot bearing or distance range measurements). Roumeliotis et al. [109] proposed a decentralized algorithm that achieves performance equivalent to the centralized formulation, but required communication between all robots and increases in computational cost due to its centralized nature as the number of robots grow. Other works such as [76] have investigated the approximation of the robot-to-robot cross-covariances that are not involved in a relative measurement update to reduce the computational cost, and while it performs close to its centralized, it is unable to guarantee consistency and

thus can easily diverge. More recently, Jung et al. [56] extended this work to the 3D case, but inherits the same underlying issues and requires maintaining of the approximated robot-to-robot cross-covariances. There exist other works aiming at estimating the relative poses between robots using relative measurements [81, 129]. Alternative approaches have leveraged CI [21, 137] to guarantee consistency and only requires that each robot maintains its own state and auto-covariance (the correlations between robots are ignored). By contrast, in our work we specifically take advantage of the CI formulation for 3D multi-robot state estimation, enabling a consistent distributed algorithm which fuses inertial and visual sparse environmental feature information.

As compared to CL with relative distance, bearing, or poses between robots [109, 76, 68, 21, 137, 56, 81, 80, 65, 129], common sparse environmental features are used in [102, 60, 87, 112], which is appealing as getting relative robot information can be difficult with visual-inertial sensors in practice and requires both the detection and tracking of other robots. For example, Melnyk et al. [87] introduced CL-MSCKF using common environmental feature constraints within a centralized formulation that jointly estimated all robot states. They required that robots communicate all sensor data to a common fusion center and demonstrated its use for the two robot case in simulation. Karrer et al. [60] developed a graph-based centralized server which handled non-realtime computationally expensive loop closure detection and optimization of all robot maps to find the joint global optimal. In this chapter, we instead focus on the computationally efficient *distributed* localization problem where each robot only estimates its *own* state and tries to leverage information from other robots without a centralized server or joint optimization.

As closest to our work, Sartipi et al. [112] introduced a distributed method for multi-user AR experiences through the use of multi-map feature constraints. Common features were detected in environmental maps received from other users and the transmitted feature position estimates were used to constrain the user's state directly. Instead of inflating measurement noise to compensate for the unknown correlations between the current user and the other user's map, we leverage CI that theoretically guarantee consistency to handle the unknown correlations. Also, instead of requiring that all common features must match to sparse features in the other user's map, we leverage the other user's common feature measurements directly allowing for update with additional measurements.

4.2 Cooperative Visual-Inertial System

In this section, we briefly describe the cooperative visual-inertial system that serves the basis for the proposed distributed CI-based estimator. The state vector for the i'th robot contains its current IMU navigation state $\mathbf{x}_{I_i}$, sliding window of cloned IMU poses $\mathbf{x}_{C_i}$, spatial-temporal calibration parameters $\mathbf{x}_{W_i}$, along with a small temporal map (i.e., SLAM features) $\mathbf{x}_{M_i}$ (see [34, 138]).

$$\mathbf{x}_{i,k} = \begin{bmatrix} \mathbf{x}_{I_i}^\top & \mathbf{x}_{W_i}^\top & \mathbf{x}_{C_i}^\top & \mathbf{x}_{M_i}^\top \end{bmatrix}^\top \tag{4.1}$$

$$\mathbf{x}_{I_i} = \begin{bmatrix} {}^{I_{i,k}}_{G}\bar{q}^\top & {}^{G}\mathbf{p}_{I_{i,k}}^\top & {}^{G}\mathbf{v}_{I_{i,k}}^\top & \mathbf{b}_{\omega_{i,k}}^\top & \mathbf{b}_{a_{i,k}}^\top \end{bmatrix}^\top \tag{4.2}$$

$$\mathbf{x}_{W_i} = \begin{bmatrix} {}^{C_i}t_{I_i} & {}^{C_i}_{I_i}\bar{q}^\top & {}^{C_i}\mathbf{p}_{I_i}^\top & \boldsymbol{\zeta}_i^\top \end{bmatrix}^\top \tag{4.3}$$

$$\mathbf{x}_{C_i} = \begin{bmatrix} {}^{I_{i,k-1}}_{G}\bar{q}^\top & {}^{G}\mathbf{p}_{I_{i,k-1}}^\top & \cdots & {}^{I_{i,k-c}}_{G}\bar{q}^\top & {}^{G}\mathbf{p}_{I_{i,k-c}}^\top \end{bmatrix}^\top \tag{4.4}$$

$$\mathbf{x}_{M_i} = \begin{bmatrix} {}^{G}\mathbf{p}_{f1}^\top & \cdots & {}^{G}\mathbf{p}_{fm}^\top \end{bmatrix}^\top \tag{4.5}$$

where ${}^{I_{i,k}}_{G}\bar{q}$ is the unit quaternion parameterizing the rotation $\mathbf{C}({}^{I_{i,k}}_{G}\bar{q}) = {}^{I_{i,k}}_{G}\mathbf{R}$ from the global frame of reference $\{G\}$ to the IMU local frame $\{I_k\}$ at time k for the i'th robot [119], $\mathbf{b}_{\omega_{i,k}}$ and $\mathbf{b}_{a_{i,k}}$ are the gyroscope and accelerometer biases, and ${}^{G}\mathbf{v}_{I_{i,k}}$ and ${}^{G}\mathbf{p}_{I_{i,k}}$ are the velocity and position of the IMU expressed in the global frame, respectively. The clone state $\mathbf{x}_C$ contains c historical IMU poses in a sliding window, while the temporal map state $\mathbf{x}_M$ has m features. Each robot additionally calibrates its camera intrinsics $\boldsymbol{\zeta}_i$, camera-IMU extrinsics, and camera-IMU temporal offset ${}^{C_i}t_{I_i}$ [34]. Finally, given a group of n robots, we have the following combined state and covariance matrix decomposition:

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_{1,k}^\top & \cdots & \mathbf{x}_{n,k}^\top \end{bmatrix}^\top, \quad \mathbf{P}_k = \begin{bmatrix} \mathbf{P}_{11_k} & \cdots & \mathbf{P}_{N1_k} \\ \vdots & \ddots & \vdots \\ \mathbf{P}_{1N_k} & \cdots & \mathbf{P}_{NN_k} \end{bmatrix} \tag{4.6}$$

Here we note that in the centralized formulation this is the state that we jointly estimate along with the cross-covariance terms, while in the distributed case each robot only estimates a sub-set of the total state and correlations between robots are dropped (e.g., robot i only tracks $\mathbf{x}_{i,k}$ and $\mathbf{P}_{ii_k}$).

4.2.1 Inertial Propagation

The inertial state of the i'th robot $\mathbf{x}_{I_i}$ is propagated forward using its own IMU measurements of linear accelerations ($\mathbf{a}_{m_i}$) and angular velocities (ω_{m_i}) based on the fol-

lowing generic nonlinear IMU kinematics [23]:

$$\mathbf{x}_{i,k+1} = \mathbf{f}(\mathbf{x}_{i,k}, \mathbf{a}_{m_k} - \mathbf{n}_{a_k}, \omega_{m_k} - \mathbf{n}_{\omega_k}) \tag{4.7}$$

where $\mathbf{n}_a$ and $\mathbf{n}_\omega$ are the zero-mean white Gaussian noise of the IMU measurements. We linearize this nonlinear model at the current estimate for all robots, and can then propagate the state covariance matrix forward in time:

$$\mathbf{P}_{k|k-1} = \Phi_{k-1}\mathbf{P}_{k-1|k-1}\Phi_{k-1}^\top + \mathbf{Q}_{k-1} \tag{4.8}$$

$$\Phi_{k-1} = \mathbf{Diag}\left(\Phi_{1,k-1}, \ldots, \Phi_{N,k-1}\right) \tag{4.9}$$

$$\mathbf{Q}_{k-1} = \mathbf{Diag}\left(\mathbf{Q}_{1,k-1}, \ldots, \mathbf{Q}_{N,k-1}\right) \tag{4.10}$$

where $\Phi_{i,k}$ and $\mathbf{Q}_{i,k}$ are respectively the system Jacobian and discrete noise covariance for the i'th robot [93], and $\mathbf{Diag}(\cdots)$ creates a block diagonal matrix from the specified values. In the distributed case, all states can be propagated independently since cross-covariance are not tracked.

4.2.2 Camera Measurement Update

A corner feature at time-step k can be be written as the distortion of a perspective projection of a 3D point ${}^{C_{i,k}}\mathbf{p}_f$, expressed in the i'th robot's camera frame:

$$\mathbf{z}_k = \mathbf{h}_{dist}(\mathbf{z}_{k,n}, \boldsymbol{\zeta}_i) + \mathbf{n}_{f_k} \tag{4.11}$$

$$\mathbf{z}_{k,n} = \frac{1}{{}^{C_{i,k}}z_f}\begin{bmatrix} {}^{C_{i,k}}x_f \\ {}^{C_{i,k}}y_f \end{bmatrix} \tag{4.12}$$

$${}^{C_{i,k}}\mathbf{p}_f = {}^{C_i}_{I_i}\mathbf{R}\,{}^{I_{i,k}}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_f - {}^{G}\mathbf{p}_{I_{i,k}}\right) + {}^{C_i}\mathbf{p}_{I_i} \tag{4.13}$$

where $\mathbf{n}_{f_k}$ is the zero-mean white Gaussian measurement noise with covariance $\mathbf{R}_k$, and $\mathbf{h}_{dist}(\cdot)$ is the camera distortion function which maps a normalized bearing $\mathbf{z}_{k,n}$ to the raw distorted image plane. The linearization of this measurement model (4.11) yields the following:

$$\mathbf{r}_{f_k} = \mathbf{H}_k \widetilde{\mathbf{x}}_k + \mathbf{n}_{f_k} = \mathbf{H}_{x_{i,k}} \widetilde{\mathbf{x}}_{I_i} + \mathbf{H}_{f_k}{}^{G}\widetilde{\mathbf{p}}_f + \mathbf{n}_{f_k} \tag{4.14}$$

Once the measurement residual and Jacobian are computed the state and error covariance can be updated using the standard EKF update equations [85].

4.3 Distributed Visual-Inertial CL

As it is known that the standard EKF in the worst case has cubic computation complexity due to its covariance update, a naive implementation of the multi-robot visual-inertial CL can become prohibitively expensive as the number of robots grow in size. Note also that due to communication constraints, the robots might not be able to communicate with all the other robots or a common fusion center. To address these issues, the key idea of our CL approach is to leverage CI [55] to reduce the estimation cost, by only updating the state and error covariance of the current robot (i.e., robot i only updates $\mathbf{x}_{i,k}$ and $\mathbf{P}_{ii_k}$) while ensuring consistency.

In particular, each robot independently propagates its own state and updates with measurements that are only a function of its own state. When updating with measurements of features observed from multiple robots, CI is employed to consistently handle the unknown and untracked cross-covariance terms between the involved robots. This means that robots need to communicate their state and covariance, along with visual feature informa-

tion to the other robots. Each robot tracks a set of visual features using KLT optical flow [74], and communicates its latest tracks and extracted ORB descriptors [111] to the other robots in communication range. A robot then performs descriptor-based feature matching and loop-closure detection to find correspondences between its most recent features and other robots' feature tracks. After tracking and matching, feature tracks are categorized as follows:

(A) VIO features which have only been tracked for a short period of time.

(B) Temporal SLAM features which have been tracked beyond the current sliding window.

(C) Common VIO features which have been matched to features in another robot and tracked for only a short period of time.

(D) Common SLAM features which have been matched to features in another robot. Note that this feature might be either a VIO or SLAM feature in the other robot.

In the following, we present in detail how we update our state with these different feature variants. Note that for the centralized case independent features update the full state and covariance since cross-covariances are tracked, while in the distributed case only the i'th robot state and covariance is updated thus allowing for computational savings.

4.3.1 Independent VIO Feature: MSCKF Update

For VIO features that have lost active track in the current window, we perform MSCKF update [93]. In particular, we first triangulate these features for computing the feature Jacobians $\mathbf{H}_{f_k}$, and then project $\mathbf{r}_{f_k}$ [see (4.14)] onto the left nullspace of $\mathbf{H}_{f_k}$ (i.e.,

$\mathbf{Q}_2^\top \mathbf{H}_{f_k} = \mathbf{0}$) to yield the measurement noise independent of state:

$$\mathbf{Q}_2^\top \mathbf{r}_{f_k} = \mathbf{Q}_2^\top \mathbf{H}_{x_{i,k}} \tilde{\mathbf{x}}_{i,k} + \mathbf{Q}_2^\top \mathbf{H}_{f_k}{}^G\tilde{\mathbf{p}}_f + \mathbf{Q}_2^\top \mathbf{n}_{f_k} \tag{4.15}$$

$$\Rightarrow \ \mathbf{r}'_{f_k} = \mathbf{H}'_x \tilde{\mathbf{x}}_k + \mathbf{n}'_{f_k} \tag{4.16}$$

where $\mathbf{H}_{x_{i,k}}$ is the stacked measurement Jacobians with respect to the navigation states in the current robot's window.

