Aerospace Science and Technology 123 (2022) 107491

Contents lists available at ScienceDirect

Aerospace Science and Technology

www.elsevier.com/locate/aescte

Offline optimization of sensor configuration and integration architecture for efficient cooperative navigation

Hanyu Wang [a,b], Shizhuang Wang [a], Xingqun Zhan [a,*], Jiawen Shen [a]

[a] School of Aeronautics and Astronautics, Shanghai Jiao Tong University, Shanghai, 200240, China
[b] Laboratory of Science and Technology on Marine Navigation and Control, China State Shipbuilding Corporation, China

ARTICLE INFO

Article history:
Received 1 December 2021
Received in revised form 27 February 2022
Accepted 6 March 2022
Available online 14 March 2022
Communicated by Chaoyong Li

Keywords:
Cooperative navigation
Sensor configuration
Integration architecture
Combinatorial optimization

ABSTRACT

Multi-agent cooperative navigation can improve navigation performance by taking advantage of inter-agent communication and relative-sensing capabilities. For Multi-Agent Systems (MAS), the achievable navigation performance, communication cost, computation complexity, and sensor cost are directly influenced by the sensor configuration and the integration architecture. To reduce hardware cost and computation/communication load, this paper proposes a feasible method to realize the offline design of sensor configuration and integration architecture for cooperative navigation. Specifically, this goal is achieved by solving a multi-objective combinatorial optimization problem, where the sensor configuration and integration architecture are considered as the optimization variables; the navigation performance requirement is the constraint; and the communication cost, computation complexity, and sensor cost are considered as the optimization objectives. This optimization problem is solved by a Multi-Objective Simulated Annealing (MOSA) algorithm. Using a MAS with three unmanned aerial vehicles as an example, simulations are carried out to validate the proposed method, and the results show its feasibility and effectiveness. The Pareto solutions reveal the characteristics and applicability of different integration architectures in different scenarios, which is meaningful for practical applications.

1. Introduction

Recently, Multi-Agent Systems (MAS) have attracted increasing research interest especially on their high efficiency and high robustness [1]. A variety of MAS have been developed and put into the civilian and military fields, including multi-robot systems, multi-missile systems, multi-Unmanned Aerial Vehicle (UAV) systems, etc [2–5]. High-precision navigation is precondition for MAS to achieve specific goals such as cooperative search, cooperative combat, and cooperative package delivery. To reduce the cost of the navigation system and improve its performance, various multi-agent cooperative navigation technologies have been developed. The basic idea of cooperative navigation is employing the communication and relative-sensing capabilities in the MAS to improve the navigation performance of each agent [6–8].

Over the past decade, there have been many researches on cooperative navigation algorithms for different platforms, including autonomous underwater vehicles [9–11], UAVs [12–14], robots [15–18], vehicular networks [19–22] and so on. In these studies, the following three types of state estimation methods are commonly used: least-squares estimation, filter, and optimization. The least-squares estimation is a snapshot method [23–25], which only uses the measurements at the current time to estimate the states. Filter algorithms mainly include Kalman Filter (KF), Extended KF (EKF) [9,14], Unscented KF (UKF) [26], Particle Filter (PF) [27], etc. Among them, KF and EKF are the most widely used algorithms because of their simplicity. As compared to least-squares estimation, filter-based estimation methods can generally achieve higher accuracy because they further utilize the dynamic models of vehicles. Optimization methods have also attracted extensive attention in recent years [28–30], which show the superior performance to filters in some complex situations. However, the computation complexity of optimization methods (e.g., factor graph optimization) is much higher than that of the filters.

It is worth mentioning that the prior researches mainly focus on designing the cooperative navigation algorithms for small-scale MAS with specific sensor configurations in static operation scenarios. These researches paid little attention to large-scale swarms, e.g., a multi-UAV system with hundreds of or even thousands of UAVs. For these large-scale MAS, the traditional centralized architecture is no longer practically feasible due to the huge computation and communication load. On the contrary, in decentralized architectures, each vehicle just communicates with its neighbors

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

https://doi.org/10.1016/j.ast.2022.107491
1270-9638/

and updates its own state locally. It dramatically reduces the requirements for communication and computation abilities. Besides, the loss of a subset of nodes and links does not necessarily prevent the rest of the system from functioning, improving the reliability and flexibility of MAS. Therefore, from the perspective of cooperative navigation, decentralized architectures will be the mainstream solutions for large-scale swarms.

