Aerospace Science and Technology 112 (2021) 106628

Contents lists available at ScienceDirect

Aerospace Science and Technology

www.elsevier.com/locate/aescte

Performance estimation for Kalman filter based multi-agent cooperative navigation by employing graph theory

Shizhuang Wang, Xingqun Zhan *, Yawei Zhai, Jiawen Shen, Hanyu Wang

School of Aeronautics and Astronautics, Shanghai Jiao Tong University, Shanghai, 200240, China

ARTICLE INFO

Article history:
Received 21 January 2021
Received in revised form 27 February 2021
Accepted 28 February 2021
Available online 5 March 2021
Communicated by Chaoyong Li

Keywords:
Cooperative navigation
Kalman filter
Multi-agent system
Graph theory
Navigation performance analysis

ABSTRACT

Cooperative Navigation (CN) exploits inter-agent relative measurement and communication to achieve navigation performance improvement. This technique has attracted worldwide interest in many multi-agent (e.g., multiple unmanned air vehicles) applications, due to its significant advantage over non-cooperative approaches. Comparing to prior studies which mainly focused on performance analysis under a fixed CN framework, this work aims at establishing the theoretical basis to quantify the navigation performance for different CN integration architectures. Two graph variables are defined to describe the architecture: relative measurement graph and communication & fusion graph. The measurements are integrated through extended Kalman filters, and the state covariance matrices are rigorously derived and bounded to establish the relationship between the navigation performance and the integration architecture. Simulations are carried out to demonstrate and validate the proposed framework, and the results show its feasibility and effectiveness.

1. Introduction

Multi-agent systems (e.g., multiple aerial vehicles, multiple missiles, etc.) have attracted increasing interest because they bring significant benefits to both civilian (e.g., surveillance, environment monitoring, and transportation) and military (e.g., target tracking, air exploration, air combat) applications [1–4]. Aside from cooperative control [5,6], cooperative guidance [7,8] and task allocation [9], Cooperative Navigation (CN) is one of the basic and key abilities to enable multi-agent cooperative operations. As a novel direction of navigation technique, CN can potentially improve navigation performance, enhance navigation robustness, and reduce navigation costs [10].

CN is a "*navigation plus communication*" technique focusing on the navigation demands of clustered moving agents. Essentially, CN is a state estimation problem using both the local information from the onboard sensors and the external information shared by the cooperative agents. In this way, CN can provide improved navigation performance and enhanced robustness as compared to non-cooperative approaches because CN explicitly utilizes more information in the state estimation process [11].

Prior research on CN can be categorized by agent type as follows: Unmanned Aerial Vehicles (UAVs) [1,12–16], spacecrafts [17], Automated Underwater Vehicles (AUVs) [18,19], ground vehicles [20,21], smartphones [22], etc. Research on CN for multi-AUV has lasted for a long time, and there has been rich and in-depth literature in this field. In contrast, the studies have been recently focusing more on the ground, air, and space multi-agent systems. These agents are equipped with multiple sensors for navigation purpose, including Global Navigation Satellite System (GNSS), Inertial Measurement Unit (IMU), camera, Ultra-Wideband (UWB), etc. It is worth noting that there are also various researches focusing on the cooperative static node localization in Wireless Sensor Network [23], in which the node localization is usually estimated using Least-Squares (LS) method [24] or optimization-based methods (e.g., semi-definite programing [25], multidimensional scaling [26], etc.). Cooperative node localization in WSN is beyond the scope of this work, however, the framework developed in this paper can be extended to this case with some modifications.

Different state estimation methods are employed in existing CN approaches, including the snapshot LS method [27], Extended Kalman Filter (EKF) [28,29], Federated Kalman Filter (FKF) [30], Unscented Kalman Filter (UKF) [31], Particle Filter (PF) [22,32], Factor Graph (FG) [1,21,33], etc. The main advantage of EKF is its relatively low complexity and its ability to provide the quality of the estimates (i.e., the

* Corresponding author.
E-mail address: xqzhan@sjtu.edu.cn (X. Zhan).

https://doi.org/10.1016/j.ast.2021.106628
1270-9638/

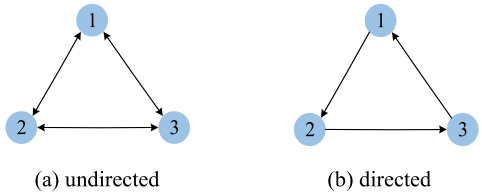


Fig. 1. An illustrative example of undirected and directed graphs.

variances). Therefore, it is widely used in many CN approaches, and its effectiveness has also been proved in these studies. This work will employ EKF as the state estimator in CN systems. And the proposed approaches in this work can also be adapted to support some other estimators, e.g., FG, in future work.

The performance of a CN system is highly dependent on the integration architecture. In the literature, the most common integration architectures include the "fully-centralized" and the "fully-distributed" schemes. In the fully-centralized architecture, the states of all agents are estimated together in the Fusion Center (FC) using all the available information [30,34,35]. And in the fully-distributed scheme, each agent estimates its own navigation states using the local information and the information shared by its "neighbors" (defined as the adjacent agents that this agent can communicate with) [36,37]. The former gives the best estimation accuracy at the expense of large computation and communication loads, and it is even not practically feasible in the case of a large network. Conversely, the latter reduces the computation and communication loads while its performance is worse than the fully-centralized one. Aside from the two schemes above, there are various other "centralized" and "distributed" integration architectures with the performance and payload in between. This work not only addresses the two typical schemes but also the general ones.

Prior studies shown above mainly focus on proving the performance improvement offered by CN over non-cooperative approaches by simulations or experiments. Although different algorithms are proposed in these researches, little work aims at revealing the theoretical relationship between the CN performance and the integration architecture. In response, this work establishes a performance estimation framework for EKF-based CN systems. We first introduce two graph variables, Relative Measurement Graph (RMG) and Communication & Fusion Graph (CFG), to mathematically describe the integration architectures. Using graph theory and EKF principles, we then derive the relationship between the navigation accuracy with the integration architecture. This framework lays the foundation for the design of the CN architectures and is beneficial to navigation performance prediction before or during multi-agent cooperative operations. Besides, this framework is developed in a general multi-agent case and thus can be applied to any multi-agent systems, e.g., multi-UAV, multi-missile, multi-vehicle systems.

The rest of paper is organized as follows. Section 2 formulates the multi-agent CN problem after providing the preliminaries on graph theory and EKF. Then, we propose the performance estimation framework for EKF-based CN systems in Section 3. Simulations are carried out in Section 4. Finally, Section 5 draws the conclusions.

2. Preliminaries and problem formulation

2.1. Preliminary on graph theory

First, we introduce the preliminaries on graph theory and algebraic graph theory. A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is composed of a non-empty node set $\mathcal{V}$ and an edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ [38]. The edge set $\mathcal{E}$ describes the connectivity among the nodes in $\mathcal{V}$. A distinction is made between undirected graph and directed graph: the edges link the two nodes symmetrically in the former while the link is asymmetric in the latter. Fig. 1 shows the difference between undirected and directed graphs. We will use the directed graph for the following analysis because it represents general cases.

If (j, i) is an edge in a directed graph $\mathcal{G}$, i.e., there is a link from j to i, then node j is called an in-neighbor (or a neighbor) of node i while i is called an out-neighbor of j. We define $\mathcal{N}_i$ as the in-neighbor set of node i, i.e., $\mathcal{N}_i = \{j \in \mathcal{V} : (j, i) \in \mathcal{E}\}$. In algebraic graph theory, there are some commonly-used matrices that describe the interconnection topology of a network [39]. The *adjacency matrix* $A = [a_{ij}]$ is a nonnegative matrix defined by: $a_{ij} = 1$, if $j \in \mathcal{N}_i$; $a_{ij} = 0$, otherwise. The *degree matrix* Δ is a diagonal matrix describing the number of in-neighbors of each node. Finally, the *Laplacian matrix* $\mathcal{L}$ is defined as $\mathcal{L} = \Delta - \mathcal{A}$.

2.2. Preliminary on EKF

EKF is the extension of classic Kalman Filter (KF) for nonlinear filtering problems and thus is more general than KF [40]. In EKF, the state transition and observation models can be nonlinear functions. Equations (1) and (2) show the discrete form of these two functions.

$$\boldsymbol{x}_k = \boldsymbol{f}\left(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k\right) + \boldsymbol{\omega}_k \tag{1}$$

$$\boldsymbol{z}_k = \boldsymbol{h}\left(\boldsymbol{x}_k\right) + \boldsymbol{\upsilon}_k \tag{2}$$

where $\boldsymbol{x}$ is the state vector, $\boldsymbol{f}$ the state transition function, $\boldsymbol{u}$ the input, $\boldsymbol{\omega}$ the process noise, $\boldsymbol{z}$ the measurement vector, $\boldsymbol{h}$ the observation function, $\boldsymbol{\upsilon}$ the measurement noise vector, and the subscript k denotes the time epoch.

In EKF, the state transition function $\boldsymbol{f}$ is used to compute the predicted states using the previous estimates, and the observation function $\boldsymbol{h}$ is used to calculate the predicted measurements from the predicted states. To derive the state covariances, we need to compute the Jacobian matrices (including state transition matrix $\boldsymbol{F}$ and observation matrix $\boldsymbol{H}$) by linearizing $\boldsymbol{f}$ and $\boldsymbol{h}$, respectively. Specifically, the Jacobians are evaluated with current predicted states at each epoch.

Similar to KF, the steps of EKF can be divided into two distinct phases: "prediction" and "update". As shown in Equations (3)–(4), the prediction step is used to predict the current states and the associated covariance matrix.

$$\overline{\boldsymbol{x}}_k = \boldsymbol{f}\left(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k\right) \tag{3}$$

$$\overline{\boldsymbol{P}}_k = \boldsymbol{F}_k \hat{\boldsymbol{P}}_{k-1} \boldsymbol{F}_k^{\mathrm{T}} + \boldsymbol{Q}_k \tag{4}$$

where $\boldsymbol{P}$ denotes the state covariance matrix, $\boldsymbol{Q}$ is the covariance matrix of the process noise $\boldsymbol{\omega}$, and the symbols "$-$" and "$\wedge$" on the top of the variables indicate "predicted" and "updated", respectively. Then, Equations (5)–(7) describe the update steps of EKF.

$$\boldsymbol{K}_k = \overline{\boldsymbol{P}}_k \boldsymbol{H}_k^{\mathrm{T}} \cdot \left(\boldsymbol{H}_k \overline{\boldsymbol{P}}_k \boldsymbol{H}_k^{\mathrm{T}} + \boldsymbol{R}_k\right)^{-1} \tag{5}$$

$$\hat{\boldsymbol{x}}_k = \overline{\boldsymbol{x}}_k + \boldsymbol{K}_k \left(\boldsymbol{z}_k - \boldsymbol{h}\left(\overline{\boldsymbol{x}}_k\right)\right) \tag{6}$$

$$\hat{\boldsymbol{P}}_k = \left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right) \overline{\boldsymbol{P}}_k \tag{7}$$

where $\boldsymbol{I}$ is the identity matrix, $\boldsymbol{K}$ denotes the gain matrix, and $\boldsymbol{R}$ is the covariance matrix of the measurement noise $\boldsymbol{\upsilon}$.

2.3. Problem formulation for multi-agent cooperative navigation

Here, we will formulate the multi-agent cooperative navigation as a state estimation problem over a sensor network. It is assumed that the multi-agent system has n agents with a specific interconnection (i.e., inter-agent sensing and communication) topology. Let $\boldsymbol{x}_k^i$ be the true navigation state vector of agent i at time epoch k, which usually consists of the agent's position, velocity, attitude, etc. The dynamic models of each agent i ($\forall i$=1, 2, …, n) are assumed to be known and expressed by the following:

$$\boldsymbol{x}_k^i = \boldsymbol{f}_k^i\left(\boldsymbol{x}_{k-1}^i, \boldsymbol{u}_k^i\right) + \boldsymbol{\omega}_k^i, \ \forall i=1, 2, \ldots, n \tag{8}$$

where $\boldsymbol{\omega}_k^i$ is the process noise vector whose covariance is $\boldsymbol{Q}_k^i$.