4.3.2 Independent SLAM Feature: FEJ-EKF Update

SLAM features which a robot is able to reliably track longer then its sliding window in length, will be initialized into the SLAM map state vector $\mathbf{x}_{M_i}$. These features are directly updated using the linearized system (4.14) and will remain in the state until they have lost tracking. To improve consistency, we employ First Estimate Jacobians (FEJ) [51] ensuring Jacobians are evaluated at the same linearization points to prevent spurious information gain.

4.3.3 Common VIO Feature: CI-EKF Update

Consider we find a feature which has been seen from multiple robots and want to use this information to update the state. In the centralized case, we would directly update our state with all available measurements (4.14) through the standard EKF since we track the cross-covariance (e.g., $\mathbf{P}_{iN_k}$). In the distributed case, a robot only tracks its own state and autocovariance to ensure computational efficiency and scalability with respect to the robot team size. This presents two key challenges: (i) how to efficiently and consistently fuse multiple robots' autocovariances, and (ii) how to find the data association between different

features, which motivates us to leverage CI to fuse estimates and covariances transmitted from other robots.

CI-EKF Update

Consider the i'th robot has a measurement which is a function of L other robot states. The linearized measurement model can be computed as:

$$\mathbf{r}_{f_k} = \mathbf{H}_{x_{i,k}} \widetilde{\mathbf{x}}_{i,k} + \mathbf{H}_{x_{1..L,k}} \widetilde{\mathbf{x}}_{1..L,k} + \mathbf{H}_{f_k} {}^{G}\widetilde{\mathbf{p}}_f + \mathbf{n}_{f_k} \tag{4.17}$$

where $\mathbf{H}_{x_{i,k}}$ is the Jacobian in respect to the i'th robot state using the k'th estimates, and $\mathbf{H}_{x_{1..L,k}}$ is the stacked Jacobian with respect to all other robots the measurement is a function of. To guarantee consistency when updating with this measurement, we adopt the CI-EKF update [55] to construct a prior covariance such that:

$$\mathbf{Diag}\left(\frac{1}{\omega_i}\mathbf{P}_{ii_k}, \frac{1}{\omega_1}\mathbf{P}_{11_k}, \cdots, \frac{1}{\omega_L}\mathbf{P}_{LL_k}\right) \geq \mathbf{P}_k \tag{4.18}$$

where the left side is the CI covariance with zero off-diagonal elements and the right hand side is the unknown true covariance of the state with cross-covariances [see (4.6)]. The weights $\omega_l > 0$ and $\sum_l \omega_l = 1$, for $l \in \{i, 1..L\}$, can be found optimally [55]. Substituting (4.18) into the standard EKF equations and only selecting the portion that updates the current robot's state (say robot i) yields:

$$\delta\mathbf{x}_{i,k} = \frac{1}{\omega_i}\mathbf{P}_{ii,k|k-1}\mathbf{H}_{x_{i,k}}^{\top}\mathbf{S}_k^{-1}\mathbf{r}'_{f_k} \tag{4.19}$$

$$\mathbf{P}_{ii,k|k} = \frac{1}{\omega_i}\mathbf{P}_{ii,k|k-1} - \frac{1}{\omega_i^2}\mathbf{P}_{ii,k|k-1}\mathbf{H}_{x_{i,k}}^{\top}\mathbf{S}_k^{-1}\mathbf{H}_{x_{i,k}}\mathbf{P}_{ii,k|k-1} \tag{4.20}$$

$$\mathbf{S}_k = \sum_{o\in\{i,1..L\}} \frac{1}{\omega_o}\mathbf{H}_{x_{o,k}}\mathbf{P}_{oo,k|k-1}\mathbf{H}_{x_{o,k}}^{\top} + \mathbf{R}_{f_k} \tag{4.21}$$

where $\delta\mathbf{x}_{i,k}$ is the correction to the state estimate $\hat{\mathbf{x}}_{i,k}$.

Efficient Nullspace Projection

To process common features which are short in length, we leverage the similar logic as in Sec. 4.3.1. For example, we have multiple measurements from two different robots and wish to update our state:

$$\begin{bmatrix} \mathbf{r}_{f_{i,k}} \\ \mathbf{r}_{f_{2,k}} \end{bmatrix} = \begin{bmatrix} \mathbf{H}_{x_{i,k}} & \mathbf{0} & \mathbf{H}_{f_{i,k}} \\ \mathbf{0} & \mathbf{H}_{x_{2,k}} & \mathbf{H}_{f_{2,k}} \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{x}}_{I_i} \\ \widetilde{\mathbf{x}}_{I_2} \\ {}^{G}\widetilde{\mathbf{p}}_{f} \end{bmatrix} + \begin{bmatrix} \mathbf{n}_{f_{i,k}} \\ \mathbf{n}_{f_{2,k}} \end{bmatrix} \tag{4.22}$$

We can then project both equations onto their left range and nullspace (e.g., $\mathbf{H}_{f_{i,k}} = [\mathbf{Q}_{i,1}\ \mathbf{Q}_{i,2}][\mathbf{U}_i\ \mathbf{0}]^\top$):

$$\begin{bmatrix} \mathbf{r}^1_{f_{i,k}} \\ \mathbf{r}^2_{f_{i,k}} \\ \mathbf{r}^1_{f_{2,k}} \\ \mathbf{r}^2_{f_{2,k}} \end{bmatrix} = \begin{bmatrix} \mathbf{Q}_{i,1}^\top \mathbf{H}_{x_{i,k}} & \mathbf{0} & \mathbf{U}_i \\ \mathbf{Q}_{i,2}^\top \mathbf{H}_{x_{i,k}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_{2,1}^\top \mathbf{H}_{x_{2,k}} & \mathbf{U}_2 \\ \mathbf{0} & \mathbf{Q}_{2,2}^\top \mathbf{H}_{x_{2,k}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{x}}_{I_i} \\ \widetilde{\mathbf{x}}_{I_2} \\ {}^{G}\widetilde{\mathbf{p}}_{f} \end{bmatrix} + \begin{bmatrix} \mathbf{n}^1_{f_{i,k}} \\ \mathbf{n}^2_{f_{i,k}} \\ \mathbf{n}^1_{f_{2,k}} \\ \mathbf{n}^2_{f_{2,k}} \end{bmatrix}$$

where we have defined that $\mathbf{r}^1_{f_{i,k}} = \mathbf{Q}_{i,1}^\top \mathbf{r}_{f_{i,k}}$ and $\mathbf{n}^1_{f_{i,k}} = \mathbf{Q}_{i,1}^\top \mathbf{n}_{f_{i,k}}$. Note that the last row is no longer dependent on the current robot's state, $\mathbf{x}_{I_i}$, and thus, this can be discarded since it will not update the state or covariance due to the lack of tracked cross-covariances. This directly reduces the number of measurements involved during update and makes the computation of $\mathbf{S}_k^{-1}$ substantially cheaper [see (4.21)]. We then have the following linear

systems:

$$\begin{bmatrix} \mathbf{r}^1_{f_{i,k}} \\ \mathbf{r}^1_{f_{2,k}} \end{bmatrix} = \begin{bmatrix} \mathbf{Q}^\top_{i,1}\mathbf{H}_{x_{i,k}} & \mathbf{0} & \mathbf{U}_i \\ \mathbf{0} & \mathbf{Q}^\top_{2,1}\mathbf{H}_{x_{2,k}} & \mathbf{U}_2 \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{x}}_{I_i} \\ \widetilde{\mathbf{x}}_{I_2} \\ {}^G\widetilde{\mathbf{p}}_f \end{bmatrix} + \begin{bmatrix} \mathbf{n}^1_{f_{i,k}} \\ \mathbf{n}^1_{f_{2,k}} \end{bmatrix}$$

$$\mathbf{r}^2_{f_{i,k}} = \mathbf{Q}^\top_{i,2}\mathbf{H}_{x_{i,k}}\widetilde{\mathbf{x}}_{I_i} + \mathbf{n}^2_{f_{i,k}} \tag{4.23}$$

A second nullspace projection onto the left nullspace of $\mathbf{H}_f = [\mathbf{U}_i \ \mathbf{U}_2]^\top$ is performed to create a linear system which is only a function of the $\mathbf{x}_{I_i}$ and $\mathbf{x}_{I_2}$ states. The CI-EKF update [see (4.19) and (4.20)] is then used to update the state $\mathbf{x}_{I_i}$. The second equation [see (4.23)] can update the current robot state without CI through the standard EKF equations since it is only a function of the current robot state. This update contains the same information as in the case that we performed a "large" nullspace projection using the full feature Jacobians in (4.22), but results in a much smaller measurement size since we can drop measurement residuals which are not a function of the i'th robot's state.

4.3.4 Common SLAM Feature: CI-EKF Update

There are two different cases for temporal SLAM features: (i) a SLAM feature in the current robot state matches to a feature that is not a SLAM feature in another robot, and (ii) a SLAM feature matches to another robot's SLAM feature. For example as in Fig. 4.1, in the first case we collect the measurements from the other robot ($\mathbf{z}_{1..N}$) and directly apply (4.17) and update both the current robot's poses and its estimate of the SLAM feature. In the second case, we can either follow this same logic (i.e., grab the measurements from the other robot and update current robot's estimate) or we can leverage the knowledge

that the 3D position of these two features should be equal. This SLAM feature constraint model is similar to the one introduced in [39] for cooperative mapping. Consider we have the following two robots:

$$\mathbf{x}_{i,k} = \begin{bmatrix} \mathbf{x}_{I_i}^\top & \mathbf{x}_{W_i}^\top & \mathbf{x}_{C_i}^\top & {}^G\mathbf{p}_{fa}^\top \end{bmatrix}^\top \tag{4.24}$$

$$\mathbf{x}_{2,k} = \begin{bmatrix} \mathbf{x}_{I_2}^\top & \mathbf{x}_{W_2}^\top & \mathbf{x}_{C_2}^\top & {}^G\mathbf{p}_{fb}^\top \end{bmatrix}^\top \tag{4.25}$$

If we have matched feature ${}^G\mathbf{p}_{fa}$ in the current i'th robot to the ${}^G\mathbf{p}_{fb}$ in the other robot, then we can construct the following feature constraint (see Fig. 4.1):

$${}^G\mathbf{p}_{fa} - {}^G\mathbf{p}_{fb} = \mathbf{0} \Rightarrow \ \mathbf{r}_c\left(\mathbf{x}_{i,k}, \mathbf{x}_{2,k}\right) = \mathbf{0} \tag{4.26}$$

which can be linearized to yield:

$$\mathbf{r}_c\left(\hat{\mathbf{x}}_{i,k}, \hat{\mathbf{x}}_{2,k}\right) + \mathbf{H}_{fa}{}^G\widetilde{\mathbf{p}}_{fa} + \mathbf{H}_{fb}{}^G\widetilde{\mathbf{p}}_{fb} \approx \mathbf{0} \tag{4.27}$$

$$\Rightarrow \mathbf{0} - \mathbf{r}_c\left(\hat{\mathbf{x}}_{i,k}, \hat{\mathbf{x}}_{2,k}\right) \approx \mathbf{H}_{fa}{}^G\widetilde{\mathbf{p}}_{fa} + \mathbf{H}_{fb}{}^G\widetilde{\mathbf{p}}_{fb} \tag{4.28}$$

This linearized system can then update the i'th robot state estimate using the CI-EKF update [see (4.19) and (4.20)]. Note that this is a very efficient update, as it is only a function of the two estimated feature positions.

4.3.5 Historical Features: CI-EKF Update

We now explain how to leverage loop-closure constraints to previous robot states. First, to find the feature correspondences between robots, as in [33, 36], each robot create DBoW2 [32] databases for all other robots. When a robot receives feature tracks and descriptors from other robots they are appended to their corresponding DBoW2 database.

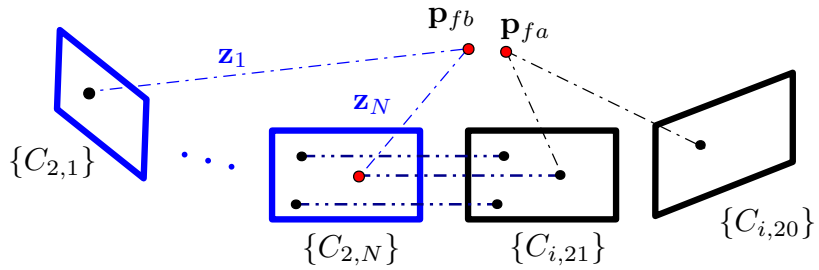

Figure 4.1: Illustration of the keyframe-aided 2D-to-2D matching for data association. Assuming robot i's 21st frame $\{C_{i,21}\}$ matches to the 2nd robot's N'th frame $\{C_{2,N}\}$. We are able to find all feature correspondences between the features the robot's observer, namely $\mathbf{z}_{1..N}$.

The current image can then be queried against the other robots' databases to see if any other robots are or have been at the current location. If a loop-closure is detected and verified using a fundamental matrix geometric check, then we assume that we have detected that another robot has been at our current location. After matching descriptors, we know the correspondences between a feature in the current robot, and that of the features in the other robot (see Fig. 4.1). We can then grab the history of measurements and formulate a common feature update.