For a MAS, there will be various decentralized architectures, each of which has a specific fusion strategy. The navigation performance, computation complexity, and communication cost are directly influenced by the fusion strategy. Prior researches have proposed a series of decentralized fusion strategies to reduce communication and computation load. In [17], the authors propose a cooperative navigation fusion strategy that each vehicle just sends its own estimate to the neighbors, so that the communication cost can be greatly reduces. This fusion strategy has become a mainstream way for decentralized cooperative navigation. Nevertheless, few research considered how to select a decentralized fusion strategy by a global optimization method. In other words, the existing fusion strategies are designed under some potential constraints, e.g., communication links are completely connected. Therefore, they can only be applied to some specific scenarios. In order to solve this problem, in [31], the authors proposed the concept of integration architecture to generally describe the relative measurement topology and the fusion strategy, and then the relationship between integration architecture and navigation performance is revealed. On the basis of [31], this paper further focuses on how to design the integration architecture in an offline manner by optimization method. The offline design of integration architecture before the operational flight is essential for the practical applications of MAS, because a proper integration architecture can achieve the balance between navigation performance and communication/computation load. Besides, this work also takes the sensor configuration into account, which determines the maximum achievable navigation performance and also the hardware cost. In a word, we propose a method to jointly design the sensor configuration and integration architecture for a multi-agent cooperative navigation system, with the objectives of (a) satisfying navigation performance requirement and (b) reducing software complexity and hardware cost. As the proposed method is based on optimization, it can be applied to the general case.

The offline design of sensor configuration and integration architecture could be formulated as a multi-objective combinatorial optimization problem after a detailed analysis. The sensor configuration and integration architecture are considered to be the optimization variables. The optimization objectives include sensor cost, computational complexity, and communication cost. The required navigation performance is considered as a constraint. In this work, the navigation performance is described by a covariance matrix. And it is noteworthy that EKF is selected as the estimator in this work, and the theoretical navigation performance is derived from the covariance matrix output from the filters. Finally, the optimization problem is solved using Multi-Objective Simulated Annealing (MOSA). A simulation was carried out to find the corresponding Pareto solution set, which proves the validity of this algorithm.

The rest of this paper is organized as follows: Section 2 gives the mathematical model of cooperative navigation and presents the problem statement for the offline design of sensor configuration and integration architecture. Then, Section 3 formulates the optimization problem as a combinatorial optimization mathematical model. And in Section 4, the MOSA algorithm is adopted to solve the optimization problem. Simulations are carried out in Section 5 to validate the proposed method. And finally, Section 6 draws the conclusions.

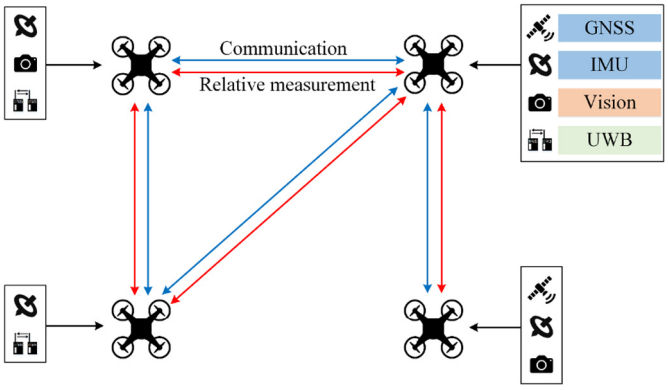

Fig. 1. An example of multi-UAV cooperative navigation.

2. Problem statement

Consider a cluster with N collaborating UAVs, as shown in Fig. 1. Each UAV is equipped with one or more navigation sensors, including Global Navigation Satellite System (GNSS) receiver, Inertial Measurement Unit (IMU), vision sensor, Ultra-Wideband sensor (UWB) and so on. Let $\boldsymbol{x}_k^i$ be the navigation state vector of UAV i at time epoch k, generally including position, velocity, etc. The state transition model of UAVs is assumed to be known, which is shown as follows:

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

where $\boldsymbol{\omega}_k^i$ is the process noise.