Besides, each agent is equipped with multiple sensors that output absolute and/or relative measurements. The absolute exteroceptive measurement is only for one agent, e.g., pseudoranges from GNSS receivers, while the relative (i.e., inter-agent) measurement is associated with two different agents, e.g., range or bearing measurements between them. We use $\boldsymbol{z}_k^i$ to denote the absolute measurement vector for agent i at epoch k, and the relationship between $\boldsymbol{z}_k^i$ and $\boldsymbol{x}_k^i$ is given by:

$$\boldsymbol{z}_k^i = \boldsymbol{h}_k^i\left(\boldsymbol{x}_k^i\right) + \boldsymbol{\upsilon}_k^i \tag{9}$$

The associated observation matrix is denoted by $\boldsymbol{H}_k^i$ which is calculated by linearizing the above equation at $\boldsymbol{x}_k^i = \overline{\boldsymbol{x}}_k^i$. Similarly, when agent i finds agent j ($i \neq j$) and obtains the relative measurements between them, the relative measurement can be modeled as:

$$\boldsymbol{z}_k^{ij} = \boldsymbol{h}_k^{ij}\left(\boldsymbol{x}_k^i, \boldsymbol{x}_k^j\right) + \boldsymbol{\upsilon}_k^{ij} \tag{10}$$

where $\boldsymbol{\upsilon}_k^i$ and $\boldsymbol{\upsilon}_k^{ij}$ are the measurement noise vectors, and their covariance matrices are $\boldsymbol{R}_k^i$ and $\boldsymbol{R}_k^{ij}$, respectively. The linearized observation matrices associated with $\boldsymbol{x}_k^i$ and $\boldsymbol{x}_k^j$ are separately denoted by $\overleftarrow{\boldsymbol{H}}_k^{ij}$ and $\overrightarrow{\boldsymbol{H}}_k^{ij}$, where $\overleftarrow{\boldsymbol{H}}_k^{ij} = \left(\partial \boldsymbol{z}_k^{ij} / \partial \boldsymbol{x}_k^i\right)\Big|_{\left(\overline{\boldsymbol{x}}_k^i, \ \overline{\boldsymbol{x}}_k^j\right)}$ and $\overrightarrow{\boldsymbol{H}}_k^{ij} = \left(\partial \boldsymbol{z}_k^{ij} / \partial \boldsymbol{x}_k^j\right)\Big|_{\left(\overline{\boldsymbol{x}}_k^i, \overline{\boldsymbol{x}}_k^j\right)}$. In practice, a relative measurement may be available to both agents, e.g., both the agents have access to the same range measurement between them. In this case, we have $\boldsymbol{z}_k^{ij} = \boldsymbol{z}_k^{ji}$, which means $\boldsymbol{\upsilon}_k^{ij}$ is always equal to $\boldsymbol{\upsilon}_k^{ji}$. On the contrary, the relative measurement may be only available to one agent (e.g., i), and then only $\boldsymbol{z}_k^{ij}$ exists in this case.

Then, we introduce a directed graph variable $\mathcal{G}_{\mathrm{M}} = (\mathcal{V}, \mathcal{E}_{\mathrm{M}})$, called Relative Measurement Graph (RMG), to describe the relative sensing topology in the network. The node set $\mathcal{V}$ is composed of each agent, and each edge (j, i) in the edge set $\mathcal{E}_{\mathrm{M}} \subseteq \mathcal{V} \times \mathcal{V}$ indicates that agent i can obtain the relative measurements between agents i and j. The adjacency matrix of $\mathcal{G}_{\mathrm{M}}$ is denoted by $A_{\mathrm{M}} = \left[a_{\mathrm{M}}^{ij}\right]$ where we have: a_{M}^{ij}=1, if agent i can obtain the measurement $\boldsymbol{z}_k^{ij}$; a_{M}^{ij}=0, otherwise.

For further illustration, Fig. 2 offers an example of a multi-agent system which has 4 agents. This figure shows not only the state transition relationship between epochs but also the graph of absolute and relative measurements. And it also gives the corresponding adjacency matrix of the RMG. From an EKF perspective, this figure demonstrates all the information available in the network at current epoch. In the non-cooperative case, each agent i can estimate its own states $\boldsymbol{x}_k^i$ using the local state transition model $\boldsymbol{f}_k^i$ and the absolute measurements $\boldsymbol{z}_k^i$. Then in the following, we will focus on the implementation of cooperative state estimation using both local information and inter-agent relative measurements.

For a given multi-agent system, there are usually various CN integration architectures, each of which uses different sensing and communication topology from the others. For each architecture, the information in use is a subset of all the information available in the network. Similarly, the actual communication topology is within the communication topology constraint. Taking the system shown above as an example, Fig. 3 provides three different integration architectures: the fully-centralized, the semi-distributed, and the distributed ones. Please note, these architectures are only a small subset of all the possible ones.

As shown in Fig. 3, under the fully-centralized architecture, all the states are estimated together in a fusion center by employing all the information. On the contrary, under the semi-distributed and the distributed architectures, there are multiple parallel filters, each of which utilizes a subset of the information to estimate the states of some agents. And there exist communication links among these filters to exchange information (i.e., state estimates) for estimation performance enhancement.

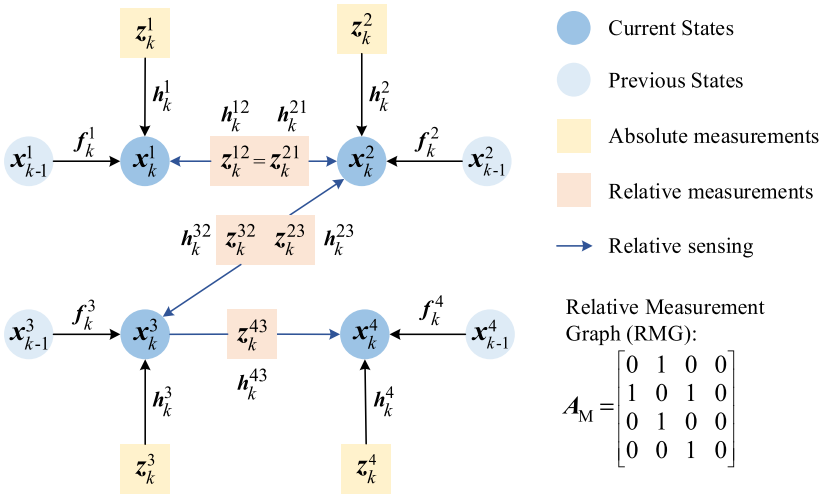


Fig. 2. State transition and measurement graph in an example of a multi-agent system. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

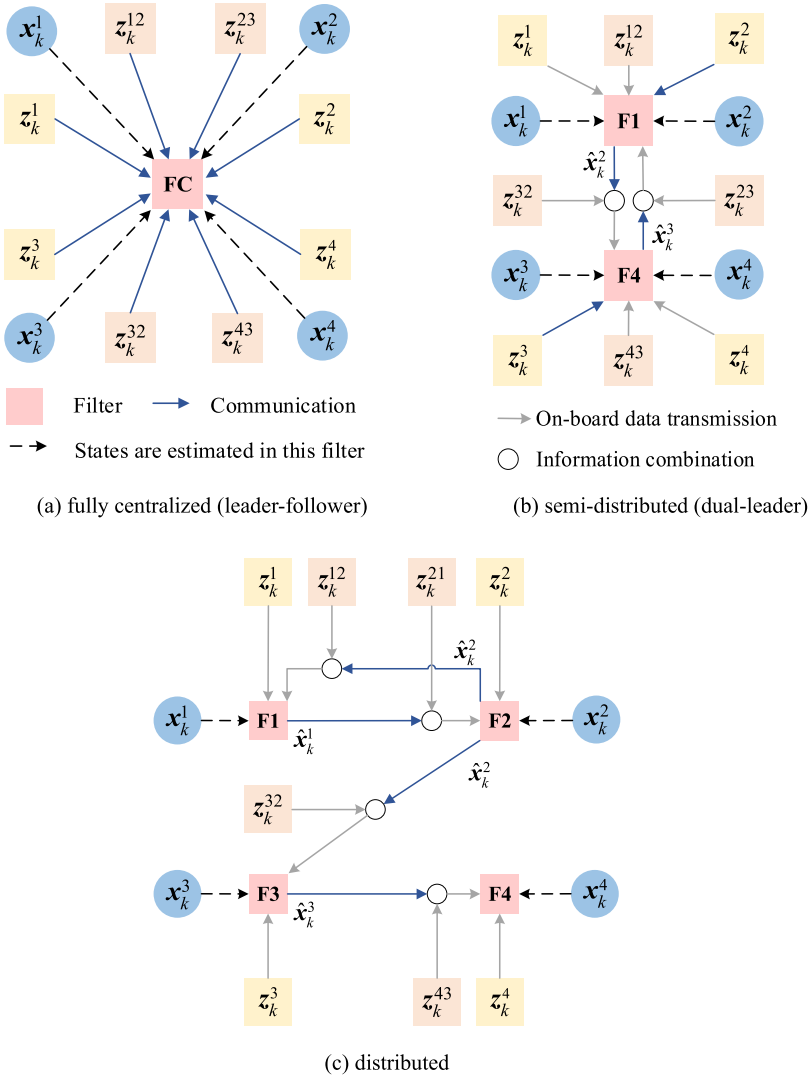


Fig. 3. Three different CN integration architectures for the above multi-agent system. Note that, the numbers after "F" indicate which agent the filter exists in. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

The CN performance differs significantly among different integration architectures. Therefore, we introduce a directed graph variable $\mathcal{G}_{\mathrm{CF}} = (\mathcal{V}_{\mathrm{CF}}, \mathcal{E}_{\mathrm{CF}})$, namely Communication & Fusion Graph (CFG), to describe the communication and fusion topology used in a CN system.

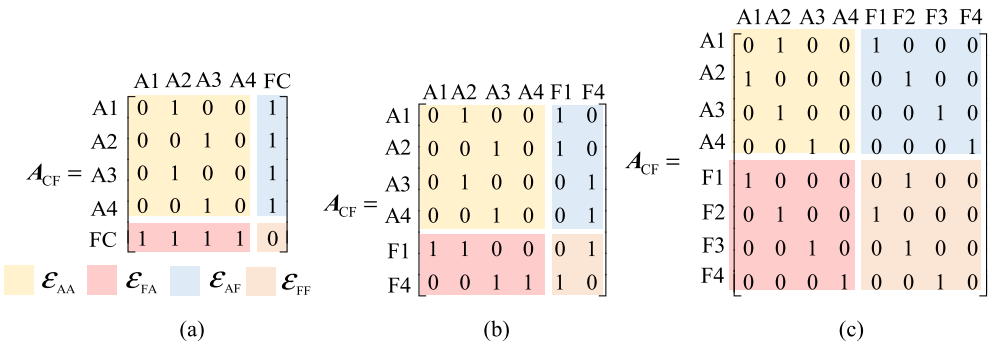


Fig. 4. The adjacency matrices of the CFG for the three different integration architectures. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

Before the illustration of $\mathcal{V}_{CF}$ and $\mathcal{E}_{CF}$, an important assumption is made as follows, under which we can establish the one-to-one mapping between $\mathcal{G}_{CF}$ and the actual integration architecture.

Assumption 1. The states of each agent, $\boldsymbol{x}_k^i$, are estimated on and only on one filter.

It is worth noting that this assumption lays the foundation for the following derivations and it is true in most cases. For the cases when this assumption is not valid, e.g., $\boldsymbol{x}_k^i$ are estimated on several filters, the derivations below should be re-conducted and the consensus among different estimates on $\boldsymbol{x}_k^i$ needs to be considered. These issues are beyond the scope of this work, and they will be addressed in the future.

The node set $\mathcal{V}_{CF}$ consists of each agent in $\mathcal{V}$ and each filter node, i.e., $\mathcal{V}_{CF} = \mathcal{V} + \mathcal{V}_F$, where $\mathcal{V}_F$ denotes the node set of filters. The edge set $\mathcal{E}_{CF} \subseteq \mathcal{V}_{CF} \times \mathcal{V}_{CF}$ can be divided into four blocks: the agent-agent edge set $\mathcal{E}_{AA}$, the agent-filter edge set $\mathcal{E}_{AF}$, the filter-agent edge set $\mathcal{E}_{FA}$, and the filter-filter edge set $\mathcal{E}_{FF}$. Their definitions are illustrated as follows. Each edge (j, i) in $\mathcal{E}_{AA}$ indicates that the filter estimating $\boldsymbol{x}_k^i$ sends $\hat{\boldsymbol{x}}_k^i$ to the filter estimating $\boldsymbol{x}_k^j$ (including the case where $\boldsymbol{x}_k^i$ and $\boldsymbol{x}_k^j$ are estimated in one filter). Each edge (j, i) in $\mathcal{E}_{AF}$ means that the state of agent i is estimated in filter j. Similarly, each edge (i, j) in $\mathcal{E}_{FA}$ represents that filter j sends the estimate $\hat{\boldsymbol{x}}_k^i$ to agent i. In this work, it is assumed that each state estimate is sent back to the corresponding agent. If there exists an edge (j, i) in $\mathcal{E}_{FF}$, then filter i sends some information (specified by $\mathcal{E}_{AA}$) to filter j.