To incorporate these measurements from historical states, each robot records the measurement and previous states received from the other agents.[1] Outside of the most recent sliding window, these historical states can provide loop-closure information if we are able to generate measurement constraints to them. Specifically we store the following historical states and covariances in addition to their most recent states published:

$$\mathbf{x}_i = \{\mathbf{x}_{i,0}, \cdots, \mathbf{x}_{i,k-1}\}, \ \mathbf{P}_i = \left\{\mathbf{P}_{ii_0}, \cdots, \mathbf{P}_{ii_{k-1}}\right\} \tag{4.29}$$

Since each one of these historical states contain a sliding window of poses and SLAM

[1] In the future, we plan to investigate the latency introduced due to communication constraints, but historical matching ensures that the robot will leverage *all* available information at the current time including delayed information recently communicated.

features, we only store non-overlapping sliding windows. To accelerate lookup we only store historical descriptor information at a fixed rate (normally 1Hz) since recent frames in the same sliding window contain redundant loop-closure information. More ideal heuristic could be leveraged here to increase match rates. Once loop-closure is detected, we know old historical feature correspondences which we can then use to retrieve measurements and update our current robot state. This update is identical to the CI-EKF update as in Sec. 4.3.3–4.3.4, which only needs to involve the historical windows that contain the historical measurements, and thus is efficient since historical states are not updated.

4.4 Simulations

To validate the proposed method, we have simulated two realistic scenarios both with three robots (see Fig. 4.2). The first is a hand-held mobile AR dataset which has a series of users look and move around a central table, while the second is a series of trajectories from the ETH EuRoC MAV dataset [19]. We employ the OpenVINS simulator [33] to generate realistic visual-bearing and inertial measurements from these supplied trajectories. On average each robot is able to find common features on, respectively, 79.0% and 83.5% (43.7% and 62.7%) of the frames without or with loop-closure in AR datasets (ETH dataset). This clearly shows that advantage of historical loop-closure on datasets which have limited temporal view overlaps between robots. Simulation parameters used are documented in Tab. 4.1. We fix the weight of other robots' covariance in the CI-EKF update as $\omega_o = 0.001$. While for the constraint measurement update presented in Sec. 4.3.4, we use the value $\omega_o = 0.005$ and a synthetic measurement noise of 2cm. Note that while

these weights can be found by minimizing the trace or determinant of $\mathbf{P}_{ii,k|k}$ [55], we have empirically found that using fixed weights still ensures consistent performance. For fair and thorough comparison, we define the following variations of the centralized and proposed distributed CL estimators:

indp – No common features are found between robots and all measurements are processed as independent features which only relate to the current robot.

indp-slam – Same as *indp*, but temporal SLAM features are included in each robot to show the relative improvement.

ce-cmsckf – The centralized estimator using the common VIO features over the sliding window.

ce-cmsckf-cslam – The centralized estimator using the common VIO and SLAM features over the sliding window.

dc-cmsckf [138] – The distributed estimator using the common VIO features over the sliding window.

dc-cmsckf-cslam – The distributed estimator using the common VIO and SLAM features over the sliding window without enforcing the same feature constraint. For example, even if a common SLAM feature is a SLAM feature in another robot's state, we grab the measurements from the other robot and update as the first case in Sec. 4.3.4.

dc-full-window – The distributed estimator using the common VIO and SLAM features over the sliding window with enforcing the same feature constraint.

dc-full-history – The distributed estimator using both the common VIO and SLAM features over the sliding window and from historical matching.

Note that the observed independent VIO features and SLAM features are used in all these estimators. To ensure a fair comparison, the same parameters reported in Tab. 4.1 are used for all algorithms and for all robots.

Table 4.1: Simulation parameters and prior standard deviations that perturbations of measurements and initial states were drawn from.

Parameter	**Value**	**Parameter**	**Value**
Gyro. White Noise	1.6968e-04	Gyro. Rand. Walk	1.9393e-05
Accel. White Noise	2.0000e-3	Accel. Rand. Walk	3.0000e-3
Pixel Proj. (px)	1	Robot Num.	3
IMU Freq. (hz)	400	Cam Freq. (hz)	10
AR Avg. Feats	25	AR Num. SLAM	3
ETH Avg. Feats	50	ETH Num. SLAM	5
Num. Clones	11	Feat. Rep.	GLOBAL

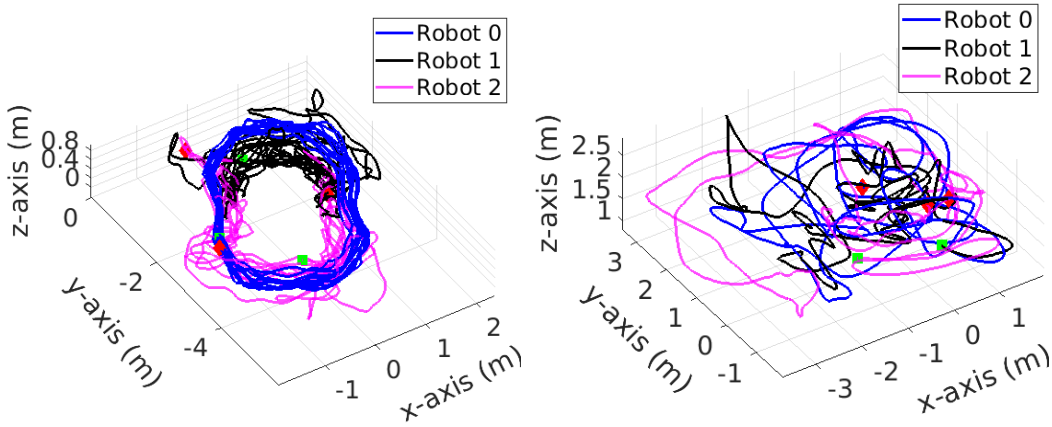

Figure 4.2: Simulated trajectories, axes are in units of meters. General hand-held AR dataset (left) are 147, 93, and 100 meters long, while ETH EuRoC MAV Vicon room datasets (right) are 70, 58, and 59 meters long for each robot. Green square denotes the start and red diamond denotes the end.

Table 4.2: ATE on simulated AR datasets in degrees / meters for each algorithm variation. Green denotes the best, while blue is second best.

Algorithm	Robot 0	Robot 1	Robot 2	Average
indp	1.957 / 0.072	0.811 / 0.041	0.742 / 0.039	1.170 / 0.051
indp-slam	1.396 / 0.046	0.602 / 0.029	0.557 / 0.022	0.852 / 0.032
ce-cmsckf	0.364 / 0.017	0.323 / 0.015	0.355 / 0.015	0.347 / 0.016
ce-cmsckf-cslam	**0.232** / **0.011**	**0.228** / **0.011**	**0.220** / **0.010**	**0.227** / **0.011**
dc-cmsckf	0.759 / 0.029	0.540 / 0.025	0.553 / 0.020	0.617 / 0.025
dc-cmsckf-cslam	0.643 / 0.025	0.496 / 0.022	0.478 / 0.017	0.539 / 0.022
dc-full-window	0.644 / 0.024	0.547 / 0.022	0.480 / 0.017	0.557 / 0.021
dc-full-history	**0.356** / **0.017**	**0.299** / **0.014**	**0.319** / **0.013**	**0.325** / **0.014**

Table 4.3: ATE on simulated ETH datasets in degrees / meters for each algorithm variation. Green denotes the best, while blue is second best.

Algorithm	Robot 0	Robot 1	Robot 2	Average
indp	0.569 / 0.088	0.578 / 0.092	0.560 / 0.093	0.569 / 0.091
indp-slam	0.371 / 0.070	0.406 / 0.069	0.444 / 0.075	0.407 / 0.071
ce-cmsckf	0.221 / 0.052	0.221 / 0.049	0.221 / 0.051	0.221 / 0.050
ce-cmsckf-cslam	**0.151** / **0.042**	**0.143** / **0.038**	**0.144** / **0.040**	**0.146** / **0.040**
dc-cmsckf	0.329 / 0.064	0.342 / 0.061	0.319 / 0.062	0.330 / 0.062
dc-cmsckf-cslam	0.298 / 0.054	0.325 / 0.050	0.290 / 0.052	0.304 / 0.052
dc-full-window	0.285 / 0.052	0.287 / 0.047	0.268 / 0.047	0.280 / 0.049
dc-full-history	**0.211** / **0.029**	**0.207** / **0.031**	**0.218** / **0.030**	**0.212** / **0.030**

4.4.1 Accuracy and Consistency Evaluation

We performed 20 Monte Carlo simulations on each dataset. The average Absolute Trajectory Error (ATE) [132] can be found in Tab. 4.2 and 4.3. It is clear from the top two rows that the additional SLAM features improve `indp`. In the cooperative case, when using the common VIO features, both `ce-msckf` and `dc-msckf` outperform the `indp-slam`, and when including common SLAM features, the accuracy is further improved. It is worth noting that the efficient `dc-full-window` with feature constraint has close accuracy to its

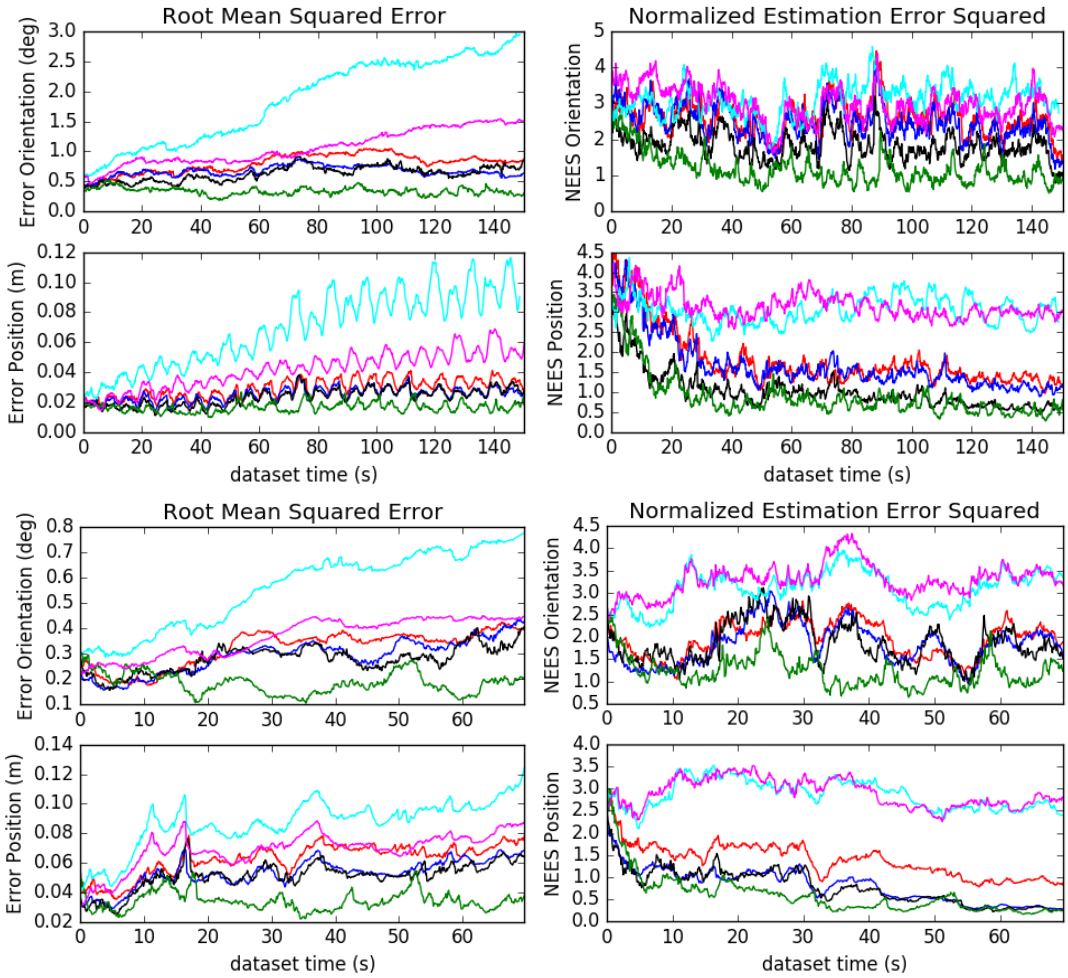

Figure 4.3: Robot 0's average RMSE (left) and NEES (right) results in the simulated AR (top) and ETH datasets (bottom). Cyan represents indp, magenta represents indp-slam, red represents dc-msckf, blue represents dc-cmsckf-cslam, green represents dc-full-window and green represent dc-full-history. Please refer to the color figure.

counterpart `dc-cmsckf-cslam`. Moreover, when including the historical common features, the distributed estimator becomes more accurate as expected. Interestingly, with only the common features over the sliding window, the `ce-cmsckf-cslam` can achieve the best performance on the AR dataset even without loop-closure. This is likely due to the fact that over the whole dataset all robots look in the same general location thus negating any benefit of loop-closure detection. As show in Tab. 4.3 and in the following real-world

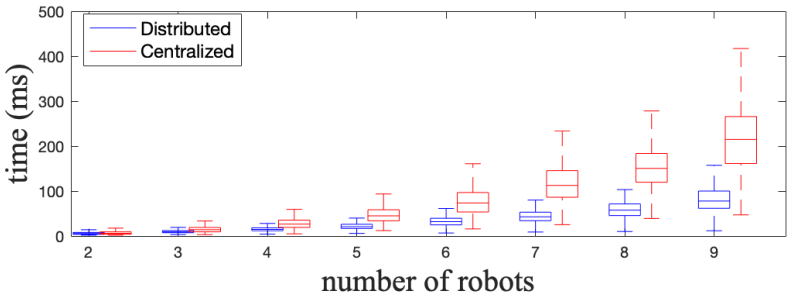

Figure 4.4: Sequential propagation and update time (ms). Note that while decentralized can update in parallel, here we report its sequential timings.

experiments, when robots do not have many overlapping views, the historical information plays an important role.