Sensor configuration describes the sensors that are installed on the UAVs. These sensors can provide absolute and relative measurements. Combining all the measurements into a vector $\boldsymbol{z}_k$, the measurement model can be formulated as:

$$\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{v}_k, \tag{2}$$

where $\boldsymbol{v}_k$ is the measurement noise.

In practical applications, the UAVs can either power on all the sensors or only use some of the sensors depending on the surrounding conditions and required navigation performance. For example, more sensors should be powered on in a scenario with high navigation performance requirement and more sensors should be powered off in the opposite case. This is because using more available sensors can provide higher navigation performance while using less can save more energy. To describe the sensor on-off situations, a directed graph variable $G_M = \langle V_M, E_M \rangle$ (V_M is the set of vertexes, and E_M is the set of edges between vertexes), named as measurement topology, is introduced. In measurement topology, V_M represents the UAVs, and E_M represents the absolute and relative measurements of UAVs.

In the inter-agent communication phase, the UAVs share the measurements and navigation results with their neighbors, which relies on the communication capability. And in the state estimation phase, the navigation solutions are obtained by the estimators on the UAVs, which is at the expense of computation load. Different fusion strategies will result in different communication cost and computation complexity. Another graph variable $G_F = \langle V_F, E_F \rangle$, called fusion topology, is defined to describe the fusion strategy. In this variable, V_F represents the UAVs, and the edge in E_F from vertex i to vertex j indicates that the states of UAV i are estimated by the filter on UAV j. The two graph variables, measurement topology and fusion topology defined above, combine into the integration architecture.

A comparison among a fully-centralized architecture, a locally-centralized architecture, a locally-decentralized architecture, and a

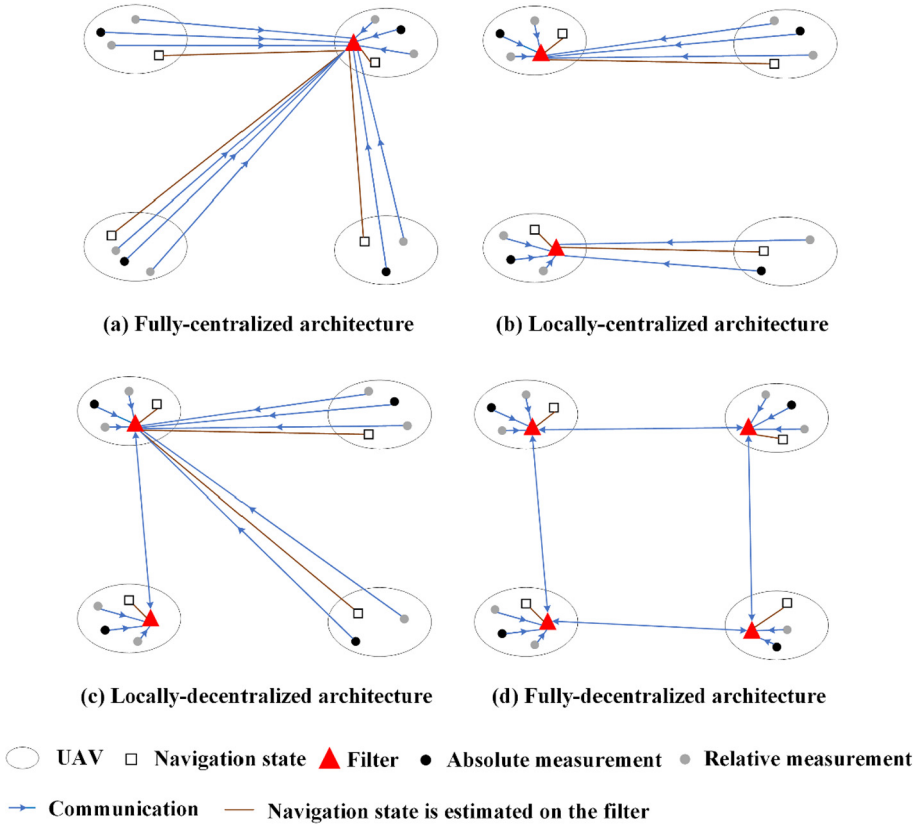

Fig. 2. A comparison among different fusion topologies.