The graph $\mathcal{G}_{CF}$ can also be expressed by its adjacency matrix, $A_{CF} = \left[a_{CF}^{ij}\right]$, where a_{CF}^{ij} is determined by: a_{CF}^{ij}=1, if the edge (j, i) exists in $\mathcal{E}_{CF}$; a_{CF}^{ij}=0, otherwise. Fig. 4 provides the adjacency matrices of each integration architecture shown above, and it also shows the four blocks for each matrix. If the RMG ($\mathcal{G}_M$) is given, the integration architecture can uniquely determine the adjacency matrix A_{CF}, and vice versa.

As shown above, CN systems are obviously more complicated than the traditional multi-sensor integration, especially because of the diversity in the integration architectures. Together with the state transition and measurement models, the two newly-defined graph variables (i.e., RMG and CFG) can clearly describe a multi-agent CN system. And multi-agent CN can be further formulated to a state estimation problem in an algebraic space after introducing the adjacency matrices of the two graphs. Finally, the formulation of a multi-agent CN problem can be summarized by Algorithm 1.

Algorithm 1 Multi-agent CN problem formulation.

Given	(1) Number of agents, n; (2) Navigation state vector $\boldsymbol{x}_k^i$ of each agent i; (3) State transition model $\boldsymbol{f}_k^i$ and absolute measurement model $\boldsymbol{h}_k^i$ for each agent i; (4) Relative measurement model $\boldsymbol{h}_k^{ij}$ for each directed combination of agents (j, i); (5) CN integration architectures (e.g., Fig. 3).
Do	(1) Determination of RMG ($\mathcal{G}_M$) and its adjacency matrix A_M; (2) Determination of CFG ($\mathcal{G}_{CF}$) and its adjacency matrix A_{CF}.
Output	(1) The description of the given multi-agent CN system in an algebraic space.

3. EKF-based CN and its performance estimation framework

3.1. EKF-based CN without filter interaction

There are usually multiple CN integration architectures for a given multi-agent system, and they can be classified into two categories: (a) the architectures without filter interaction; and (b) the architectures where filter interaction exists. Under **Assumption 1**, filter interaction refers to that a filter uses some information shared by another filter, i.e., there exists non-zero element in the adjacency matrix of $\mathcal{E}_{FF}$. Filter interaction will lead to unknown noise correlations, which will be further illustrated in Section 3.2.

This section focuses on the implementation of EKF-based cooperative state estimation for the architectures without filter interaction. Because the fully-centralized architecture is a typical example for category (a), we first use it to illustrate the detailed implementation of

the filter algorithm. Under this architecture, the filter estimates all the states of every agent and employs all the available information. Therefore, the state vector of this filter is a batch of the state vectors associated with every agent, as given below:

$$\boldsymbol{x}_k = \left[\left(\boldsymbol{x}_k^1\right)^{\mathrm{T}}, \left(\boldsymbol{x}_k^2\right)^{\mathrm{T}}, \ldots, \left(\boldsymbol{x}_k^n\right)^{\mathrm{T}} \right]^{\mathrm{T}} \tag{11}$$

Similarly, the state transition model $\boldsymbol{f}_k$ is also a batch of $\boldsymbol{f}_k^i$ for every agent i. The prediction step of this centralized filter can be expressed by:

$$\overline{\boldsymbol{x}}_k = \boldsymbol{f}_k(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k) \tag{12}$$

$$\overline{\boldsymbol{P}}_k = \boldsymbol{F}_k \hat{\boldsymbol{P}}_{k-1} \boldsymbol{F}_k^{\mathrm{T}} + \boldsymbol{Q}_k \tag{13}$$

where $\boldsymbol{u}_k$ is a batch of the input vector $\boldsymbol{u}_k^i$, $\boldsymbol{P}$ denotes the state covariance matrix, $\boldsymbol{F}_k$ is the batched state transition matrix that is a block-diagonal matrix composed of each $\boldsymbol{F}_k^i$, and the process noise covariance $\boldsymbol{Q}_k$ is also a block-diagonal matrix consisting of each $\boldsymbol{Q}_k^i$.

As mentioned above, this filter employs all the measurements available in the multi-agent system. Consequently, the measurement vector consists of both the absolute measurements and the relative measurements. The measurement equation can be expressed as:

$$\begin{bmatrix} \boldsymbol{z}_k^1 \\ \vdots \\ \boldsymbol{z}_k^n \\ \boldsymbol{z}_k^{12} \\ \vdots \\ \boldsymbol{z}_k^{n(n-1)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{h}_k^1\left(\boldsymbol{x}_k^1\right) \\ \vdots \\ \boldsymbol{h}_k^n\left(\boldsymbol{x}_k^n\right) \\ \boldsymbol{h}_k^{12}\left(\boldsymbol{x}_k^1, \boldsymbol{x}_k^2\right) \\ \vdots \\ \boldsymbol{h}_k^{n(n-1)}\left(\boldsymbol{x}_k^n, \boldsymbol{x}_k^{n-1}\right) \end{bmatrix} + \begin{bmatrix} \boldsymbol{v}_k^1 \\ \vdots \\ \boldsymbol{v}_k^n \\ \boldsymbol{v}_k^{12} \\ \vdots \\ \boldsymbol{v}_k^{n(n-1)} \end{bmatrix} \tag{14}$$

This equation can also be written in a compact form as follows:

$$\boldsymbol{z}_k = \boldsymbol{h}_k\left(\boldsymbol{x}_k\right) + \boldsymbol{v}_k \tag{15}$$

The noise characteristics of $\boldsymbol{v}_k$ are described by the measurement noise covariance matrix $\boldsymbol{R}_k$. And the linearized observation matrix associated with $\boldsymbol{h}_k$ is given by:

$$\boldsymbol{H}_k = \begin{bmatrix} \boldsymbol{H}_k^1 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ & & \vdots & & \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{H}_k^n \\ \overleftarrow{\boldsymbol{H}}_k^{12} & \overrightarrow{\boldsymbol{H}}_k^{12} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ & & \vdots & & \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \overrightarrow{\boldsymbol{H}}_k^{n(n-1)} & \overleftarrow{\boldsymbol{H}}_k^{n(n-1)} \end{bmatrix} \tag{16}$$

In Equations (14) and (16), it is assumed that the relative measurements $\boldsymbol{z}_k^{12}$ and $\boldsymbol{z}_k^{n(n-1)}$ are available in the system. The availability of each relative measurement is indicated by the RMG. And if measurement $\boldsymbol{z}_k^{ij}$ is unavailable (i.e., $a_{\mathrm{M}}^{ij} = 0$), then $\boldsymbol{z}_k^{ij}$ should not appear in these equations.

Finally, the navigation states can be updated based on the standard EKF equations shown in Equations (5)–(7).

Aside from the fully-centralized one, there are also a variety of other architectures that belong to category (a). For each of these architectures, each filter inside it estimates a part of the complete state vector shown in Equation (11) by using a subset of the measurements given in Equation (14). The state vector of each filter is specified by the CFG, and the associated measurement vector is indicated by both CFG and RMG. The implementation of each filter is completely independent and follows the same procedures as those shown above. For the sake of brevity, the detailed implementation is not repeated here.

3.2. Bounding the state covariance for EKF-based CN with filter interaction

Different from the architectures in category (a), there exists filter interaction in the architectures belonging to category (b). The interaction between filters will lead to complex noise correlations, and the correlations are usually unknown or very difficult to accurately compute. In the CN field, coping with these unknown correlations in EKF is an open research topic. There will be various approaches proposed in the future, and each of them may show different performance from the others. Therefore, this section focuses on offering the upper and lower bounds on the state covariance instead of providing a deterministic solution. Please note, in this work, a matrix $\boldsymbol{A}$ is an upper bound of $\boldsymbol{B}$ if and only if $\boldsymbol{A}-\boldsymbol{B}$ is positive semidefinite, and inversely, $\boldsymbol{B}$ is a lower bound of $\boldsymbol{A}$.

The unknown correlation caused by filter interaction is also widely known as the data incest problem [41]. This issue is due to the re-use of the same information in the fusion process. In the architectures with filter interaction, there are usually iterations in each update step. Fig. 5 gives an example of an integration architecture with filter interaction and shows the process of iterative state update. This example can be used to explain the mechanism of the data incest problem.

As shown in Fig. 5, the estimates on $\boldsymbol{x}_k^1$ and $\boldsymbol{x}_k^2$ are coupled with each other, i.e., the estimate on $\boldsymbol{x}_k^2$ depends on $\hat{\boldsymbol{x}}_k^1$, and in turn the determination of $\hat{\boldsymbol{x}}_k^1$ also relies on $\hat{\boldsymbol{x}}_k^2$. Even if there is only 1-step iteration (i.e., steps 1 and 2) at epoch k, $\hat{\boldsymbol{x}}_{k+1}^1$ will still be coupled with

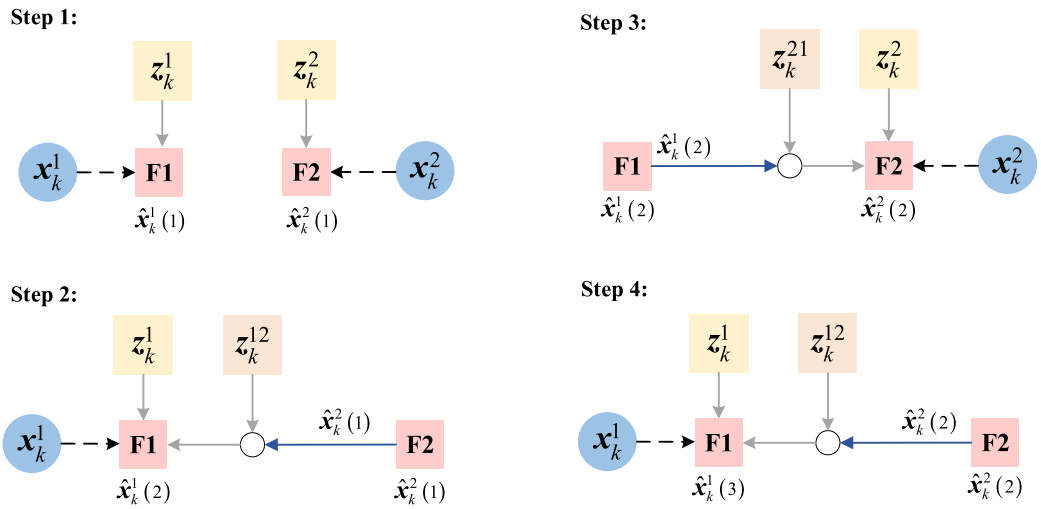


Fig. 5. The process of iterative state update in an integration architecture with filter interaction, where the number in parentheses is used to distinguish different estimates.

$\hat{\boldsymbol{x}}_{k+1}^2$ because their determination uses common information (i.e., the measurements related with $\hat{\boldsymbol{x}}_k^2$). It becomes very difficult to calculate this correlation (between the information from neighbors and the local one) after several filter epochs. And if the unknown correlation is ignored in the fusion process, it may lead to over-convergence problems, i.e., the estimation is overly confident, and even the divergence.

Based on the fusion process shown above, the upper bound on the state covariance (i.e., $\hat{\boldsymbol{P}}_k$) is derived as follows. Let the *adjoint filter* of a filter in CN be the one that discards this filter's relative measurements. This kind of measurements is called *cross-filter relative measurements*, and the others are called *inner-filter measurements*. For example, the adjoint filter of F1 in Fig. 5 is a standard EKF that estimates $\boldsymbol{x}_k^1$ using $\boldsymbol{z}_k^1$; and the adjoint filter of F1 in Fig. 3(b) is a centralized filter that jointly estimates $\boldsymbol{x}_k^1$ and $\boldsymbol{x}_k^2$ by employing $\boldsymbol{z}_k^1$, $\boldsymbol{z}_k^2$, and $\boldsymbol{z}_k^{12}$. The upper bound on $\hat{\boldsymbol{P}}_k$ is computed by completely ignoring the effect of iterative state update. Detailed illustrations and derivations are given below.