We additionally show the average Root Mean Square Error (RMSE) [132] and Normalized Estimation Error Squared (NEES) [9] of the distributed algorithms for Robot 0 in Fig. 4.3. The results for the other two robots are similar and are omitted here for space. The `indp` has the largest drift that can be reduced as shown by `indp-slam` and leveraging common features. The `dc-cmsckf-cslam` and `dc-full-window` have almost the same performance while the `dc-full-history` achieves the best accuracy. It is clear that all the distributed algorithms are conservative in nature (NEES is smaller than three) and have smaller NEES than the centralized ones.

4.4.2 Timing Analysis

Multiple Robots

We now investigate the computational efficiency of the proposed work in comparison to the centralized estimator using only common features over the sliding window. We compare the timing results of `dc-full-window` and `ce-cmsckf-cslam` while processing the

Table 4.4: Timing for AR dataset. Millisecond mean and deviation.

Algorithm	**Proposed**	**Combined**
MSCKF update (window)	1.20 ± 0.94	2.88 ± 3.90
MSCKF update (hist)	4.11 ± 5.52	22.75 ± 159.65
Algorithm	**Constraint**	**No Constraint**
SLAM update (window)	0.10 ± 0.03	0.15 ± 0.16
SLAM update (hist)	0.17 ± 0.06	27.11 ± 160.95

same amount of measurements. We first investigate the performance as more robots are added to show the efficiency gains from the distributed formulation. The results in Fig. 4.4 show that as more robots are added, the centralized estimator quickly becomes computationally expensive while the distributed one is able to remain efficient since each robot only needs to propagate and update its own state and auto-covariance. Additionally, if one robot does not find common features in a given frame, the robot can update the estimator independently in the distributed case. On the contrary, the centralized estimator needs to collected all data, propagate, and update the whole state even if there are no common features. The distributed algorithm does have a slight increase in cost, which is due to the increase of common measurements from the additional robots.

Common VIO Features

We next investigate the efficiency of the common VIO feature nullspace projection and subsequent CI-EKF update introduced in Sec. 4.3.3. We report the update time for `dc-full-window` (window) and `dc-full-history` (history) without common SLAM features. The results presented in Tab. 4.4 show that if we use the proposed method to first perform nullspace projection and separate each robot's systems into two systems

(Proposed) we are able to outperform the naive way of performing nullspace projection on a "stacked" Jacobian containing all robot feature Jacobians (Combined). It is clear that in both algorithms, the proposed method is able to have less computational cost, especially in the historical case due to the proposed system reducing the number of measurements in the update. We also note that there is a high level of variance in the historical case due to loop-closure introducing large amounts of measurements in short intervals.

SLAM Constraint Update

Now we investigate the efficiency of the common SLAM feature update introduced in Sec. 4.3.4. Only common SLAM features that can be matched to another robot's SLAM feature are used to ensure that both variants have the same number of measurements in the update. When we match features in the current window, the constraint update (Constraint) is slight more efficient than the naive way of grabbing all the measurements from the other robots (No Constraint) since all robots only have the most recent measurements (in most cases just one). During historical SLAM matching, by definition SLAM features are long feature tracks, and thus many measurements and clones states are associated with a historical SLAM feature. This means that after loop-closure in the naive case (No Constraint) we will process all measurements ever recorded for a SLAM feature which can easily reach many sliding windows in length. If instead we use the constraint update, only the two feature positions are involved, thus the update is extremely efficient in nature (bottom Tab. 4.4).

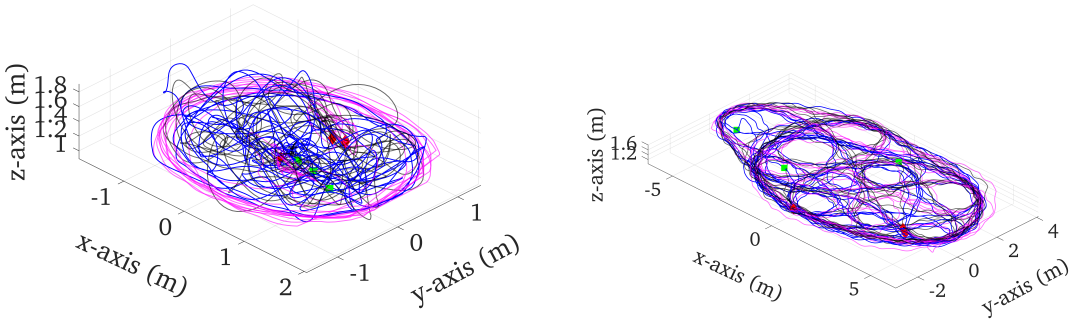

Figure 4.5: TUM-VI groundtruth (left) and Vicon room groundtruth trajectories (right) TUM-VI trajectories are 146, 131, and 134 meters long, while the Vicon room datasets are 507, 509, and 501 meters long.

4.5 Experiments

We have also evaluated the proposed distributed CL estimators on the TUM-VI dataset [113] and a hand collected 10 minute long Vicon room dataset (see Fig. 4.5).[2] Both datasets provide monochrome stereo images at 20Hz and IMU readings at 200Hz. We only leverage the left camera and initialize all robots based on the groundtruth orientation and position with zero velocity. The specific datasets we run on for the TUM-VI are the room1, room3, and room5. For the Vicon room dataset, the groundtruth has been generated using the vicon2gt utility [35]. The shorter TUM-VI dataset has more time periods where multiple robots are looking at the same environmental location (26.7% and 41.8% of the frames detected common features without and with loop-closure), thus provides a good insight into an expected performance in a multi-user AR case where many users are observing the same environment at the *same time*. On the other hand, the Vicon room dataset has near-zero time periods where we are able to detect common features between robots by matching the most recent features. Thus, we use the Vicon room dataset to show the accuracy gain

[2] A video demo https://youtu.be/boHBcVoMKk8

Table 4.5: Relative pose error (RPE) on TUM-VI datasets in degrees / meters averaged over all robots for the dataset.

Algorithm	**40m**	**60m**	**80m**	**100m**	**120m**
indp-slam	1.818 / 0.093	2.833 / 0.126	2.604 / 0.154	2.774 / 0.185	2.716 / 0.215
ce-cmsckf	**1.358** / 0.071	**1.321** / 0.091	1.357 / 0.108	0.843 / 0.128	0.932 / 0.140
ce-cmsckf-cslam	1.758 / **0.069**	1.350 /**0.079**	**1.027** / **0.100**	**0.718** / 0.119	0.938 / **0.130**
dc-cmsckf	1.662 / 0.075	2.005 / 0.104	1.605 / 0.129	1.142 / 0.141	1.531 / 0.170
dc-cmsckf-cslam	1.800 / 0.080	2.642 / 0.093	2.233 / 0.106	1.544 / **0.114**	0.934 / 0.157
dc-full-window	1.768 / 0.075	2.218 / 0.091	1.788 / 0.109	1.257 / 0.123	**0.854** / 0.159
dc-full-history	**1.213** / **0.067**	**1.232** / **0.061**	**1.029** / **0.065**	**1.004** / **0.068**	**0.784** / **0.072**

Table 4.6: Relative pose error (RPE) on Vicon room dataset in degrees / meters averaged over all robots.

Algorithm	**80m**	**100m**	**200m**	**300m**	**420m**
indp-slam	**2.022** / **0.276**	2.416 / 0.334	3.872 / 0.613	5.222 / 0.870	8.045 / 1.189
ce-cmsckf-cslam	2.180 / 0.288	2.603 / 0.333	**2.771** / **0.548**	**3.050** / **0.770**	**3.557** / **1.044**
dc-full-window	2.197 / 0.281	**2.340** / **0.332**	3.322 / 0.580	3.670 / 0.804	5.977 / 1.102
dc-full-history	**1.271** / **0.145**	**1.307** / **0.151**	**1.346** / **0.158**	**1.267** / **0.157**	**1.343** / **0.160**

from leveraging historical loop-closure information by matching to historical states (28.8% of the frames detected common loop-closure features).

4.5.1 TUM-VI Dataset

We use a sliding window of 11, a max of 5 SLAM features, max 30 VIO features per update, 300 active tracks, and perform online calibration of all parameters. For the historical method, we insert keyframes into our database at 5Hz and detect and match to historical keyframes at each timestep. We used a static weight of $\omega_i = 0.99$ and distribute the remaining weight to all other robot covariances used in the CI-EKF update, and for constraint measurement updates [see Eq. (4.28)], we used a value of $\omega_i = 0.995$ and injected a synthetic measurement noise of 2cm to relax the hard constraint.

The Relative Pose Error (RPE) [132] results are shown in Tab. 4.5 solidify the performance gains due to leveraging common features from other robot agents. The inde-

pendent methods which leverage only independent VIO and SLAM feature updates have about three times the error compared to the distributed method which leverages loop-closure information. Additionally, we can see that all variations which leverage common features are able to reduce errors due to the additional information. It is also important to note that even though the distributed variants do not track the cross-covariances between robotic states, the use of CI allows the accuracy to be near the same level as that of the centralized algorithm, and in the case where we leverage historical information (which the centralized algorithm is unable to do), we can slightly outperform for longer trajectory length. The `dc-full-history` method, which leverages loop-closure information, has a relatively constant error as the trajectory lengths increase as expected (showing its drift-free nature).

4.5.2 Vicon Room Dataset

We now present results on the longer hand-held, approximately 500 meter and 10 minute trajectory. We use a sliding window of 11, a max of 20 SLAM features, max 30 VIO features per update, 200 active tracks, and perform online calibration of all parameters. The RPE results for different segment lengths can be found in Tab. 4.6 and give the same conclusion as the previous TUM-VI dataset. It is also important to note that there is very similar performance of the `indp-slam` and `ce-cmsckf-cslam` methods (and their distributed equivalents). This is expected as there are no time periods in any of the robotic trajectories where robots are looking at the same location at the *same* time. Compared to these cases, we have huge accuracy gains due to the inclusion of common feature measurement constraints

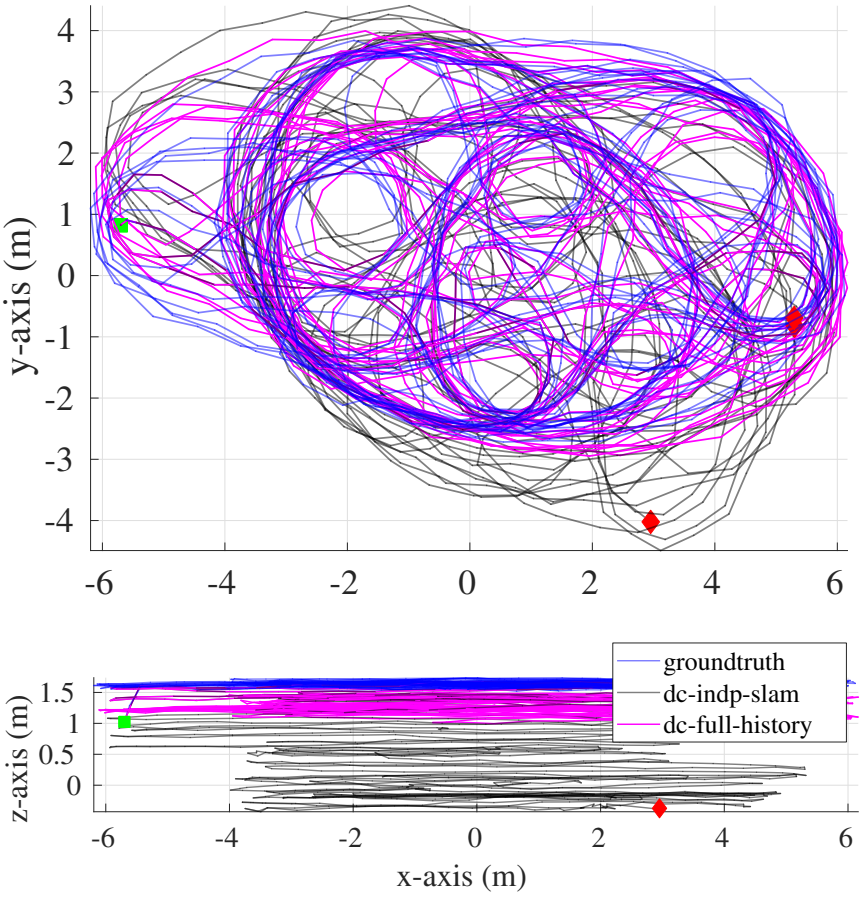

Figure 4.6: Trajectory of groundtruth, independent, and distributed historical trajectory for Robot 0 in the Vicon room dataset. It can be seen that the use of common historical features limit drift in the z-axis along with improvements in x-y accuracy. Please refer to the color figure.

in the historical case, with halved orientation errors and a quarter of the position error at long trajectory lengths. We also plot the groundtruth, `indp-slam`, and `dc-full-history` Robot 0 trajectories in Fig. 4.6, which reinforces that by leveraging historical information we are able to prevent inherent drift in the loop-closure-free case.