fully-decentralized architecture is shown in Fig. 2. It can be seen that in the fully-centralized architecture, all the measurements are transmitted to one filter, in which all the navigation states are estimated together. The fully-centralized architecture can offer the highest navigation performance while its communication and computation load are heavy. In the locally-centralized architecture, there are multiple filters, each of which estimates the states of a subset of agents using only the information inside this subset. Similarly, in the locally-decentralized, there are also multiple filters, each of which estimate the states of multiple UAVs. The difference is that these filters will share information with each other. In contrast, in the fully-decentralized architecture, each UAV just communicates with its neighbors and estimates its own state locally. Because of the different information fusion strategies among these four integration architectures, the corresponding navigation performance can be very different. Generally, the order of overall navigation performance from highest to lowest is fully-centralized, locally-decentralized, locally-centralized, fully-decentralized. In return, the communication/computation load varies greatly among these four integration architectures. Besides, an obvious difference between the fully-centralized/locally-centralized architectures and the locally-decentralized/fully-decentralized ones is that there is filter interaction (defined in [31]) in the latter. Filter interaction indicates that two filters share information with each other. It will lead to data incest problem, which is analyzed in Section 3.3.

In order to understand sensor configuration and integration architecture (measurement topology and fusion topology) better, Fig. 3 further illustrates the concept of sensor configuration, measurement topology, and fusion topology. This figure also shows the relationship between the navigation performance and these three variables.

During the entire flight profile, the operation scenario of the UAV cluster is time-variant, which leads to the change in the required navigation performance. Note that the navigation system should not be targeted at achieving the highest navigation performance but aim at satisfying the navigation performance requirement. This is because achieving unnecessarily-high navigation performance will lead to a waste of communication and computation resources. This motivates us to properly design the sensor configuration and integration architecture before the flight.

In this work, the offline design of sensor configuration and integration architecture is realized through solving a multi-objective optimization problem. As mentioned above, the navigation performance requirement is variant in different phases of the mission profile. Approximately, the whole flight profile can be divided into several segments, and in each segment the navigation performance requirement is invariant. Each segment is named as a baseline scenario in this paper. The designer needs to set the corresponding weights for each baseline scenario according to the flight mileage or time. Due to the different navigation performance requirements in each baseline scenario, the corresponding integration architecture is also different. The final optimization result is the Pareto solution set of sensor configuration and integration architecture sequence for different baseline scenarios. The objective functions are sensor cost and the weighted average of communication cost and computation complexity in different baseline scenarios, and the constraint is navigation performance requirements in different baseline scenarios. In this paper, state covariance is taken as the navigation performance index.

Finally, the offline design of Sensor Configuration (SC) and Integration Architecture (IA) is formulated as:

$$\begin{aligned} &\text{minimize}\left\{ f_{sensor}(\text{SC}),\, f_{communication}(\text{SC, IA}),\, f_{computation}(\text{SC, IA}) \right\}, \\ &s.t.\ C_p \leq 0 \end{aligned} \tag{3}$$

where f_{sensor}, $f_{communication}$ and $f_{computation}$ are sensor cost, communication cost and computation complexity, respectively, and C_p