The upper bound on ${}_i\hat{\boldsymbol{P}}_k$ for filter i, ${}_i^{\mathrm{u}}\hat{\boldsymbol{P}}_k$, is calculated by:

$$ {}_i^{\mathrm{u}}\hat{\boldsymbol{P}}_k = \left(\boldsymbol{I} - {}_i^{\mathrm{u}}\boldsymbol{K}_k \cdot {}_i^{\mathrm{u}}\boldsymbol{H}_k\right) \cdot {}_i^{\mathrm{a}}\overline{\boldsymbol{P}}_k \tag{17} $$

where ${}_i^{\mathrm{a}}\overline{\boldsymbol{P}}_k$ is the *a priori* state covariance matrix in the adjoint filter, and ${}_i^{\mathrm{u}}\boldsymbol{H}_k$ and ${}_i^{\mathrm{u}}\boldsymbol{K}_k$ are computed by Equations (18) and (19), respectively.

$$ {}_i^{\mathrm{u}}\boldsymbol{H}_k = \begin{bmatrix} \boldsymbol{H}_k^{i_1} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ & & \vdots & & \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{H}_k^{i_m} \\ \overleftarrow{\boldsymbol{H}}_k^{i_1 i_2} & \overrightarrow{\boldsymbol{H}}_k^{i_1 i_2} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ & & \vdots & & \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \overrightarrow{\boldsymbol{H}}_k^{i_m i_{m-1}} & \overleftarrow{\boldsymbol{H}}_k^{i_m i_{m-1}} \\ \overleftarrow{\boldsymbol{H}}_k^{i_1 j_1} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ & & \vdots & & \end{bmatrix} \tag{18} $$

$$ {}_i^{\mathrm{u}}\boldsymbol{K}_k = {}_i^{\mathrm{a}}\overline{\boldsymbol{P}}_k {}_i^{\mathrm{u}}\boldsymbol{H}_k^{\mathrm{T}} \cdot \left({}_i^{\mathrm{u}}\boldsymbol{H}_k {}_i^{\mathrm{a}}\overline{\boldsymbol{P}}_k {}_i^{\mathrm{u}}\boldsymbol{H}_k^{\mathrm{T}} + {}_i^{\mathrm{u}}\boldsymbol{R}_k\right)^{-1} \tag{19} $$

in which $i_1, i_2, \ldots, i_m$ are the agents whose states are estimated in this filter; the existence of $\overleftarrow{\boldsymbol{H}}_k^{i_1 j_1}$ in (18) indicates that filter j estimates the states of agent j_1 and sends its information to filter i, i.e., j_1 is a neighbor of i_1 with relative measurement $\boldsymbol{z}_k^{i_1 j_1}$ between them; and the measurement noise covariance matrix ${}_i^{\mathrm{u}}\boldsymbol{R}_k$ is given by:

$$ {}_i^{\mathrm{u}}\boldsymbol{R}_k = \mathrm{blkdiag}\left(\left[\boldsymbol{R}_k^{i_1}, \ldots, \boldsymbol{R}_k^{i_m}, \boldsymbol{R}_k^{i_1 i_2}, \ldots, \boldsymbol{R}_k^{i_m i_{m-1}}, \tilde{\boldsymbol{R}}_k^{i_1 j_1}, \ldots\right]\right) \tag{20} $$

where blkdiag() is a function to generate the block-diagonal matrix using the matrices in parentheses; the matrices $\boldsymbol{R}_k^{i_1}$, $\boldsymbol{R}_k^{i_2}$, $\ldots$, $\boldsymbol{R}_k^{i_m i_{m-1}}$ denote the covariances of $\boldsymbol{v}_k^{i_1}$, $\boldsymbol{v}_k^{i_2}$, $\ldots$, $\boldsymbol{v}_k^{i_m i_{m-1}}$, respectively; and the covariance matrix $\tilde{\boldsymbol{R}}_k^{i_1 j_1}$ associated with the measurement $\boldsymbol{z}_k^{i_1 j_1}$ is computed as:

$$ \tilde{\boldsymbol{R}}_k^{i_1 j_1} = \boldsymbol{R}_k^{i_1 j_1} + \left(\overrightarrow{\boldsymbol{H}}_k^{i_1 j_1}\right)^{\mathrm{T}} {}_j^{\mathrm{a}}\hat{\boldsymbol{P}}_k^1 \overrightarrow{\boldsymbol{H}}_k^{i_1 j_1} \tag{21} $$

where $\boldsymbol{R}_k^{i_1 j_1}$ is the covariance matrix of $\boldsymbol{v}_k^{i_1 j_1}$; and ${}_j^{\mathrm{a}}\hat{\boldsymbol{P}}_k^1$ is the covariance associated with $\boldsymbol{x}_k^{j_1}$ in the *posterior* state covariance matrix of filter j, ${}_j^{\mathrm{a}}\hat{\boldsymbol{P}}_k$. The second term in the right-hand side of (21) reflects the effect of the noise of ${}^{\mathrm{a}}\hat{\boldsymbol{x}}_k^{j_1}$ (the estimate of $\boldsymbol{x}_k^{j_1}$ in the adjacency filter) on the estimate of $\boldsymbol{x}_k^{i_1}$. This equation can be derived by linearizing the original measurement equation of $\boldsymbol{z}_k^{i_1 j_1}$:

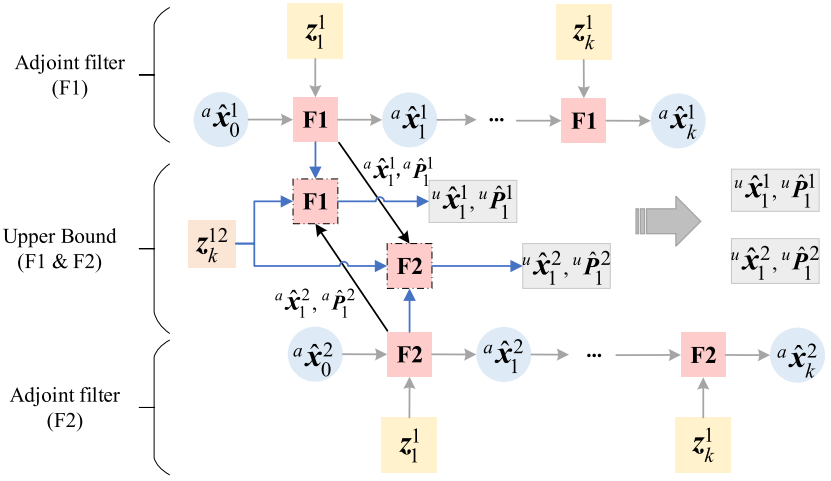


Fig. 6. Graphic illustration of the adjoint filters and the calculation of the upper bounds.

$$\boldsymbol{z}_k^{i_1 j_1} = \boldsymbol{h}_k^{i_1 j_1}\left({}^{\mathrm{a}}\overline{\boldsymbol{x}}_k^{i_1}, {}^{\mathrm{a}}\hat{\boldsymbol{x}}_k^{j_1}\right) + \overleftarrow{\boldsymbol{H}}_k^{i_1 j_1}\left(\boldsymbol{x}_k^{i_1} - {}^{\mathrm{a}}\overline{\boldsymbol{x}}_k^{i_1}\right) + \overrightarrow{\boldsymbol{H}}_k^{i_1 j_1}\left(\boldsymbol{x}_k^{j_1} - {}^{\mathrm{a}}\hat{\boldsymbol{x}}_k^{j_1}\right) + \boldsymbol{\upsilon}_k^{i_1 j_1} \tag{22}$$

where $\overleftarrow{\boldsymbol{H}}_k^{i_1 j_1} = \left(\partial \boldsymbol{z}_k^{i_1 j_1}/\partial \boldsymbol{x}_k^{i_1}\right)\Big|_{\left({}^{\mathrm{a}}\overline{\boldsymbol{x}}_k^{i_1},\ {}^{\mathrm{a}}\hat{\boldsymbol{x}}_k^{j_1}\right)}$, $\overrightarrow{\boldsymbol{H}}_k^{i_1 j_1} = \left(\partial \boldsymbol{z}_k^{i_1 j_1}/\partial \boldsymbol{x}_k^{j_1}\right)\Big|_{\left({}^{\mathrm{a}}\overline{\boldsymbol{x}}_k^{i_1},\ {}^{\mathrm{a}}\hat{\boldsymbol{x}}_k^{j_1}\right)}$, and $\left(\boldsymbol{x}_k^{j_1} - {}^{\mathrm{a}}\hat{\boldsymbol{x}}_k^{j_1}\right)$ is the noise of ${}^{\mathrm{a}}\hat{\boldsymbol{x}}_k^{j_1}$.

Using the system in Fig. 5 as an example, Fig. 6 shows the generation of ${}^{\mathrm{u}}_{1}\hat{\boldsymbol{P}}_k$ and ${}^{\mathrm{u}}_{2}\hat{\boldsymbol{P}}_k$ (i.e., the upper bounds on the state covariance for filter 1 and filter 2, respectively) to further illustrate the approach above. As shown in this figure, the calculation of the upper bounds takes use of the corresponding adjoint filters, and the adjoint filters run in parallel without any interaction. Together with these adjoint filters, the cross-filter relative measurement models offer the necessary inputs to the calculation of the upper bounds on $\hat{\boldsymbol{P}}_k$.

The proof for ${}^{\mathrm{u}}_{i}\hat{\boldsymbol{P}}_k \geq {}_{i}\hat{\boldsymbol{P}}_k$ is given as follows. Without loss of generality, we use the case in Fig. 5 for the proof. With one or more iterations, filter 2 will receive a more accurate $\hat{\boldsymbol{x}}_k^1$ and a smaller $\hat{\boldsymbol{P}}_k^1$ from filter 1 than those from the adjoint filter of filter 1. And when filter 2 uses the relative measurement $\boldsymbol{z}_k^{12}$, a more accurate $\hat{\boldsymbol{x}}_k^1$ will result in a more accurate $\hat{\boldsymbol{x}}_k^2$. Consequently, filter 2 will obtain a more accurate $\hat{\boldsymbol{x}}_k^2$ and a smaller $\hat{\boldsymbol{P}}_k^2$ after iterations than without any iteration (i.e., filter 2 directly uses the information from the adjoint filter of filter 1). Because the upper bound, ${}^{\mathrm{u}}_{2}\hat{\boldsymbol{P}}_k$, is calculated by completely ignoring the effect of iterations, it will be larger than the actual covariance matrix after iterations, ${}_{2}\hat{\boldsymbol{P}}_k$. This is also true for filter 1, and this conclusion can be extended to the general case.

Then, the lower bound on ${}_{i}\hat{\boldsymbol{P}}_k$ for filter i, ${}^{\mathrm{l}}_{i}\hat{\boldsymbol{P}}_k$, is derived by directly ignoring the unknown correlations caused by filter interaction. The unknown correlations essentially come from the data incest problem, and ignoring them can lead to an overly-confident state estimate, i.e., the estimated state covariance is smaller than the true one. Therefore, ${}^{\mathrm{l}}_{i}\hat{\boldsymbol{P}}_k$ can serve as a lower bound for ${}_{i}\hat{\boldsymbol{P}}_k$.

Let the *ideal filter* of a filter in CN be the one that ignores the unknown correlations. The lower bound ${}^{\mathrm{l}}_{i}\hat{\boldsymbol{P}}_k$ is recursively calculated based on this filter. Specifically, ${}^{\mathrm{l}}_{i}\hat{\boldsymbol{P}}_k$ is calculated by the following:

$${}^{\mathrm{l}}_{i}\hat{\boldsymbol{P}}_k = \left(\boldsymbol{I} - {}^{\mathrm{l}}_{i}\boldsymbol{K}_k \cdot {}^{\mathrm{l}}_{i}\boldsymbol{H}_k\right) \cdot {}^{\mathrm{l}}_{i}\overline{\boldsymbol{P}}_k \tag{23}$$

where ${}^{\mathrm{l}}_{i}\overline{\boldsymbol{P}}_k$ denotes the predicted state covariance, ${}^{\mathrm{l}}_{i}\boldsymbol{H}_k$ is equal to ${}^{\mathrm{u}}_{i}\boldsymbol{H}_k$ given in (18), and ${}^{\mathrm{u}}_{i}\boldsymbol{K}_k$ are computed as:

$${}^{\mathrm{l}}_{i}\boldsymbol{K}_k = {}^{\mathrm{l}}_{i}\overline{\boldsymbol{P}}_k {}^{\mathrm{l}}_{i}\boldsymbol{H}_k^{\mathrm{T}} \cdot \left({}^{\mathrm{l}}_{i}\boldsymbol{H}_k {}^{\mathrm{l}}_{i}\overline{\boldsymbol{P}}_k {}^{\mathrm{l}}_{i}\boldsymbol{H}_k^{\mathrm{T}} + {}^{\mathrm{l}}_{i}\boldsymbol{R}_k\right)^{-1} \tag{24}$$

where ${}^{\mathrm{l}}_{i}\boldsymbol{R}_k$ is the measurement noise covariance used in the ideal filter. The matrix ${}^{\mathrm{l}}_{i}\boldsymbol{R}_k$ has the same form as ${}^{\mathrm{u}}_{i}\boldsymbol{R}_k$ in (20) with a different $\tilde{\boldsymbol{R}}_k^{i_1 j_1}$. In ${}^{\mathrm{l}}_{i}\boldsymbol{R}_k$, $\tilde{\boldsymbol{R}}_k^{i_1 j_1}$ is calculated by

$$\tilde{\boldsymbol{R}}_k^{i_1 j_1} = \boldsymbol{R}_k^{i_1 j_1} + \overrightarrow{\boldsymbol{H}}_k^{i_1 j_1}\, {}^{\mathrm{l}}_{j}\hat{\boldsymbol{P}}_k^1 \left(\overrightarrow{\boldsymbol{H}}_k^{i_1 j_1}\right)^{\mathrm{T}} \tag{25}$$

with $\overrightarrow{\boldsymbol{H}}_k^{i_1 j_1} = \left(\partial \boldsymbol{z}_k^{i_1 j_1}/\partial \boldsymbol{x}_k^{j_1}\right)\Big|_{\left({}^{\mathrm{l}}\overline{\boldsymbol{x}}_k^{i_1},\ {}^{\mathrm{l}}\hat{\boldsymbol{x}}_k^{j_1}\right)}$ where ${}^{\mathrm{l}}\overline{\boldsymbol{x}}_k^{i_1}$ is the prediction of $\boldsymbol{x}_k^{i_1}$ and ${}^{\mathrm{l}}\hat{\boldsymbol{x}}_k^{j_1}$ is the *posterior* estimate on $\boldsymbol{x}_k^{j_1}$ in the associated ideal filters. Please note, different number of iterations will result in different $\tilde{\boldsymbol{R}}_k^{i_1 j_1}$ because the value of ${}^{\mathrm{l}}_{j}\hat{\boldsymbol{P}}_k^1$ will change with the iterations.