4.6 Conclusions

In this chapter, we have presented a distributed visual-inertial cooperative CL estimator that efficiently fuses constraints between robots and leverages temporal SLAM and loop-closure information. We have introduced two different ways to incorporate temporal SLAM features: (i) directly update using the other robot's measurements, and (ii) if both robots are estimating the SLAM feature, a constraint between the two feature positions is leveraged. We have adapted CI to ensure consistent fusion of loop-closure constraints to other agent's historical poses and SLAM features whose cross-correlations are unknown. Extensive simulation and real-world evaluations have demonstrated the performance of the proposed method in realistic scenarios and showed impressive accuracy gains over the single robot case.

Chapter 5

Distributed Joint Visual-Inertial Localization and Target Tracking

5.1 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 visual-inertial navigation systems (VINS) is to achieve multi-robot localization rather than cooperative mapping [38, 40]. The single robot VINS problem has been studied extensively in recent years [93, 69, 104, 94, 28]. 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) [93] 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 [88] 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 [82, 83] where they rely on robot-to-robot camera measurements. But the same as [88], it inherits the drawback of VIO that the estimator exhibits long-term navigation drifts. In contrast, visual-inertial SLAM (VI-SLAM) [69, 94, 104] 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 [61] 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 [25, 106, 105, 27]. Ref. [25] 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 [106], a monocular VINS is built to track the 6-DoF pose of the target. Camera poses are estimated with VINS [105], 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 [27] 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 [110] and [120] where [110] jointly estimates the 6-DoF trajectory of a flying object and the cameras' poses while [120] propose a spatio-temporal BA to jointly estimate the 3-D trajectories of dynamic points and camera intrinsics and extrinsics. Both [110] and [120] 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 [97, 57, 14, 45] 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 [49, 1, 90] or distributed way [135]. 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. 5.1, 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.

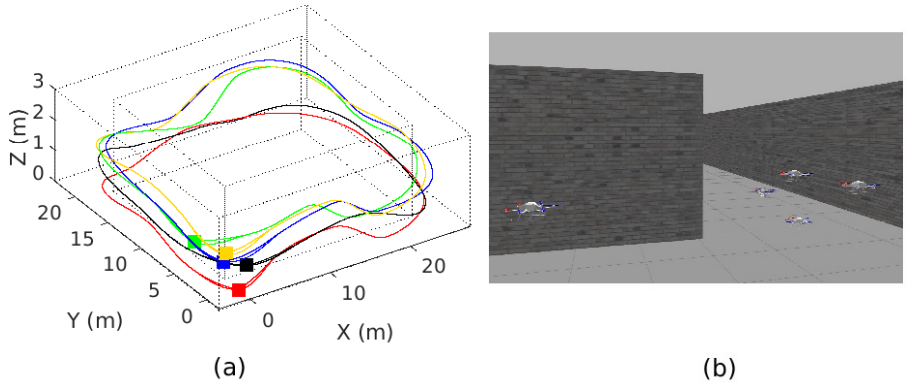

Figure 5.1: Four Firefly drones equipped with visual-inertial sensors track a Pelican drone in a corridor: (a) 3D trajectories for robot 1 (red), robot 2 (green), robot 3 (blue), robot 4 (yellow) and the target (black). The corresponding squares denote the trajectory starts ; (b) Gazebo environment [64].

5.2 Preliminaries

5.2.1 Notations and Definitions

Let the quantity $\mathbf{x}$ represent the true value, $\hat{\mathbf{x}}$ denote the estimated value and $\delta\mathbf{x}$ 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 $\bar{q}$ [18] 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 [30]. For the orientation error, we use the minimal 3-dimensional representation

${}^{G_i}\delta\boldsymbol{\theta}_{I_i}$ [72] which is encoded in G_i. For all the other quantities, $\delta\mathbf{x}$ is defined as the standard additive error $\delta\mathbf{x} = \mathbf{x} - \hat{\mathbf{x}}$ (e.g., ${}^{G_i}\delta\mathbf{p}_{I_i} = {}^{G_i}\mathbf{p}_{I_i} - {}^{G_i}\mathbf{p}_{I_i}$). For a vector $\mathbf{x} = [x\ y\ z]^\mathsf{T}$, the perspective projection function is defined as $\Pi(\mathbf{x}) = \frac{1}{z}[x\ y]^\mathsf{T}$.

5.2.2 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 $i_j \in \mathcal{V}$.

5.3 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.

5.3.1 IMU State

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

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

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

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

With the IMU dynamics [24], each robot can perform the EKF propagation to evolve the current IMU state and the covairance matrix according to [72].

5.3.2 Update Strategy for Robot 1

Localization State Vector

At the imaging timestep k, the localization state for robot 1 is given by

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

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 $\mathbf{x}_{r_1} = [\mathbf{x}_{I_1}^\mathsf{T}\ \mathbf{x}_{C_1}^\mathsf{T}]^\mathsf{T}$ as the robot state. The error states of $\mathbf{x}_{C_1}^k$ and $\mathbf{x}_S$ take the following

form

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

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

$$\begin{aligned} \mathbf{z}_{R_1}^k &= \Pi({}^{C_{1,k}}\mathbf{p}_{f_i}) + \mathbf{n}_1^k, \\ {}^{C_{1,k}}\mathbf{p}_{f_i} &= {}_{I_1}^{C_1}\mathbf{R}\, {}_{G_1}^{I_{1,k}}\mathbf{R}\left({}^{G_1}\mathbf{p}_{f_i} - {}^{G_1}\mathbf{p}_{I_{1,k}}\right) + {}^{C_1}\mathbf{p}_{I_1}, \end{aligned} \tag{5.1}$$

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

Next we briefly describe the adopted VI-SLAM update strategy presented in [37]. 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 [93]. 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

$$\tilde{\mathbf{z}}_{R_1} = \mathbf{H}_{r_1}\delta\mathbf{x}_{r_1}^{k/k-1} + \mathbf{H}_{f_i}{}^{G_1}\delta\mathbf{p}_{f_i} + \mathbf{n}_1, \tag{5.2}$$

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

$$\tilde{\mathbf{z}}'_{R_1} = \mathbf{H}'_{r_1}\delta\mathbf{x}_{r_1}^{k/k-1} + \mathbf{n}'_1 = \mathbf{H}'_{R_1}\delta\mathbf{x}_{R_1}^{k/k-1} + \mathbf{n}'_1, \tag{5.3}$$

where $\tilde{\mathbf{z}}'_{R_1} = \mathbf{N}^\mathsf{T}\tilde{\mathbf{z}}_{R_1}$, $\mathbf{H}'_{r_1} = \mathbf{N}^\mathsf{T}\mathbf{H}_{r_1}$, $\mathbf{n}'_1 = \mathbf{N}^\mathsf{T}\mathbf{n}_1$, and $\mathbf{H}'_{R_1} = [\mathbf{H}'_{r_1}\ \mathbf{0}_{3\times 3n}]$ with $\mathbf{N}^\mathsf{T}\mathbf{H}_{f_i} = \mathbf{0}$. Here, (5.3) 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

$$\tilde{\mathbf{z}}_{R_1}^k = \mathbf{H}_{r_1}^k\delta\mathbf{x}_{r_1}^{k/k-1} + \mathbf{H}_{f_i}^{k\ G_1}\delta\mathbf{p}_{f_1}^{k/k-1} + \mathbf{n}_1^k, \tag{5.4}$$

which can be further written as

$$\tilde{\mathbf{z}}_{R_1}^k = \mathbf{H}_{R_1}^k\delta\mathbf{x}_{R_1}^{k/k-1} + \mathbf{n}_1^k, \tag{5.5}$$

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 (5.5). 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 [37]. 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.

5.3.3 Update Strategy for Robot j

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

$$\mathbf{x}_{R_j}^k = \begin{bmatrix} \mathbf{x}_{I_j}^k & \mathbf{x}_{C_j}^k & {}^{G_j}\mathbf{F}_{G_1} \end{bmatrix}^\mathsf{T},$$

$$\mathbf{x}_{C_j}^k = \begin{bmatrix} {}^{I_{j,k}}_{G_j}\bar{q}^\mathsf{T} & {}^{G_j}\mathbf{p}_{I_{j,k}}^\mathsf{T} & \cdots & {}^{I_{j,k-m}}_{G_j}\bar{q}^\mathsf{T} & {}^{G_j}\mathbf{p}_{I_{j,k-m}}^\mathsf{T} \end{bmatrix}^\mathsf{T},$$

where $\mathbf{x}_{I_j}^k$ and $\mathbf{x}_{C_j}^k$ are the current IMU state and a sliding window containing m cloned historical IMU poses of robot j. We also refer $\mathbf{x}_{r_j} = [\mathbf{x}_{I_j}^\mathsf{T} \ \mathbf{x}_{C_j}^\mathsf{T}]^\mathsf{T}$ as the robot state. The error state of $\mathbf{x}_{C_j}^k$ is defined the same as $\mathbf{x}_{C_1}^k$ and the error state of ${}^{G_j}\mathbf{F}_{G_1}$ is defined as ${}^{G_j}\tilde{\mathbf{F}}_{G_1} = [{}^{G_1}\delta\boldsymbol{\theta}_{G_j}^\mathsf{T} \ {}^{G_j}\delta\mathbf{p}_{G_1}^\mathsf{T}]^\mathsf{T}$. Note that to estimate ${}^{G_j}\mathbf{F}_{G_1}$, we need an initial guess which can be obtained with Horn's method [43] by using the first few (more than three) detected map features.

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 (5.1), 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

$$\mathbf{z}_{R_j}^k = \Pi({}^{C_{j,k}}\mathbf{p}_{f_j}) + \mathbf{n}_j^k, \tag{5.6a}$$

$${}^{C_{j,k}}\mathbf{p}_{f_j} = {}^{C_j}_{I_j}\mathbf{R}\,{}^{I_{j,k}}_{G_j}\mathbf{R}\left({}^{G_j}\mathbf{p}_{f_j} - {}^{G_j}\mathbf{p}_{I_{j,k}}\right) + {}^{C_j}\mathbf{p}_{I_j}, \tag{5.6b}$$

where $\mathbf{n}_j^k$ is the zero-mean white Gaussian noise with covariance $\mathbf{Q}_j^k$. The standard MSCKF update can be performed for the MSCKF feature as described in Section 5.3.2 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

(5.6b) with

$$ {}^{C_{j,k}}\mathbf{p}_{f_j} = {}^{C_j}_{I_j}\mathbf{R}\,{}^{I_{j,k}}_{G_j}\mathbf{R}\left({}^{G_j}_{G_1}\mathbf{R}\,{}^{G_1}\mathbf{p}_{f_j} + {}^{G_j}\mathbf{p}_{G_1} - {}^{G_j}\mathbf{p}_{I_{j,k}}\right) + {}^{C_j}\mathbf{p}_{I_j}. \tag{5.7} $$

Note that (5.7) provides not only the constraints of the IMU pose, but also the constraints of the transformation between two global frames. Linearizaton of (5.6a), (5.7) yields the following residual

$$ \tilde{\mathbf{z}}^k_{R_j} = \mathbf{H}^k_{R_j}\delta\mathbf{x}^{k/k-1}_{R_j} + \mathbf{H}^k_{f_j}\,{}^{G_1}\delta\mathbf{p}_{f_j} + \mathbf{n}^k_j, \tag{5.8} $$

where $\mathbf{H}^k_{R_j}$ and $\mathbf{H}^k_{f_j}$ are the corresponding Jacobians. Unlike (5.2) and (5.5), 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 $\tilde{\mathbf{n}}^k_j = \mathbf{H}^k_{f_j}\,{}^{G_1}\delta\mathbf{p}_{f_j} + \mathbf{n}^k_j$ with the covariance $\tilde{\mathbf{Q}}^k_j = \mathbf{H}^k_{f_j}\mathbf{P}_{f_j}(\mathbf{H}^k_{f_j})^\mathsf{T} + \mathbf{Q}^k_j$. Then (5.8) turns into

$$ \tilde{\mathbf{z}}^k_{R_j} = \mathbf{H}^k_{R_j}\delta\mathbf{x}^{k/k-1}_{R_j} + \tilde{\mathbf{n}}^k_j. \tag{5.9} $$

Equation (5.9) 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 (5.9) 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) [93], 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.

5.4 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.