For filter i, the use of the cross-filter relative measurements (e.g., $\boldsymbol{z}_k^{i_1 j_1}$) will lead to unknown correlations between the information from the neighbors (e.g., filter j) and the local one. In the approach above, the unknown correlations are completely ignored by (a) letting ${}^{\mathrm{l}}_{i}\boldsymbol{R}_k$ be a block-diagonal matrix and (b) simply using the standard EKF equations. Therefore, the obtained ${}^{\mathrm{l}}_{i}\hat{\boldsymbol{P}}_k$ is a lower bound for the achievable one.

For some architectures, the approach above may generate a lower bound that is even smaller than the result under the fully-centralized architecture. This means the lower bound may be overly loose. Therefore, the lower bound can be further optimized to be tighter by introducing the following constraint: if architecture b is generated by combining two or more filters in architecture a into a centralized filter, then the covariance lower bound associated with an agent under architecture a should not be smaller than that under architecture b. For example, the lower bound on the covariance of $\boldsymbol{x}_k^i$ ($i = 1, 2, 3, 4$) under architecture (b) in Fig. 3 should not be larger than the results

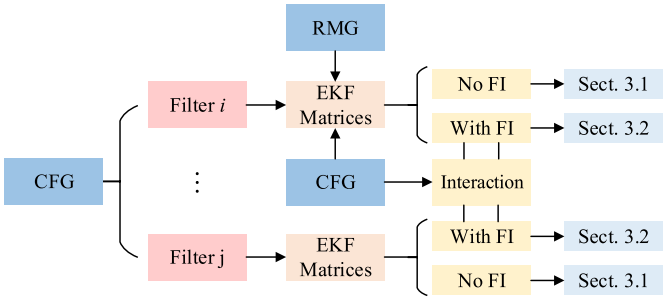


Fig. 7. Overall CN performance estimation framework (FI refers to "filter interaction").

under the fully-distributed architecture. This is justified because combining several filters into a centralized filter can certainly result in improved estimation accuracy than the original case.

3.3. Relationship between CN performance and CN integration architecture

Sections 3.1 and 3.2 present the methods to estimate the CN performance (i.e., state covariance) for a given integration architecture, and this section will further establish the relationship between the estimated navigation performance and the CN integration architecture using RMG and CFG. Fig. 7 briefly demonstrates the overall CN performance estimation framework, and more detailed illustrations are given below.

For a filter $i \in \mathcal{V}_{\mathrm{F}}$, we need to determine all the necessary matrices for the EKF implementation. These matrices are determined based on the given multi-agent system, RMG and CFG. Equations (11)–(16) give the matrices under the fully-centralized architecture, and we re-label these variables as ${}_{\mathrm{fc}}\boldsymbol{F}_k$, ${}_{\mathrm{fc}}\boldsymbol{Q}_k$, ${}_{\mathrm{fc}}\boldsymbol{H}_k$, ${}_{\mathrm{fc}}\boldsymbol{R}_k$, respectively. The construction of ${}_{\mathrm{fc}}\boldsymbol{H}_k$ and ${}_{\mathrm{fc}}\boldsymbol{R}_k$ has indeed taken the RMG into account. To explicitly illustrate the effect of RMG, we first define the full measurement matrix ${}_{\mathrm{f}}\boldsymbol{H}_k$ in (26) and the full measurement noise covariance ${}_{\mathrm{f}}\boldsymbol{R}_k$ in (27).

$$
{}_{\mathrm{f}}\boldsymbol{H}_k = \begin{bmatrix} {}_{\mathrm{f}}\boldsymbol{H}_k^{\mathrm{A}} \\ {}_{\mathrm{f}}\boldsymbol{H}_k^{\mathrm{R}} \end{bmatrix} = \overbrace{\begin{bmatrix} \boldsymbol{H}_k^1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ & & \vdots & & \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{H}_k^n \\ \overleftarrow{\boldsymbol{H}}_k^{12} & \overrightarrow{\boldsymbol{H}}_k^{12} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ & & \vdots & & \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \overrightarrow{\boldsymbol{H}}_k^{n(n-1)} & \overleftarrow{\boldsymbol{H}}_k^{n(n-1)} \end{bmatrix}}^{n^2 \times n \text{ blocks}} \tag{26}
$$

$$
{}_{\mathrm{f}}\boldsymbol{R}_k = \begin{bmatrix} {}_{\mathrm{f}}\boldsymbol{R}_k^{\mathrm{A}} & \\ & {}_{\mathrm{f}}\boldsymbol{R}_k^{\mathrm{R}} \end{bmatrix} = \overbrace{\begin{bmatrix} \boldsymbol{R}_k^1 & & & & \\ & \ddots & & & \\ & & \boldsymbol{R}_k^n & & \\ & & & \boldsymbol{R}_k^{12} & \\ & & & & \ddots \\ & & & & \boldsymbol{R}_k^{n(n-1)} \end{bmatrix}}^{n^2 \times n^2 \text{ blocks}} \tag{27}
$$

where the superscripts "A" and "R" indicate absolute and relative measurements, respectively; ${}_{\mathrm{f}}\boldsymbol{H}_k^{\mathrm{R}}$ and ${}_{\mathrm{f}}\boldsymbol{R}_k^{\mathrm{R}}$ include all the possible inter-agent relative measurements.

Then we define an RMG-based selection matrix ${}_{\mathrm{fc}}\boldsymbol{S}_k^{\mathrm{RM}}$ in (28) to indicate which relative measurements are available in the multi-agent system.

$$
{}_{\mathrm{fc}}\boldsymbol{S}_k^{\mathrm{RM}} = \overbrace{\begin{bmatrix} a_{\mathrm{M}}^{12} \cdot \boldsymbol{I}_{12} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & a_{\mathrm{M}}^{n(n-1)} \cdot \boldsymbol{I}_{n(n-1)} \end{bmatrix}}^{\text{remove all-zero rows}} \tag{28}
$$

where the size of the identity matrix $\boldsymbol{I}_{ij}$ is equal to the number of rows in $\boldsymbol{H}_k^{ij}$. Based on ${}_{\mathrm{fc}}\boldsymbol{S}_k^{\mathrm{RM}}$, we re-write ${}_{\mathrm{fc}}\boldsymbol{H}_k$ and ${}_{\mathrm{fc}}\boldsymbol{R}_k$ as follows:

$$
{}_{\mathrm{fc}}\boldsymbol{H}_k = \begin{bmatrix} {}_{\mathrm{f}}\boldsymbol{H}_k^{\mathrm{A}} \\ {}_{\mathrm{fc}}\boldsymbol{S}_k^{\mathrm{RM}} \cdot {}_{\mathrm{f}}\boldsymbol{H}_k^{\mathrm{R}} \end{bmatrix} \tag{29}
$$

$$
{}_{\mathrm{fc}}\boldsymbol{R}_k = \begin{bmatrix} {}_{\mathrm{f}}\boldsymbol{R}_k^{\mathrm{A}} \\ {}_{\mathrm{fc}}\boldsymbol{S}_k^{\mathrm{RM}} \cdot {}_{\mathrm{f}}\boldsymbol{R}_k^{\mathrm{R}} \cdot \left({}_{\mathrm{fc}}\boldsymbol{S}_k^{\mathrm{RM}}\right)^{\mathrm{T}} \end{bmatrix} \tag{30}
$$

Based on the matrices above, the corresponding four matrices for filter i are then constructed by employing the CFG. As shown below, we define a CFG-based selection matrix ${}_i\boldsymbol{S}_k^{\mathrm{A}}$ to describe which agents' states are estimated in filter i.

$$ {}_i\boldsymbol{S}_k^{\mathrm{A}} = \overbrace{\begin{bmatrix} a_{\mathrm{FA}}^{i1}\cdot\boldsymbol{I}_1 & & \boldsymbol{0} \\ & \ddots & \\ \boldsymbol{0} & & a_{\mathrm{FA}}^{in}\cdot\boldsymbol{I}_n \end{bmatrix}}^{\text{remove all}-\text{zero rows}} \tag{31} $$

where a_{FA} is the adjacency matrix of $\mathcal{E}_{\mathrm{FA}}$ and $a_{\mathrm{FA}}^{ij}=1$ indicates that $\boldsymbol{x}_k^j$ is estimated in filter i. The size of $\boldsymbol{I}_j$ equals to the length of $\boldsymbol{x}_k^j$. Based on ${}_i\boldsymbol{S}_k^{\mathrm{A}}$, the state transition matrix ${}_i\boldsymbol{F}_k$ and the process noise covariance ${}_i\boldsymbol{Q}_k$ for filter i are given by:

$$ {}_i\boldsymbol{F}_k = {}_i\boldsymbol{S}_{\boldsymbol{k}}^{\mathrm{A}}\cdot{}_{\mathrm{fc}}\boldsymbol{F}_k\cdot\left({}_i\boldsymbol{S}_{\boldsymbol{k}}^{\mathrm{A}}\right)^{\mathrm{T}} \tag{32} $$

$$ {}_i\boldsymbol{Q}_k = {}_i\boldsymbol{S}_{\boldsymbol{k}}^{\mathrm{A}}\cdot{}_{\mathrm{fc}}\boldsymbol{Q}_k\cdot\left({}_i\boldsymbol{S}_{\boldsymbol{k}}^{\mathrm{A}}\right)^{\mathrm{T}} \tag{33} $$

Based on the CFG, we also define a selection matrix ${}_i\boldsymbol{S}_k^{\mathrm{R}}$ in (34) to indicate which relative measurements are used in this filter as inner-filter measurements.

$$ {}_i\boldsymbol{S}_k^{\mathrm{R}} = \overbrace{\begin{bmatrix} a_{\mathrm{FA}}^{i1}a_{\mathrm{FA}}^{i2}a_{\mathrm{AA}}^{12}\cdot\boldsymbol{I}_{12} & & \boldsymbol{0} \\ & \ddots & \\ \boldsymbol{0} & & a_{\mathrm{FA}}^{in}a_{\mathrm{FA}}^{i(n-1)}a_{\mathrm{AA}}^{n(n-1)}\cdot\boldsymbol{I}_{n(n-1)} \end{bmatrix}}^{\text{remove all}-\text{zero rows}} \tag{34} $$

where a_{AA} is the adjacency matrix of $\mathcal{E}_{\mathrm{AA}}$ and $a_{\mathrm{AA}}^{mj}=1$ indicates that the relative measurement $\boldsymbol{z}_k^{mj}$ is used. $\boldsymbol{I}_{mj}$ will appear in (34) if and only if $a_{\mathrm{M}}^{mj}=1$, i.e., $\boldsymbol{z}_k^{mj}$ is available in the system. Based on ${}_i\boldsymbol{S}_k^{\mathrm{R}}$, the measurement matrix ${}_i\boldsymbol{H}_k$ and the measurement noise covariance ${}_i\boldsymbol{R}_k$ are calculated by:

$$ {}_i\boldsymbol{H}_k = \begin{bmatrix} {}_i\boldsymbol{S}_k^{\mathrm{A}}\cdot{}_{\mathrm{fc}}\boldsymbol{H}_k^{\mathrm{A}} \\ {}_i\boldsymbol{S}_k^{\mathrm{R}}\cdot{}_{\mathrm{fc}}\boldsymbol{H}_k^{\mathrm{R}} \end{bmatrix} \tag{35} $$

$$ {}_i\boldsymbol{R}_k = \begin{bmatrix} {}_i\boldsymbol{S}_k^{\mathrm{A}}\cdot{}_{\mathrm{fc}}\boldsymbol{R}_k^{\mathrm{A}}\cdot\left({}_i\boldsymbol{S}_k^{\mathrm{A}}\right)^{\mathrm{T}} & \boldsymbol{0} \\ \boldsymbol{0} & {}_i\boldsymbol{S}_k^{\mathrm{R}}\cdot{}_{\mathrm{fc}}\boldsymbol{R}_k^{\mathrm{R}}\cdot\left({}_i\boldsymbol{S}_k^{\mathrm{R}}\right)^{\mathrm{T}} \end{bmatrix} \tag{36} $$

If filter i does not interact with any other filter, the state covariance ${}_i\hat{\boldsymbol{P}}_k$ can be recursively calculated using ${}_i\boldsymbol{F}_k$, ${}_i\boldsymbol{Q}_k$, ${}_i\boldsymbol{H}_k$, and ${}_i\boldsymbol{R}_k$ and following the approach in Section 3.1. Otherwise, we define another CFG-based selection matrix ${}_i\boldsymbol{S}_k^{\mathrm{FF}}$ to account for filter interaction:

$$ {}_i\boldsymbol{S}_k^{\mathrm{FF}} = \overbrace{\begin{bmatrix} a_{\mathrm{AA}}^{12}\cdot a_{\mathrm{FA}}^{i1}\cdot\sum\limits_{j\neq i}\left(a_{\mathrm{FA}}^{j2}a_{\mathrm{FF}}^{ij}\right)\cdot\boldsymbol{I}_{12} & & \boldsymbol{0} \\ & \ddots & \\ \boldsymbol{0} & & a_{\mathrm{AA}}^{n(n-1)}\cdot a_{\mathrm{FA}}^{in}\cdot\sum\limits_{j\neq i}\left(a_{\mathrm{FA}}^{j(n-1)}a_{\mathrm{FF}}^{ij}\right)\cdot\boldsymbol{I}_{n(n-1)} \end{bmatrix}}^{\text{remove all}-\text{zero rows}} \tag{37} $$

where a_{FF} is the adjacency matrix of $\mathcal{E}_{\mathrm{FF}}$ and $a_{\mathrm{FF}}^{ij}=1$ indicates that filter j sends some information to filter i. For example, if we have $a_{\mathrm{AA}}^{12}a_{\mathrm{FA}}^{i1}a_{\mathrm{FA}}^{j2}a_{\mathrm{FF}}^{ij}=1$ $(j\neq i)$, then: (a) $\boldsymbol{x}_k^1$ is estimated in filter i; (b) $\boldsymbol{x}_k^2$ is estimated in filter j; (c) $\boldsymbol{z}_k^{12}$ is used in filter i as a cross-filter relative measurement; and (d) filter j sends $\hat{\boldsymbol{P}}_k^2$ (or ${}^{\mathrm{a}}\hat{\boldsymbol{P}}_k^2$, ${}^{\mathrm{l}}\hat{\boldsymbol{P}}_k^2$) to filter i. Similarly, $\boldsymbol{I}_{mt}$ will appear in (37) if and only if $a_{\mathrm{M}}^{mt}=1$, i.e., $\boldsymbol{z}_k^{mt}$ is available in the system.

Considering filter interaction, we derive the measurement matrix ${}_i^*\boldsymbol{H}_k$ and the measurement noise covariance ${}_i^*\boldsymbol{R}_k$ as follows, where "*" can be "u" representing "upper bound" or "l" indicating "lower bound".

$$ {}_i^{\mathrm{u}}\boldsymbol{H}_k = {}_i^{\mathrm{l}}\boldsymbol{H}_k = \begin{bmatrix} {}_i\boldsymbol{H}_k \\ {}_i\boldsymbol{S}_k^{\mathrm{FF}}\cdot{}_{\mathrm{fc}}\boldsymbol{H}_k^{\mathrm{R}} \end{bmatrix} \tag{38} $$

$$ {}_i^*\boldsymbol{R}_k = \begin{bmatrix} {}_i\boldsymbol{R}_k & \\ & {}_i\boldsymbol{S}_k^{\mathrm{FF}}\cdot{}^*\tilde{\boldsymbol{R}}_k^{\mathrm{FF}}\cdot\left({}_i\boldsymbol{S}_k^{\mathrm{FF}}\right)^{\mathrm{T}} \end{bmatrix} \tag{39} $$

where we have:

$$ {}^{\mathrm{u}}\tilde{\boldsymbol{R}}_k^{\mathrm{FF}} = \begin{bmatrix} \boldsymbol{R}_k^{12} & & \\ & \ddots & \\ & & \boldsymbol{R}_k^{n(n-1)} \end{bmatrix} + \begin{bmatrix} \overrightarrow{\boldsymbol{H}}_k^{12}\,{}^{\mathrm{a}}\hat{\boldsymbol{P}}_k^2\left(\overrightarrow{\boldsymbol{H}}_k^{12}\right)^{\mathrm{T}} & & \\ & \ddots & \\ & & \overrightarrow{\boldsymbol{H}}_k^{n(n-1)}\,{}^{\mathrm{a}}\hat{\boldsymbol{P}}_k^{(n-1)}\left(\overrightarrow{\boldsymbol{H}}_k^{n(n-1)}\right)^{\mathrm{T}} \end{bmatrix} \tag{40} $$

Table 1
Basic simulation settings.

Parameters	Values (SI units)
Number of agents	4
Agent dynamic model	Constant-Acceleration (CA) model [42], 2-dimension (X and Y)
Navigation state	$\boldsymbol{x}^i = [p_x^i;\ v_x^i;\ a_x^i;\ p_y^i;\ v_y^i;\ a_y^i]$, where the variables p, v, a, and i denote position, velocity, acceleration, and agent number, respectively.
Sampling interval, t	0.1, if not specified.
Simulation time	10
Initial state (truth)	$[p_x^1, p_x^2, p_x^3, p_x^4] = [p_y^1, p_y^2, p_y^3, p_y^4] = [0, 10, 20, 30]$ $v_x^i = v_y^i = 1$, $i = 1, 2, 3, 4$ $a_x^i = 0.5$, $a_y^i = 0.2$, $i = 1, 2, 3, 4$
Initial state error covariance, P_0	The same for agents 1-4: diag($[10^2, 3^2, 0.5^2, 10^2, 3^2, 0.5^2]$)
Acceleration noise variance, q_a	$q_{ax} = q_{ay} = [0.1^2, 0.1^2, 0.1^2, 0.1^2]$
Process noise covariance, Q	Calculated by: $Q = \text{blkdiag}(Q_x, Q_y)$ $Q_* = \begin{bmatrix} t^5/20 & t^4/8 & t^3/6 \\ t^4/8 & t^3/3 & t^2/2 \\ t^3/6 & t^2/2 & t \end{bmatrix} q_{a*}$, $* = x$ or y [42]

Table 2
Measurements and their error covariances.

Measurement type	Measurements	Error covariances (SI units)
Absolute measurements	$z^1 = [p_x^1;\ p_y^1]$	diag($[1^2, 1^2]$)
	$z^2 = [v_x^2;\ v_y^2]$	diag($[0.5^2, 0.5^2]$)
	$z^3 = [p_x^3]$	1^2
	$z^4 = [v_y^4]$	1^2
Relative measurements	$z^{12} = z^{21} = [p_x^2 - p_x^1]$	0.5^2
	$z^{23} = [p_x^3 - p_x^2;\ p_y^3 - p_y^2]$	diag($[0.5^2, 0.5^2]$)
	$z^{32} = [v_x^2 - v_x^3;\ v_y^2 - v_y^3]$	diag($[0.2^2, 0.2^2]$)
	$z^{34} = z^{43} = [p_x^3 - p_x^4;\ p_y^3 - p_y^4]$	diag($[0.5^2, 0.5^2]$)

$$ {}^{1}\tilde{\boldsymbol{R}}_k^{\mathrm{FF}} = \begin{bmatrix} \boldsymbol{R}_k^{12} & & \\ & \ddots & \\ & & \boldsymbol{R}_k^{n(n-1)} \end{bmatrix} + \begin{bmatrix} \overrightarrow{\boldsymbol{H}}_k^{12}\, {}_{1}\hat{\boldsymbol{P}}_k^{2} \left(\overrightarrow{\boldsymbol{H}}_k^{12}\right)^{\mathrm{T}} & & \\ & \ddots & \\ & & \overrightarrow{\boldsymbol{H}}_k^{n(n-1)}\, {}_{1}\hat{\boldsymbol{P}}_k^{(n-1)} \left(\overrightarrow{\boldsymbol{H}}_k^{n(n-1)}\right)^{\mathrm{T}} \end{bmatrix} \tag{41} $$

$\overrightarrow{\boldsymbol{H}}_k^{mt}$ appears in (40) and (41) if and only if $a_{\mathrm{M}}^{mt} = 1$. With these matrices determined, the upper bound and lower bound on ${}_{i}\hat{\boldsymbol{P}}_k$ can be calculated by following the approach in Section 3.2.

At the end of Section 3, it is worth mentioning that the sampling and communication intervals also impact the navigation performance. This is because they directly influence the recursive process of the EKF. In other words, they influence the availability of the measurements at each epoch.

To conclude, Section 3 establishes the theoretical relationship between the CN performance (in terms of state covariance) and the CN architecture. Using the adjacency matrices of RMG and CFG, the relationship is expressed in a complete algebraic form. This relationship will be highly conducive to analyze the performance sensitivity to the integration architecture, which will be conducted in our future work. Furthermore, this also benefits the design of the CN architectures in general multi-agent applications.

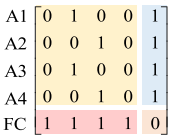
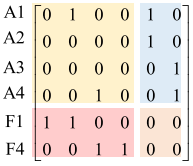
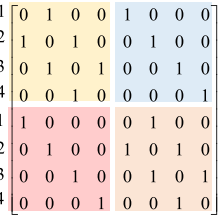
4. Simulation results

Various simulations are carried out to further illustrate and validate the proposed CN performance estimation framework. A simple multi-agent scenario is simulated for the validation, but please note, the proposed framework is generally applicable to any scenario where KF or EKF performs well. Table 1 gives the basic simulation settings, and Table 2 shows the measurements and the corresponding error covariances. In the simulations, the measurement errors are assumed to be uncorrelated with each other. It is also worth mentioning that this work focuses on establishing the theoretical framework, and thus we pay little attention to the detailed implementation of sensing and communication.

Table 3 shows 4 different CN integration architectures for navigation performance comparison. In the fully-centralized case, all measurements are used to jointly estimate the states of all the agents. There are two independent filters in the partially-centralized architecture: one is for agent 1 and agent 2, and the other is for agents 3 and 4. The difference between the partially-centralized and semi-distributed architectures is that there is filter interaction in the latter one. And filter interaction also exists in the fully-distributed architecture where each filter only estimates the states of the associated agent.

Based on the settings above, we compare the navigation performance under different architectures. First, Fig. 8 proves great significance of cooperative navigation by comparing the results under the fully-centralized architecture with those in the non-cooperative case. Taking agent 4's X-direction states as an example, this figure shows the curve of navigation performance (root of the estimated state variances)

Table 3
The communication & fusion graph (CFG) for different integration architectures.