5.4.1 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 [27]

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

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 [27], 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 [27], 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

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

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

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

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

5.4.2 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

$$\mathbf{z}_{T_1}^k = \Pi({}^{C_{1,k}}\mathbf{p}_{T_k}) + \mathbf{n}_1^k, \tag{5.10}$$

$${}^{C_{1,k}}\mathbf{p}_{T_k} = {}^{C_1}_{I_1}\mathbf{R}\,{}^{I_{1,k}}_{G_1}\mathbf{R}\left({}^{G_1}\mathbf{p}_{T_k} - {}^{G_1}\mathbf{p}_{I_{1,k}}\right) + {}^{C_1}\mathbf{p}_{I_1}. \tag{5.11}$$

While for a non-representative feature ${}^{T}\mathbf{p}_{t_j}$, (5.11) is replaced with

$${}^{C_{1,k}}\mathbf{p}_{T_k} = {}^{C_1}_{I_1}\mathbf{R}\,{}^{I_{1,k}}_{G_1}\mathbf{R}\left({}^{T_k}_{G_1}\mathbf{R}^{\mathsf{T}}\,{}^{T}\mathbf{p}_{t_j} + {}^{G_1}\mathbf{p}_{T_k} - {}^{G_1}\mathbf{p}_{I_{1,k}}\right) + {}^{C_1}\mathbf{p}_{I_1}. \tag{5.12}$$

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

$$\mathbf{z}_{T_j}^k = \Pi({}^{C_{j,k}}\mathbf{p}_T) + \mathbf{n}_j^k, \tag{5.13}$$

$${}^{C_{j,k}}\mathbf{p}_T = {}^{C_j}_{I_j}\mathbf{R}\,{}^{I_{j,k}}_{G_j}\mathbf{R}\left({}^{G_j}_{G_1}\mathbf{R}\,{}^{G_1}\mathbf{p}_{T_k} + {}^{G_j}\mathbf{p}_{G_1} - {}^{G_j}\mathbf{p}_{I_{j,k}}\right) + {}^{C_j}\mathbf{p}_{I_j}. \tag{5.14}$$

While for a non-representative feature ${}^{T}\mathbf{p}_{t_j}$, we replace (5.14) with

$${}^{C_{j,k}}\mathbf{p}_T = {}^{C_j}_{I_j}\mathbf{R}\,{}^{I_{j,k}}_{G_j}\mathbf{R}\left[{}^{G_j}_{G_1}\mathbf{R}({}^{T_k}_{G_1}\mathbf{R}^{\mathsf{T}}\,{}^{T}\mathbf{p}_{t_j} + {}^{G_1}\mathbf{p}_{T_k}) + {}^{G_j}\mathbf{p}_{G_1} - {}^{G_j}\mathbf{p}_{I_{j,k}}\right] + {}^{C_j}\mathbf{p}_{I_j}. \tag{5.15}$$

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

$$\tilde{\mathbf{z}}_{T_i}^k = \check{\mathbf{H}}_{O_i}^k \delta\mathbf{x}_{O_i}^{k/k-1} + \check{\mathbf{H}}_{R_i}^k \delta\mathbf{x}_{R_i}^{k/k-1} + \mathbf{n}_i^k. \tag{5.16}$$

Here, for robot 1, equation (5.16) is computed using (5.10), (5.11) and (5.12). While for robot j, equation (5.16) is computed using (5.13), (5.14) and (5.15). $\check{\mathbf{H}}_{O_i}^k$ and $\check{\mathbf{H}}_{R_i}^k$ are the corresponding localization and tracking state Jacobians.

5.4.3 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 [27]

$$\begin{aligned} {}^{T}_{G_1}\dot{\bar{q}} =& \frac{1}{2}\mathbf{\Omega}({}^{T}\boldsymbol{\omega}){}^{T}_{G_1}\bar{q}, \quad {}^{G_1}\dot{\mathbf{p}}_T = {}^{G_1}\mathbf{v}_T, \\ & {}^{G_1}\dot{\mathbf{v}}_T = \mathbf{n}_v, \quad {}^{T}\dot{\boldsymbol{\omega}} = \mathbf{n}_\omega, \end{aligned} \tag{5.17}$$

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 (5.17), 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 $\bar{\mathbf{n}}_i^k = \check{\mathbf{H}}_{R_i}^k \delta\mathbf{x}_{R_i}^{k/k-1} + \mathbf{n}_i^k$. The corresponding covariance is given by

$$\bar{\mathbf{Q}}_{T_i}^k = \check{\mathbf{H}}_{R_i}^k \mathbf{P}_{R_i}^{k/k-1} (\check{\mathbf{H}}_{R_i}^k)^\mathsf{T} + \mathbf{Q}_{T_i}^k, \tag{5.18}$$

which takes account of the localization states' uncertainties. Then, the measurement residuals (5.16) turn into

$$\tilde{\mathbf{z}}_{T_i}^k = \check{\mathbf{H}}_{O_i}^k \delta\mathbf{x}_{O_i}^{k/k-1} + \bar{\mathbf{n}}_i^k. \tag{5.19}$$

Further, we define two correction terms as

$$\mathbf{s}_i^k = (\check{\mathbf{H}}_{O_i}^k)^\mathsf{T}(\bar{\mathbf{Q}}_{T_i}^k)^{-1}\check{\mathbf{H}}_{O_i}^k, \quad \mathbf{y}_i^k = (\check{\mathbf{H}}_{O_i}^k)^\mathsf{T}(\bar{\mathbf{Q}}_{T_i}^k)^{-1}\tilde{\mathbf{z}}_{T_i}^k, \tag{5.20}$$

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 [78] which finds a quaternion that minimizes the weighed sum of the orientation errors estimated by each robot.

$${}_{G_1}^{T_{i,k}}\check{\bar{q}} = \arg\max_{\bar{q}\in\mathbb{S}^3} \bar{q}^\mathsf{T}\mathbf{M}\bar{q}, \quad \mathbf{M} = \sum_{l\in\mathcal{N}_i^k} \pi_l^k ({}_{G_1}^{T_{l,k/k-1}}\hat{\bar{q}})^\mathsf{T} {}_{G_1}^{T_{l,k/k-1}}\hat{\bar{q}}, \tag{5.21}$$

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

$$\begin{aligned}\mathbf{P}_{O_i}^{k/k} &= \left[\left(\sum_{l\in\mathcal{N}_i^k}\pi_l^k\mathbf{P}_{O_l}^{k/k-1}\right)^{-1} + \sum_{l\in\mathcal{N}_i^k}\mathbf{s}_l^k\right]^{-1}, \\ \hat{\mathbf{x}}_{O_i}^{k/k} &= \sum_{l\in\mathcal{N}_i^k}\pi_l^k \otimes \hat{\mathbf{x}}_{O_l}^{k/k-1} + \mathbf{P}_{O_i}^{k/k}\sum_{l\in\mathcal{N}_i^k}\mathbf{y}_l^k,\end{aligned} \tag{5.22}$$

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 (5.22) can be found in [136]. In (5.22), 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.

5.5 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 [31] 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 mea-

surements obtained by each robot are corrupted by the realistic sensor characteristics as shown in Tab. 5.1. 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 5.1: 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)

5.5.1 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. 5.2 and Fig. 5.3, while Tab. 5.2 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 Tab. 5.2, the estimated transformation from G_j to G_1 is very accurate.

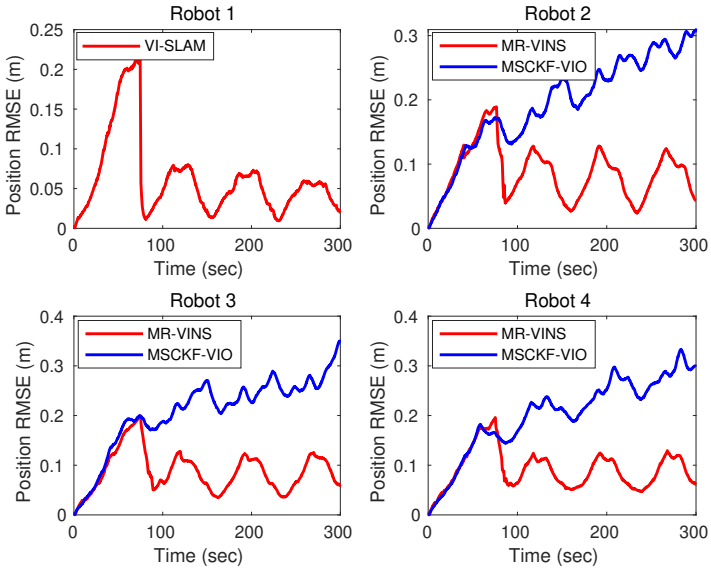

Figure 5.2: Averaged RMSE for the estimated robots' positions.

Table 5.2: 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

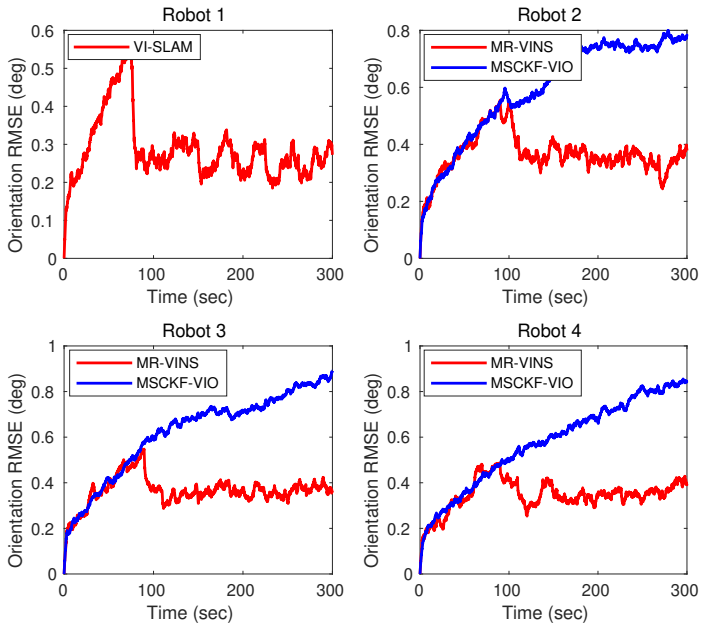

Figure 5.3: Averaged RMSE for the estimated robots' orientations.

5.5.2 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. Tab. 5.3 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. 5.4 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 (5.17) 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.

Table 5.3: 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

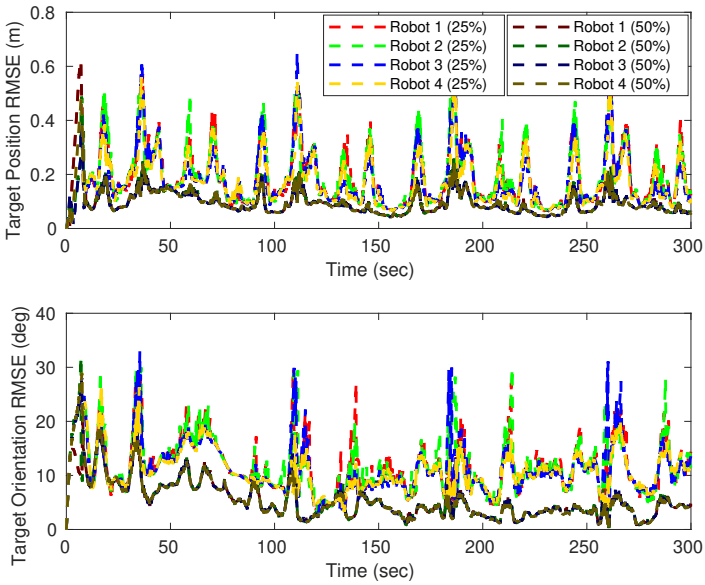

Figure 5.4: Averaged RMSE for the estimated target's poses obtained by the four robots when the communication percentages are 25% and 50%.

5.6 Conclusion

In this chapter, 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.

Chapter 6

Conclusions

The work presented in this manuscript focus on localization and target state estimation, the two crucial tasks, for multi-robot applications. We prefer distributed algorithm, as it outperforms the centralized one in efficiency, robustness and scalability.

The topics we studied is inspired by the the problem of distributed state estimation (DSE) using sensor networks. We realized that DSE assumes *static* sensor networks and *known* sensors' poses. To address these limitations, we studied the problem of jointly localization and target tracking (JLATT). Each robot estimated its own pose (localization) and the target state (tracking) using robot-to-robot and robot-to-target measurements. Furthermore, it is proved that, in the case of linear time-varying models, the estimation errors are bounded in the mean-square sense under very mild conditions. Simulations and experiments showed that better performance in localization is achieved when jointly estimating the robots and target states. The proposed estimators is only applicable to the states in vector space, which limits the application when orientations are in SO3.

Next, we specified the sensors as the most popular combination: camera and IMU. Firstly, we addressed DSE in 3-D scenarios where each camera tracked the state of a 3-D moving object. We demonstrated that each sensor can have a good estimator of the target pose via varying communication and sensing typologies. Next, we focused on the localization problem in 3-D.

By realising that robot-to-robot measurements is more difficult to obtain as compared to the rich environmental features, we explored the environmental features to perform the visual-inertial navigation. We developed a visual-inertial cooperative localization (CL) framework, in which each robot utilizes not only its own measurements but constraints of common features co-observed with its neighbors in order to improve the localization accuracy. The proposed distributed CL estimator is validated against its non-realtime centralized and the non-cooperative counterparts extensively. The estimator is shown to be able to achieve better accuracy with competitive efficiency.

Lastly, we proposed a framework to achieve visual-inertial JLATT (VIJLATT). We further removed two assumptions in the preceding chapters: the assumed known target motion models and a known common global frame for all the robots. In the localization part, one robot performed SLAM while other robots ran VIO and map-based localization. Target was represented as a point cloud and tracked by using the novel distributed Kalman filter. It is shown that each robot have improved accuracy in both localization and tracking parts in cooperative case.

Bibliography

[1] Aamir Ahmad, Gian Diego Tipaldi, Pedro Lima, and Wolfram Burgard. Cooperative robot localization and target tracking based on least squares minimization. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 5696–5701, 2013.

[2] Mehdi Alighanbari and Jonathan P How. Unbiased Kalman consensus algorithm. *Journal of Aerospace Computing, Information, and Communication*, 5(9):298–311, 2008.

[3] Pablo O Arambel, Constantino Rago, and Raman K Mehra. Covariance intersection algorithm for distributed spacecraft state estimation. In *Proc. of the American Control Conference*, pages 4398–4403, 2001.

[4] Nikolay Atanasov, Roberto Tron, Victor M Preciado, and George J Pappas. Joint estimation and localization in sensor networks. In *Proc. of the IEEE Conference on Decision and Control*, pages 6875–6882, 2014.

[5] Alexander Bahr, Matthew R Walter, and John J Leonard. Consistent cooperative localization. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3415–3422, 2009.

[6] Tim Bailey, Mitch Bryson, Hua Mu, John Vial, Lachlan McCalman, and Hugh Durrant-Whyte. Decentralised cooperative localisation for heterogeneous teams of mobile robots. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2859–2865, 2011.

[7] Yaakov Bar-Shalom and Leon Campo. The effect of the common process noise on the two-sensor fused-track covariance. *IEEE Transactions on Aerospace and Electronic Systems*, 22(6):803–805, 1986.

[8] Yaakov Bar-Shalom, X Rong Li, and Thiagalingam Kirubarajan. *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

[9] Yaakov Bar-Shalom, X Rong Li, and Thiagalingam Kirubarajan. *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

[10] Giorgio Battistelli and Luigi Chisci. Kullback–leibler average, consensus on probability densities, and distributed state estimation with guaranteed stability. *Automatica*, 50(3):707–718, 2014.

[11] Giorgio Battistelli and Luigi Chisci. Stability of consensus extended Kalman filter for distributed state estimation. *Automatica*, 68:169–178, 2016.

[12] Giorgio Battistelli, Luigi Chisci, and Claudio Fantacci. Parallel consensus on likelihoods and priors for networked nonlinear filtering. *IEEE Signal Processing Letter*, 21(7):787–791, 2014.

[13] Giorgio Battistelli, Luigi Chisci, Giovanni Mugnai, Alfonso Farina, and Antonio Graziano. Consensus-based algorithms for distributed filtering. In *Proc. of the IEEE Conference on Decision and Control*, pages 794–799, 2012.

[14] Giorgio Battistelli, Luigi Chisci, Giovanni Mugnai, Alfonso Farina, and Antonio Graziano. Consensus-based linear and nonlinear filtering. *IEEE Transactions on Automatic Control*, 60(5):1410–1415, 2014.

[15] Giorgio Battistelli, Luigi Chisci, Giovanni Mugnai, Alfonso Farina, and Antonio Graziano. Consensus-based linear and nonlinear filtering. *IEEE Transactions on Automatic Control*, 60(5):1410–1415, 2015.

[16] Giorgio Battistelli, Luigi Chisci, and Daniela Selvi. A distributed Kalman filter with event-triggered communication and guaranteed stability. *Automatica*, 93:75–82, 2018.

[17] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[18] WG Breckenridge. Quaternions proposed standard conventions. *Jet Propulsion Laboratory, Pasadena, CA, Interoffice Memorandum IOM*, pages 343–79, 1999.

[19] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W Achtelik, and Roland Siegwart. The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, 35(10):1157–1163, 2016.

[20] Luis C Carrillo-Arce, Esha D Nerurkar, José L Gordillo, and Stergios I Roumeliotis. Decentralized multi-robot cooperative localization using covariance intersection. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1412–1417, 2013.

[21] Luis C Carrillo-Arce, Esha D Nerurkar, José L Gordillo, and Stergios I Roumeliotis. Decentralized multi-robot cooperative localization using covariance intersection. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1412–1417. IEEE, 2013.

[22] Federico S Cattivelli and Ali H Sayed. Diffusion strategies for distributed Kalman filtering and smoothing. *IEEE Transactions on Automatic Control*, 55(9):2069–2084, 2010.

[23] Averil B. Chatfield. *Fundamentals of High Accuracy Inertial Navigation*. AIAA, 1997.

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

[25] Jing Chen, Tianbo Liu, and Shaojie Shen. Tracking a moving target in cluttered environments using a quadrotor. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 446–453. IEEE, 2016.

[26] Timothy H Chung, Joel W Burdick, and Richard M Murray. A decentralized motion coordination strategy for dynamic target tracking. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2416–2422. IEEE, 2006.

[27] Kevin Eckenhoff, Yulin Yang, Patrick Geneva, and Guoquan Huang. Tightly-coupled visual-inertial localization and 3-d rigid-body target tracking. *IEEE Robotics and Automation Letters*, 4(2):1541–1548, 2019.

[28] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-manifold preintegration for real-time visual–inertial odometry. *IEEE Transactions on Robotics*, 33(1):1–21, 2016.

[29] Dietrich Franken and Andreas Hupper. Improved fast covariance intersection for distributed data fusion. In *Proc. of the IEEE International Conference on Information Fusion*, volume 1, pages 7–pp, 2005.

[30] Paul Furgale, Joern Rehder, and Roland Siegwart. Unified temporal and spatial calibration for multi-sensor systems. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1280–1286. IEEE, 2013.

[31] Fadri Furrer, Michael Burri, Markus Achtelik, and Roland Siegwart. Rotors—a modular gazebo mav simulator framework. In *Robot Operating System (ROS)*, pages 595–625. Springer, 2016.

[32] Dorian Gálvez-López and J. D. Tardós. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, October 2012.

[33] Patrick Geneva, Kevin Eckenhoff, and Guoquan Huang. A linear-complexity EKF for visual-inertial navigation with loop closures. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3535–3541. IEEE, 2019.

[34] Patrick Geneva, Kevin Eckenhoff, Woosik Lee, Yulin Yang, and Guoquan Huang. Openvins: A research platform for visual-inertial estimation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4666–4672. IEEE, 2020.

[35] Patrick Geneva and Guoquan Huang. vicon2gt: Derivations and analysis. Technical Report RPNG-2020-VICON2GT, University of Delaware, 2020. Available: http://udel.edu/~ghuang/papers/tr_vicon2gt.pdf.

[36] Patrick Geneva, James Maley, and Guoquan Huang. An efficient schmidt-ekf for 3d visual-inertial slam. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12105–12115, 2019.

[37] Patrick Geneva, James Maley, and Guoquan Huang. An efficient schmidt-ekf for 3d visual-inertial slam. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12105–12115, 2019.

[38] Chao X Guo, Kourosh Sartipi, Ryan C DuToit, Georgios A Georgiou, Ruipeng Li, John O'Leary, Esha D Nerurkar, Joel A Hesch, and Stergios I Roumeliotis. Large-scale cooperative 3d visual-inertial mapping in a manhattan world. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1071–1078. IEEE, 2016.

[39] Chao X Guo, Kourosh Sartipi, Ryan C DuToit, Georgios A Georgiou, Ruipeng Li, John O'Leary, Esha D Nerurkar, Joel A Hesch, and Stergios I Roumeliotis. Resource-aware large-scale cooperative three-dimensional mapping using multiple mobile devices. *IEEE Transactions on Robotics*, 34(5):1349–1369, 2018.

[40] Chao X Guo, Kourosh Sartipi, Ryan C DuToit, Georgios A Georgiou, Ruipeng Li, John O'Leary, Esha D Nerurkar, Joel A Hesch, and Stergios I Roumeliotis. Resource-aware large-scale cooperative three-dimensional mapping using multiple mobile devices. *IEEE Transactions on Robotics*, 34(5):1349–1369, 2018.

[41] Xingkang He, Chen Hu, Yiguang Hong, Ling Shi, and Haitao Fang. Distributed Kalman filters with state equality constraints: Time-based and event-triggered communications. *IEEE Transactions on Automatic Control*, 2019.

[42] Xingkang He, Wenchao Xue, and Haitao Fang. Consistent distributed state estimation with global observability over sensor network. *Automatica*, 92:162–172, 2018.

[43] Berthold KP Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, 4(4):629–642, 1987.

[44] Roger A Horn, Roger A Horn, and Charles R Johnson. *Matrix analysis*. Cambridge university press, 1990.

[45] Jinwen Hu, Lihua Xie, and Cishen Zhang. Diffusion Kalman filtering based on covariance intersection. *IEEE Transactions on Signal Processing*, 60(2):891–902, 2011.

[46] Jinwen Hu, Lihua Xie, and Cishen Zhang. Diffusion Kalman filtering based on covariance intersection. *IEEE Transactions on Signal Processing*, 60(2):891–902, 2012.

[47] Zheng Huai and Guoquan Huang. Robocentric visual-inertial odometry. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6319–6326. IEEE, 2018.

[48] Guoquan Huang. Visual-inertial navigation: A concise review. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 9572–9582. IEEE, 2019.

[49] Guoquan Huang, Michael Kaess, and John J Leonard. Consistent unscented incremental smoothing for multi-robot cooperative target tracking. *Robotics and Autonomous Systems*, 69:52–67, 2015.

[50] Guoquan Huang, Anastasios I. Mourikis, and Stergios I. Roumeliotis. A first-estimates jacobian EKF for improving SLAM consistency. In Oussama Khatib, Vijay Kumar, and George J. Pappas, editors, *Experimental Robotics*, volume 54 of *Springer Tracts in Advanced Robotics*, pages 373–382. Springer Berlin Heidelberg, 2009.

[51] Guoquan P Huang, Anastasios I Mourikis, and Stergios I Roumeliotis. Observability-based rules for designing consistent EKF SLAM estimators. *The International Journal of Robotics Research*, 29(5):502–528, 2010.

[52] Guoquan P Huang, Nikolas Trawny, Anastasios I Mourikis, and Stergios I Roumeliotis. Observability-based consistent ekf estimators for multi-robot cooperative localization. *Autonomous Robots*, 30(1):99–122, 2011.

[53] Simon Julier and Jeffrey K Uhlmann. General decentralized data fusion with covariance intersection. In *Handbook of multisensor data fusion*, pages 339–364. CRC Press, 2017.

[54] Simon J Julier and Jeffrey K Uhlmann. A non-divergent estimation algorithm in the presence of unknown correlations. In *Proc. of the American Control Conference*, pages 2369–2373, 1997.

[55] SJ Julier and Jeffrey K Uhlmann. General decentralized data fusion with covariance intersection. *Handbook of multisensor data fusion: theory and practice*, pages 319–344, 2009.

[56] Roland Jung, Christian Brommer, and Stephan Weiss. Decentralized collaborative state estimation for aided inertial navigation. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4673–4679. IEEE, 2020.

[57] Ahmed T Kamal, Jay A Farrell, and Amit K Roy-Chowdhury. Information weighted consensus filters and their application in distributed camera networks. *IEEE Transactions on Automatic Control*, 58(12):3112–3125, 2013.

[58] Ahmed Tashrif Kamal, Chong Ding, Bi Song, Jay A Farrell, and Amit K Roy-Chowdhury. A generalized Kalman consensus filter for wide-area video networks. In *Proc. of the IEEE Conference on Decision and Control, and the European Control Conference*, pages 7863–7869. IEEE, 2011.

[59] Elliott Kaplan and Christopher Hegarty. *Understanding GPS: principles and applications*. Artech House, 2005.

[60] Marco Karrer, Patrik Schmuck, and Margarita Chli. Cvi-slam—collaborative visual-inertial slam. *IEEE Robotics and Automation Letters*, 3(4):2762–2769, 2018.

[61] Marco Karrer, Patrik Schmuck, and Margarita Chli. Cvi-slam—collaborative visual-inertial slam. *IEEE Robotics and Automation Letters*, 3(4):2762–2769, 2018.

[62] Solmaz S Kia, Stephen F Rounds, and Sonia Martinez. A centralized-equivalent decentralized implementation of extended kalman filters for cooperative localization. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3761–3766, 2014.

[63] Daniel P Koch, David O Wheeler, Randal Beard, Tim McLain, and Kevin M Brink. Relative multiplicative extended Kalman filter for observable gps-denied navigation. *[Online]. Available: https://scholarsarchive.byu.edu/facpub/1963/*, 2017.

[64] Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2149–2154. IEEE, 2004.

[65] Pierre-Yves Lajoie, Benjamin Ramtoula, Yun Chang, Luca Carlone, and Giovanni Beltrame. Door-slam: Distributed, online, and outlier resilient slam for robotic teams. *IEEE Robotics and Automation Letters*, 5(2):1656–1663, 2020.

[66] EJ Leffens, F Landis Markley, and Malcolm D Shuster. Kalman filtering for spacecraft attitude estimation. *Journal of Guidance, Control, and Dynamics*, 5(5):417–429, 1982.