Case No.	1	2	3	4
Description	Fully-centralized	Partially-centralized	Semi-distributed	Fully-distributed
CFG				

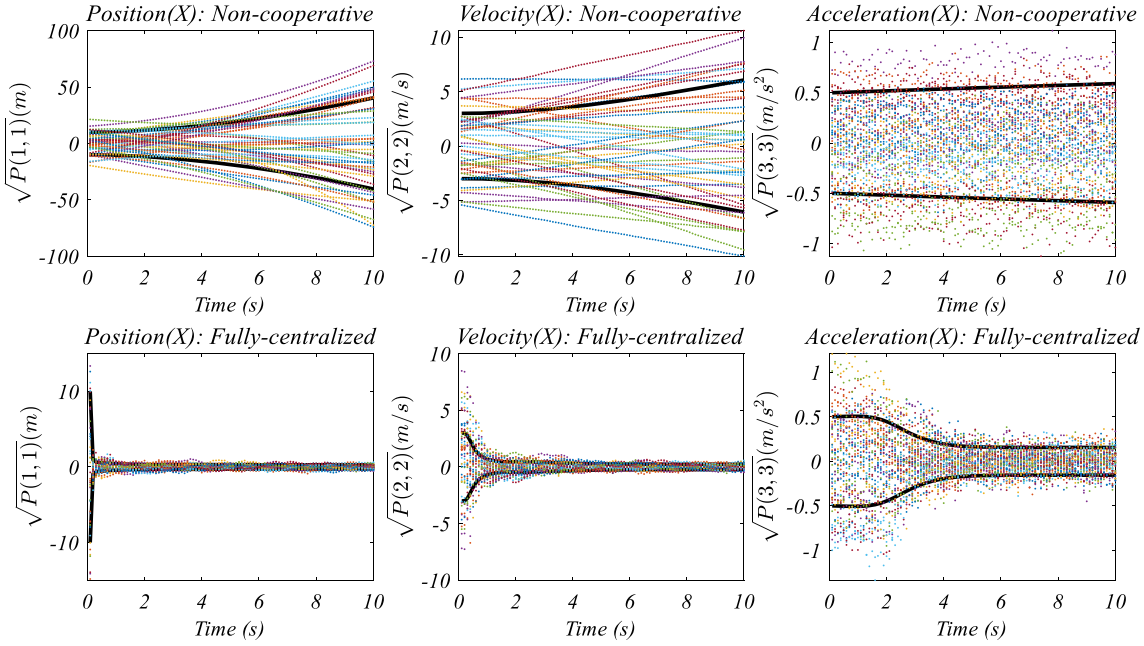


Fig. 8. Performance comparison (agent 4, X-direction) between the non-cooperative and the fully-centralized cases, where the black curves denote the estimation accuracy and the colored scatters represent the estimation errors in 50 Monte-Carlo runs. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

in the two cases. This figure also demonstrates the estimation errors in 50 Monte-Carlo runs, and we can draw the same conclusion from the Monte-Carlo result as that from the performance curve.

Then, Fig. 9 gives the comparison of navigation performance in terms of accuracy under different integration architectures without filter interaction. Accuracy is one of the most important navigation performance indicators, which is derived from the state covariance matrix in this work. Navigation accuracy is time-varying for KF-based systems, and the accuracy curves in Fig. 9 offer a direct view on the performance comparison over the simulation time span. We can draw the following conclusions from the results.

(1) For every agent, cooperative navigation can achieve improved navigation performance than the non-cooperative approach.
(2) The performance improvement is obviously different among different agents and different states of the same agent. The percentage of performance improvement mainly depends on (a) the achieved performance without cooperation and (b) the relationship between relative measurements and the states.
(3) By comparing the fully-centralized architecture with the partially-centralized one, we can find that the performance can be enhanced more by fusing the information from more agents in a centralized way. This also suggests that as the number of agents increases, cooperative navigation can bring greater performance improvement to the multi-agent system.

To demonstrate the performance under the architectures with filter interaction, Fig. 10 shows the upper bounds and lower bounds on the navigation performance under the semi-distributed and the fully-distributed architectures. Please note, the lower bounds hereafter refer to the ones with 1-step iteration. Besides, we compare the lower bounds with the performance curves derived from the fully-centralized architecture which generates the best navigation performance. As shown in this figure, the lower bounds and the upper bounds are close to each other for some states while they differ significantly for the other states. For the latter phenomenon, on the one hand, this is reasonable because there might be various schemes for handling the data incest problem induced by filter interaction, and they may result in significant different navigation performance. On the other hand, we will develop tighter lower bounds and upper bounds in the future, which will potentially reduce the difference between the two bounds. In addition, an interesting finding is that the lower bounds for some states (e.g., agent 2's X-direction acceleration) approach the best performance curve. This offers clear evidence for

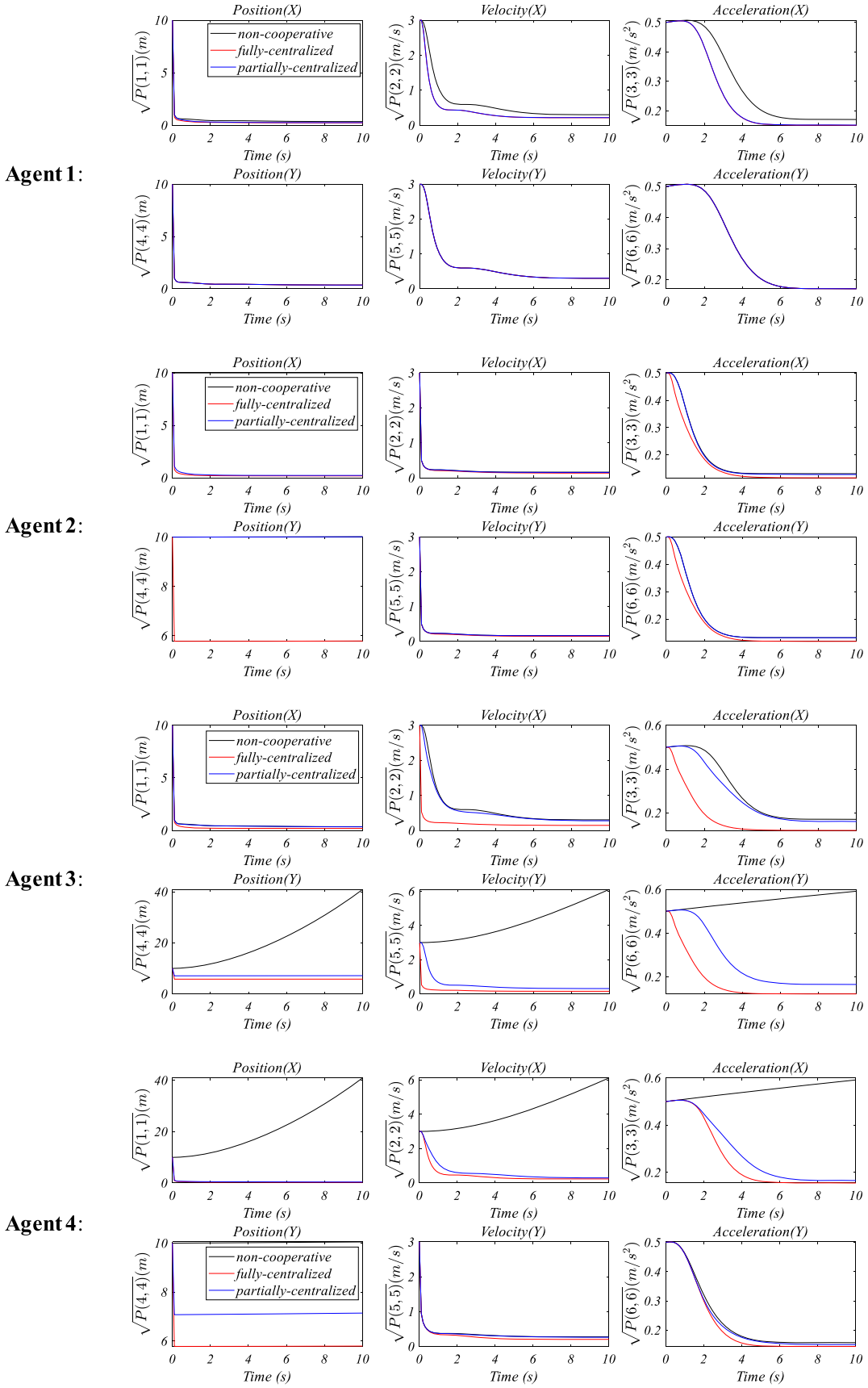


Fig. 9. Performance comparison among different integration architectures without filter interaction. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

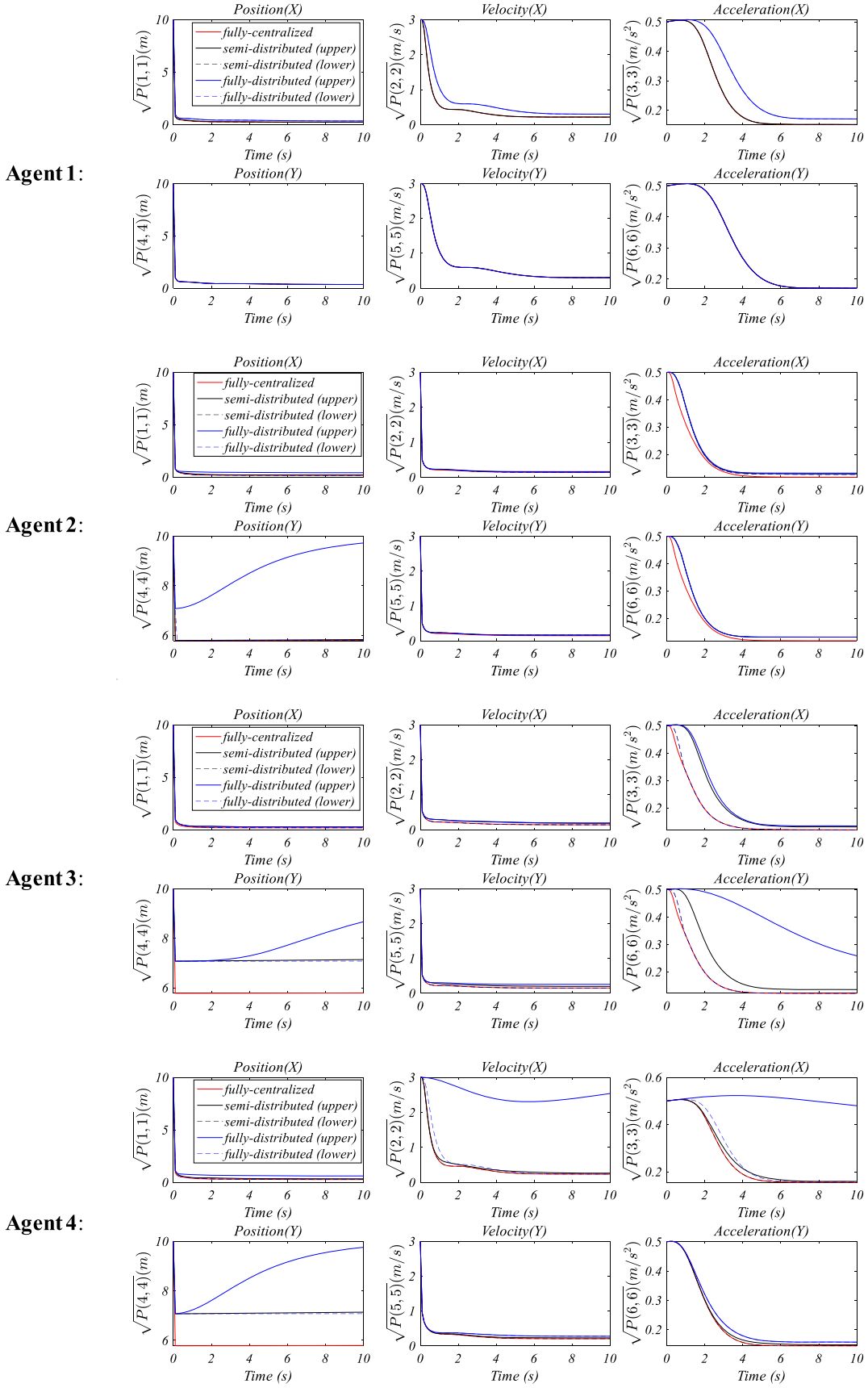


Fig. 10. Performance comparison among the fully-centralized, semi-distributed and fully-distributed architectures. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

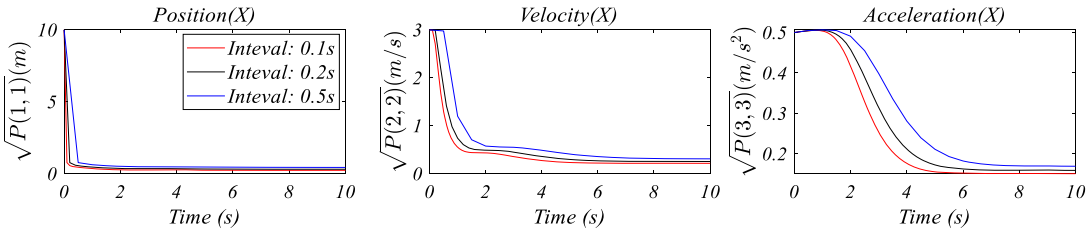


Fig. 11. The impact of sampling intervals on the navigation performance. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

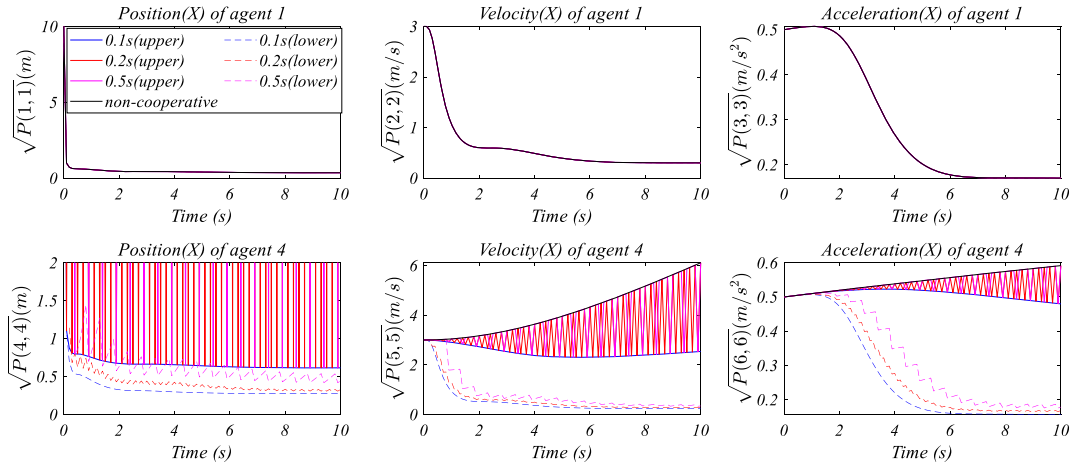


Fig. 12. The impact of communication intervals on the navigation performance. Please note, the Y-axis of the fourth subfigure is limited to [0, 2] for a clear view. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

the occurrence of data incest problem, i.e., data incest could lead to overly-optimistic estimate (the estimated covariance is smaller than the truth).