[67] Keith YK Leung, Yoni Halpern, Timothy D Barfoot, and Hugh HT Liu. The utias multi-robot cooperative localization and mapping dataset. *The International Journal of Robotics Research*, 30(8):969–974, 2011.

[68] Keith YK Leung, Yoni Halpern, Timothy D Barfoot, and Hugh HT Liu. The utias multi-robot cooperative localization and mapping dataset. *International Journal of Robotics Research*, 30(8):969–974, 2011.

[69] Stefan Leutenegger, Paul Furgale, Vincent Rabaud, Margarita Chli, Kurt Konolige, and Roland Siegwart. Keyframe-based visual-inertial slam using nonlinear optimization. *Proc. of the Robotics: Science and Systems Conference*, 2013.

[70] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334, 2015.

[71] Hao Li and Fawzi Nashashibi. Cooperative multi-vehicle localization using split covariance intersection filter. *IEEE Intelligent transportation systems magazine*, 5(2):33–44, 2013.

[72] Mingyang Li and Anastasios I Mourikis. High-precision, consistent ekf-based visual-inertial odometry. *The International Journal of Robotics Research*, 32(6):690–711, 2013.

[73] Mingyang Li and Anastasios I Mourikis. Optimization-based estimator design for vision-aided inertial navigation. In *Proc. of the Robotics: Science and Systems Conference*, pages 241–248, 2013.

[74] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, pages 674–679, Vancouver, BC, August 1981.

[75] Lukas Luft, Tobias Schubert, Stergios I Roumeliotis, and Wolfram Burgard. Recursive decentralized localization for multi-robot systems with asynchronous pairwise communication. *The International Journal of Robotics Research*, 37(10):1152–1167, 2018.

[76] Lukas Luft, Tobias Schubert, Stergios I Roumeliotis, and Wolfram Burgard. Recursive decentralized localization for multi-robot systems with asynchronous pairwise communication. *International Journal of Robotics Research*, 37(10):1152–1167, 2018.

[77] Simon Lynen, Torsten Sattler, Michael Bosse, Joel A Hesch, Marc Pollefeys, and Roland Siegwart. Get out of my lab: Large-scale, real-time visual-inertial localization. In *Proc. of the Robotics: Science and Systems Conference*, volume 1, 2015.

[78] F Landis Markley, Yang Cheng, John Lucas Crassidis, and Yaakov Oshman. Averaging quaternions. *Journal of Guidance, Control, and Dynamics*, 30(4):1193–1197, 2007.

[79] Agostino Martinelli. Improving the precision on multi robot localization by using a series of filters hierarchically distributed. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1053–1058, 2007.

[80] Agostino Martinelli. Cooperative visual-inertial odometry: Analysis of singularities, degeneracies and minimal cases. *IEEE Robotics and Automation Letters*, 5(2):668–675, 2020.

[81] Agostino Martinelli, Alexander Oliva, and Bernard Mourrain. Cooperative visual-inertial sensor fusion: The analytic solution. *IEEE Robotics and Automation Letters*, 4(2):453–460, 2019.

[82] Agostino Martinelli, Alexander Oliva, and Bernard Mourrain. Cooperative visual-inertial sensor fusion: the analytic solution. *IEEE Robotics and Automation Letters*, 4(2):453–460, 2019.

[83] Agostino Martinelli, Alessandro Renzaglia, and Alexander Oliva. Cooperative visual-inertial sensor fusion: fundamental equations and state determination in closed-form. *Autonomous Robots*, pages 1–19, 2019.

[84] Ion Matei and John S Baras. Consensus-based linear distributed filtering. *Automatica*, 48(8):1776–1782, 2012.

[85] Peter S. Maybeck. *Stochastic Models, Estimation, and Control*, volume 1. Academic Press, London, 1979.

[86] Peter S Maybeck. *Stochastic models, estimation, and control*, volume 3. Academic press, 1982.

[87] Igor V Melnyk, Joel A Hesch, and Stergios I Roumeliotis. Cooperative vision-aided inertial navigation using overlapping views. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 936–943. IEEE, 2012.

[88] Igor V Melnyk, Joel A Hesch, and Stergios I Roumeliotis. Cooperative vision-aided inertial navigation using overlapping views. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 936–943. IEEE, 2012.

[89] Pablo Millán, Luis Orihuela, Carlos Vivas, and Francisco R Rubio. Distributed consensus-based estimation considering network induced delays and dropouts. *Automatica*, 48(10):2726–2729, 2012.

[90] Faraz M Mirzaei, Anastasios I Mourikis, and Stergios I Roumeliotis. On the performance of multi-robot target tracking. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3482–3489. IEEE, 2007.

[91] Fabio Morbidi and Gian Luca Mariottini. Active target tracking and cooperative localization for teams of aerial vehicles. *IEEE Transactions on Control Systems Technology*, 21(5):1694–1707, 2012.

[92] Fabio Morbidi and Gian Luca Mariottini. Active target tracking and cooperative localization for teams of aerial vehicles. *IEEE Transactions on Control Systems Technology*, 21(5):1694–1707, 2013.

[93] Anastasios I Mourikis and Stergios I Roumeliotis. A multi-state constraint Kalman filter for vision-aided inertial navigation. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3565–3572. IEEE, 2007.

[94] Raúl Mur-Artal and Juan D Tardós. Visual-inertial monocular slam with map reuse. *IEEE Robotics and Automation Letters*, 2(2):796–803, 2017.

[95] Wolfgang Niehsen. Information fusion based on fast covariance intersection filtering. In *Proc. of the IEEE International Conference on Information Fusion*, volume 2, pages 901–904, 2002.

[96] Benjamin Noack, Joris Sijs, Marc Reinhardt, and Uwe D Hanebeck. Decentralized data fusion with inverse covariance intersection. *Automatica*, 79:35–41, 2017.

[97] Reza Olfati-Saber. Distributed Kalman filtering for sensor networks. In *Proc. of the IEEE Conference on Decision and Control*, pages 5492–5498, 2007.

[98] Reza Olfati-Saber. Kalman-consensus filter: Optimality, stability, and performance. In *Proc. of the IEEE Conference on Decision and Control, and the Chinese Control Conference*, pages 7036–7042. IEEE, 2009.

[99] Reza Olfati-Saber and Parisa Jalalkamali. Collaborative target tracking using distributed kalman filtering on mobile sensor networks. In *Proc. of the American Control Conference*, pages 1100–1105. IEEE, 2011.

[100] Stefano Panzieri, Federica Pascucci, and Roberto Setola. Multi-robot localisation using interlaced extended kalman filter. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2816–2821, 2006.

[101] Lynne E Parker and Brad A Emmons. Cooperative multi-robot observation of multiple moving targets. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 2082–2089, 1997.

[102] Liam Paull, Guoquan Huang, Mae Seto, and John J Leonard. Communication-constrained multi-auv cooperative slam. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 509–516. IEEE, 2015.

[103] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[104] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[105] Tong Qin and Shaojie Shen. Robust initialization of monocular visual-inertial estimation on aerial robots. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4225–4232. IEEE, 2017.

[106] Kejie Qiu, Tong Qin, Wenliang Gao, and Shaojie Shen. Tracking 3-d motion of dynamic objects using monocular visual-inertial sensing. *IEEE Transactions on Robotics*, 35(4):799–816, 2019.

[107] Kejie Qiu, Tong Qin, Hongwen Xie, and Shaojie Shen. Estimating metric poses of dynamic objects using monocular visual-inertial fusion. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 62–68. IEEE, 2018.

[108] Wei Ren, Randal W Beard, and Derek B Kingston. Multi-agent Kalman consensus with relative uncertainty. In *Proc. of the American Control Conference*, pages 1865–1870, 2005.

[109] Stergios I Roumeliotis and George A Bekey. Distributed multi-robot localization. *IEEE Transactions on Robotics and Automation*, 18(5):781–795, 2002.

[110] Artem Rozantsev, Sudipta N Sinha, Debadeepta Dey, and Pascal Fua. Flight dynamics-based recovery of a uav trajectory using ground cameras. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6030–6039, 2017.

[111] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Proc. of the IEEE International Conference on Computer Vision*, pages 2564–2571. IEEE, 2011.

[112] Kourosh Sartipi, Ryan C DuToit, Christopher B Cobar, and Stergios I Roumeliotis. Decentralized visual-inertial localization and mapping on mobile devices for augmented reality. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2145–2152. IEEE, 2019.

[113] David Schubert, Thore Goll, Nikolaus Demmel, Vladyslav Usenko, Jörg Stückler, and Daniel Cremers. The TUM VI benchmark for evaluating visual-inertial odometry. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1680–1687. IEEE, 2018.

[114] Malcolm D Shuster. A survey of attitude representations. *Navigation*, 8(9):439–517, 1993.

[115] Dan Simon. *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[116] Joan Sola. Quaternion kinematics for the error-state KF. *Laboratoire dAnalyse et dArchitecture des Systemes-Centre national de la recherche scientifique (LAAS-CNRS), Toulouse, France, Tech. Rep*, 2012.

[117] Ashley W Stroupe and Tucker Balch. Value-based action selection for observation with robot teams using probabilistic techniques. *Robotics and Autonomous Systems*, 50(2-3):85–97, 2005.

[118] Sebastian Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, 2002.

[119] Nikolas Trawny and Stergios I. Roumeliotis. Indirect Kalman filter for 3D attitude estimation. Technical report, University of Minnesota, Dept. of Comp. Sci. & Eng., March 2005.

[120] Minh Vo, Srinivasa G Narasimhan, and Yaser Sheikh. Spatiotemporal bundle adjustment for dynamic 3d reconstruction. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1710–1718, 2016.

[121] Thumeera R Wanasinghe, George KI Mann, and Raymond G Gosine. Distributed leader-assistive localization method for a heterogeneous multi-robotic system. *IEEE Transactions on Automation Science and Engineering*, 12(3):795–809, 2015.

[122] Shaocheng Wang, Yang Lyu, and Wei Ren. Unscented-transformation-based distributed nonlinear state estimation: Algorithm, analysis, and experiments. *IEEE Transactions on Control Systems Technology*, 27(5):2016–2029, 2018.

[123] Shaocheng Wang and Wei Ren. On the convergence conditions of distributed dynamic state estimation using sensor networks: A unified framework. *IEEE Transactions on Control Systems Technology*, 26(4):1300–1316, 2017.

[124] Shaocheng Wang and Wei Ren. On the convergence conditions of distributed dynamic state estimation using sensor networks: A unified framework. *IEEE Transactions on Control Systems Technology*, 26(4):1300–1316, 2018.

[125] Shaocheng Wang, Wei Ren, and Jie Chen. Fully distributed state estimation with multiple model approach. In *Proc. of the IEEE Conference on Decision and Control*, pages 2920–2925, 2016.

[126] Shaocheng Wang, Wei Ren, and Jie Chen. Fully distributed dynamic state estimation with uncertain process models. *IEEE Transactions on Control of Network Systems*, 5(4):1841–1851, 2017.

[127] Yimin Wang and X Rong Li. A fast and fault-tolerant convex combination fusion algorithm under unknown cross-correlation. In *Proc. of the IEEE International Conference on Information Fusion*, pages 571–578, 2009.

[128] Guoliang Wei, Wangyan Li, Derui Ding, and Yurong Liu. Stability analysis of covariance intersection-based Kalman consensus filtering for time-varying systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2018.

[129] Hao Xu, Luqi Wang, Yichen Zhang, Kejie Qiu, and Shaojie Shen. Decentralized visual-inertial-UWB fusion for relative state estimation of aerial swarm. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 8776–8782. IEEE, 2020.

[130] Hongsheng Yu and Anastasios I Mourikis. Edge-based visual-inertial odometry. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6670–6677. IEEE, 2017.

[131] Fan Zhang, Guilherme S Pereira, and Vijay Kumar. Cooperative localization and tracking in distributed robot-sensor networks. *Tsinghua Science & Technology*, 10(1):91–101, 2005.

[132] Zichao Zhang and Davide Scaramuzza. A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7244–7251. IEEE, 2018.

[133] Ke Zhou and Stergios I Roumeliotis. Optimal motion strategies for range-only constrained multi-sensor target tracking. *IEEE Transactions on Robotics*, 24(5):1168–1185, 2008.

[134] Ke Zhou, Stergios I Roumeliotis, et al. Multi-robot active target tracking with combinations of relative observations. *IEEE Transactions on Robotics*, 27(4):678–695, 2011.

[135] Pengxiang Zhu and Wei Ren. Multi-robot joint localization and target tracking with local sensing and communication. In *Proc. of the American Control Conference*, pages 3261–3266. IEEE, 2019.

[136] Pengxiang Zhu and Wei Ren. Distributed kalman filter for 3-d moving object tracking over sensor networks. In *Proc. of the IEEE Conference on Decision and Control*. IEEE, 2020.

[137] Pengxiang Zhu and Wei Ren. Fully distributed joint localization and target tracking with mobile robot networks. *IEEE Transactions on Control Systems Technology*, 2020.

[138] Pengxiang Zhu, Yulin Yang, Wei Ren, and Guoquan Huang. Cooperative visual-inertial odometry. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 13135–13141. IEEE, 2021.