Aside from the CFG, sampling interval and communication interval also impact the navigation performance. Fig. 11 shows the performance curves under the fully-centralized architecture at different sampling rates. The results suggest that a higher sampling rate can speed up the convergence and result in better navigation performance at the expense of higher power consumption.

Besides, Fig. 12 gives the performance bounds under the fully-distributed architecture at different communication rates. This figure shows that the communication rate can significantly influence the performance for some states while it has little impact on some other states. Therefore, in practice, the communication interval can be set to different values for different links to reduce the communication load and power consumption.

Finally, it is worth noting that in this work, we do not verify the proposed upper and lower bounds with a specific CN algorithm. This is because, it is still an open research topic to design the CN algorithms for the architectures with filter interaction. Various approaches will be proposed in the future, each of which will show different performance from the others. And based on the derivations in Section 3.2, their performance will be between the lower and upper bounds. In future work, we will verify and possibly improve the proposed bounds with the newly-available CN algorithms.

5. Conclusions and future work

This work establishes a theoretical performance estimation framework for Extended Kalman Filter (EKF)-based multi-agent Cooperative Navigation (CN). Two graph variables, relative measurement graph and communication & fusion graph, are introduced to mathematically describe the CN integration architecture. Then, the relationships between the navigation performance and the CN integration architecture are derived based on EKF and graph theory. Multiple sets of simulations are carried out to validate the proposed approach, and the results suggest that the navigation performance is highly sensitive to the integration architecture as well as the sampling and communication intervals. The proposed framework lays the foundation for the offline and online design of the CN integration architectures in general multi-agent applications.

Our future work includes two folds: improving the proposed framework and applying this framework to optimize the CN architecture design. In the first fold, we aim to (a) develop tighter performance bounds for the integration architectures with filter interaction and (b) reduce the computation complexity in the performance estimation process. In the second fold, we intend to (a) propose various schemes to cope with the data incest problem, (b) develop the CN architecture design theory based on the performance estimation framework, and (c) reduce the system's computational payload and power consumption by cutting down unnecessary sampling and communication actions.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] J. Xiong, Z. Xiong, J.W. Cheong, J. Xu, Y. Yu, A.G. Dempster, Cooperative positioning for low-cost close formation flight based on relative estimation and belief propagation, Aerosp. Sci. Technol. 106 (2020) 1–14.
[2] F. Shen, J.W. Cheong, A.G. Dempster, A DSRC Doppler/IMU/GNSS tightly-coupled cooperative positioning method for relative positioning in VANETs, J. Navig. 70 (1) (2016) 120–136.
[3] Q. Zhang, H.H.T. Liu, Aerodynamic model-based robust adaptive control for close formation flight, Aerosp. Sci. Technol. 79 (2018) 5–16.
[4] S. Wang, X. Zhan, Y. Zhai, C. Chi, J. Shen, Highly reliable relative navigation for multi-UAV formation flight in urban environments, Chin. J. Aeronaut. (2020) 1–14.
[5] C. Zhang, J. Wang, D. Zhang, X. Shao, Fault-tolerant adaptive finite-time attitude synchronization and tracking control for multi-spacecraft formation, Aerosp. Sci. Technol. 73 (2018) 197–209.
[6] F. Sun, K. Turkoglu, Nonlinear consensus strategies for multi-agent networks under switching topologies: real-time receding horizon approach, Aerosp. Sci. Technol. 87 (2019) 323–330.
[7] J. Zhao, S. Yang, Integrated cooperative guidance framework and cooperative guidance law for multi-missile, Chin. J. Aeronaut. 31 (3) (2017) 546–555.
[8] T. Lyu, Y. Guo, C. Li, G. Ma, H. Zhang, Multiple missiles cooperative guidance with simultaneous attack requirement under directed topologies, Aerosp. Sci. Technol. 89 (2019) 100–110.
[9] X. Zhao, Q. Zong, B. Tian, B. Zhang, M. You, Fast task allocation for heterogeneous unmanned aerial vehicles through reinforcement learning, Aerosp. Sci. Technol. 92 (2019) 588–594.
[10] S.J. Chung, A.A. Paranjape, P. Dames, S. Shen, V. Kumar, A survey on aerial swarm robotics, IEEE Trans. Robot. 34 (4) (2018) 837–855.
[11] F. Ducatelle, G.A. Di Caro, A. Förster, M. Bonani, M. Dorigo, S. Magnenat, F. Mondada, R. O'Grady, C. Pinciroli, P. Rétornaz, V. Trianni, L.M. Gambardella, Cooperative navigation in robotic swarms, Swarm Intell. 8 (1) (2014) 1–33.
[12] X. Liu, X. Zhan, S. Wang, Y. Zhai, Measurement-domain cooperative navigation for multi-UAV systems augmented by relative positions, J. Aeronaut. Astronaut. Aviat. 52 (2020) 403–416.
[13] A.R. Vetrella, G. Fasano, D. Accardo, Satellite and vision-aided sensor fusion for cooperative navigation of unmanned aircraft swarms, J. Aerosp. Inform. Syst. 14 (6) (2017) 1–18.
[14] V.O. Sivaneri, J.N. Gross, UGV-to-UAV cooperative ranging for robust navigation in GNSS-challenged environments, Aerosp. Sci. Technol. 71 (2017) 245–255.
[15] V.O. Sivaneri, J.N. Gross, Flight-testing of a cooperative UGV-to-UAV strategy for improved positioning in challenging GNSS environments, Aerosp. Sci. Technol. 82–83 (2018) 575–582.
[16] W. Lee, H. Bang, H. Leeghim, Cooperative localization between small UAVs using a combination of heterogeneous sensors, Aerosp. Sci. Technol. 27 (1) (2013) 105–111.
[17] J. Nicosia, Decentralized cooperative navigation for spacecraft, in: IEEE Aerospace Conference, Big Sky, MT, March 2007.
[18] X. Bo, A.A. Razzaqi, G. Farid, A review on optimal placement of sensors for cooperative localization of AUVs, J. Sens. 2019 (2019) 1–13.
[19] G. Xiao, B. Wang, Z. Deng, M. Fu, Y. Ling, An acoustic communication time delays compensation approach for master-slave AUV cooperative navigation, IEEE Sens. J. 17 (2017) 504–513.
[20] G. Soatti, M. Nicoli, N. Garcia, B. Denis, R. Raulefs, H. Wymeersch, Implicit cooperative positioning in vehicular networks, IEEE Trans. Intell. Transp. Syst. 19 (2018) 3964–3980.
[21] H. Ko, B. Kim, S. Kong, GNSS multipath-resistant cooperative navigation in urban vehicular networks, IEEE Trans. Veh. Technol. 64 (12) (2015) 5450–5463.
[22] A. Minetto, GNSS-only collaborative positioning methods for networked receivers, PhD Thesis, Politecnico di Torino, 2020.
[23] S. Zhang, H. Zhang, A review of wireless sensor networks and its applications, in: IEEE International Conference on Automation and Logistics, Zhengzhou, China, August 2012.
[24] L. Heng, G.X. Gao, Accuracy of range-based cooperative positioning: a lower bound analysis, IEEE Trans. Aerosp. Electron. Syst. 53 (5) (2017) 2304–2316.
[25] A.M.C. So, Y. Ye, Theory of semidefinite programming for sensor network localization, Math. Program. 109 (2007) 367–384.
[26] J.A. Costa, N. Patwari, A.O. Hero, Distributed weighted-multidimensional scaling for node localization in sensor networks, ACM Trans. Sens. Netw. 2 (1) (2006) 39–64.
[27] J. Shen, S. Wang, Y. Zhai, X. Zhan, Cooperative relative navigation for multi-UAV systems by exploiting GNSS and peer-to-peer ranging measurements, IET Radar Sonar Navig. 15 (12) (2020) 1–16.
[28] M. Chen, Z. Xiong, J. Liu, R. Wang, J. Xiong, Cooperative navigation of unmanned aerial vehicle swarm based on cooperative dilution of precision, Int. J. Adv. Robot. Syst. 17 (3) (2020) 1–10.
[29] L. Luft, T. Schubert, S.I. Roumeliotis, W. Burgard, Recursive decentralized localization for multi-robot systems with asynchronous pairwise communication, Int. J. Robot. Res. 37 (10) (2018) 1152–1167.
[30] X. Liu, S. Xu, Multi-UAV cooperative navigation algorithm based on federated filtering structure, in: 2018 IEEE CSAA Guidance, Navigation and Control Conference, GNCC, Xiamen, China, August 2018, pp. 1–5.
[31] G.M. Hoang, Cooperative multisensor localization for connected vehicles, PhD Thesis, Paris Institute of Technology, 2018.
[32] D. Fox, W. Burgard, H. Kruppa, S. Thrun, A probabilistic approach to collaborative multi-robot localization, Auton. Robots 8 (2000) 325–344.
[33] C. Tang, L. Zhang, Y. Zhang, H. Song, Factor graph-assisted distributed cooperative positioning algorithm in the GNSS system, Sensors 18 (2018) 1–12.
[34] A.K. Gostar, T. Rathnayake, R. Tennakoon, A. Bab-Hadiashar, G. Battistelli, L. Chisci, R. Hoseinnezhad, Cooperative sensor fusion in centralized sensor networks using Cauchy–Schwarz divergence, Signal Process. 167 (2020) 1–12.
[35] X. Wang, A.K. Gostar, T. Rathnayake, B. Xu, A. Bab-Hadiashar, R. Hoseinnezhad, Centralized multiple-view sensor fusion using labeled multi-Bernoulli filters, Signal Process. 150 (2018) 75–84.
[36] J. Zhou, G. Gu, X. Chen, Distributed Kalman filtering over wireless sensor networks in the presence of data packet drops, IEEE Trans. Autom. Control 64 (2018) 1603–1610.
[37] S. Tomic, M. Beko, R. Dinis, Distributed RSS-based localization in wireless sensor networks based on second-order cone programming, Sensors 14 (2014) 18410–18432.
[38] J.L. Gross, J. Yellen, Fundamentals of graph theory, in: J.L. Gross, J. Yellen, P. Zhang (Eds.), Handbook of Graph Theory, New York, 2nd ed., 2013, pp. 2–20.
[39] A.V. Aho, Computer representations of graphs, in: J.L. Gross, J. Yellen, P. Zhang (Eds.), Handbook of Graph Theory, New York, 2nd ed., 2013, pp. 56–66.
[40] A. Jazwinski, Stochastic Processes and Filtering Theory, vol. 64, Academic Press, 1970.
[41] J. Curn, D. Marinescu, N. O'Hara, V. Cahill, Data incest in cooperative localisation with the Common Past-Invariant Ensemble Kalman filter, in: Proceedings of the 16th International Conference on Information Fusion, 2013, pp. 68–76.
[42] Y. Bar-shalom, X. Li, T. Kirubarajan, Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software, John Wiley & Sons, 2001.