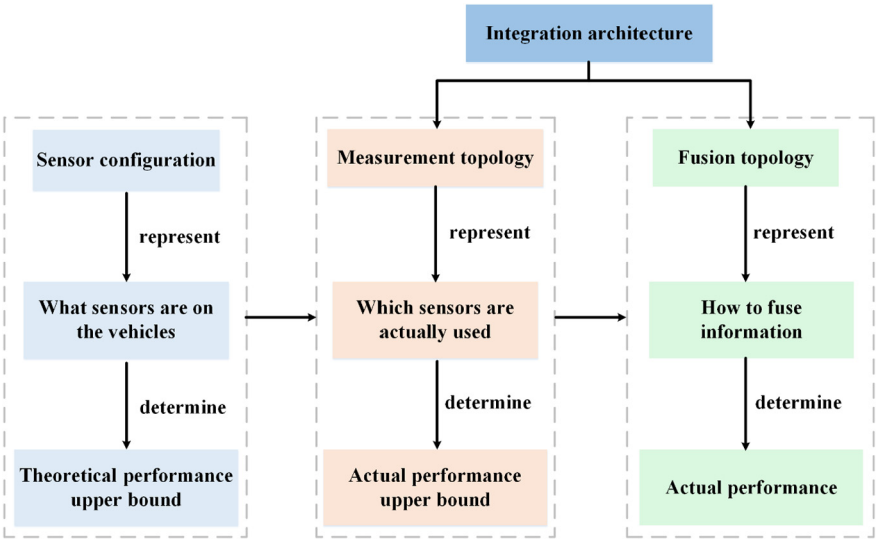

Fig. 3. The concept of sensor configuration, measurement topology and fusion topology.

is the required navigation performance minus the actual navigation performance.

3. Optimization model

The algebraic expressions of optimization variables, objective functions and constraint in the optimization problem above are defined in detail in this section. Without loss of generality, we select UAVs as the research objective so that the types of sensors can be determined.

3.1. Optimization variables

3.1.1. Sensor configuration (SC)

SC represents the sensors installed on each UAV in a cluster. This work uses GNSS receiver, UWB, and vision sensor as three typical sensors. GNSS receiver can provide absolute position information, UWB can measure the relative distance between UAVs, and vision sensor can give the inter-UAV relative orientation information. The UAVs are assumed to flight in the same 2-D plane. The measurement model of a GNSS receiver is shown as:

$$\begin{bmatrix} g_x(k) \\ g_y(k) \end{bmatrix} = \begin{bmatrix} p_x(k) \\ p_y(k) \end{bmatrix} + \begin{bmatrix} v_x(k) \\ v_y(k) \end{bmatrix}, \tag{4}$$

where p_x and p_y forms the 2-D absolute position of the UAV, v_x and v_y are the measurement errors. The measurement model of UWB between UAV i and UAV j is as follows:

$$r_{ij}(k) = r_{ji}(k) = \sqrt{\left(p_x^i(k) - p_x^j(k)\right)^2 + \left(p_y^i(k) - p_y^j(k)\right)^2} + v_r(k) \tag{5}$$

Note that r_{ij} means the measurement is available for UAV i. The measurement model of vision sensor is given by:

$$\theta_{ij}(k) = \arctan \frac{p_y^j(k) - p_y^i(k)}{p_x^j(k) - p_x^i(k)} - \theta_i(k) + v_\theta(k) \tag{6}$$

where $\theta_i(k)$ is the heading angle of UAV i and we assume its value is exactly known (this angle can be obtained by fusing an IMU and a magnetometer). Note that θ_{ij} indicates that this information is only available for UAV i.

Here, two assumptions are made according to the characteristics of the navigation sensors:

Assumption 1. To get the UWB relative range measurement, the both UAVs need equipping with the UWB sensors of the same level.

Assumption 2. To get the vision relative measurement, only one UAV needs to be equipped with the vision sensor.

For each navigation sensor, we set four different levels (0, 1, 2, 3) to represent different qualities of this sensor. Specifically, the higher level represents the higher cost and the lower measurement error. And level-0 indicates that the sensor is not equipped.

Then, the SC of a UAV cluster can be expressed as a $3 \times N$ algebraic matrix (N is the number of UAVs), where the three elements in each column from top to bottom are the levels of GNSS receiver, UWB and vision sensor, respectively. An example is given as follows:

$$\mathrm{SC} = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 2 & 2 \\ 1 & 2 & 0 \end{pmatrix}. \tag{7}$$

The SC matrix in Eq. (7) represents that in a cluster with three UAVs, UAV 1 is equipped with a level-1 GNSS receiver and a level-1 vision sensor, UAV 2 is equipped with a level-2 GNSS receiver, a level-2 UWB and a level-2 vision sensor, and UAV 3 installs a level-3 GNSS receiver and a level-2 UWB.

3.1.2. Measurement topology (MT)

MT represents which sensors are powered on and which UAV the measurements are used to estimate. MT is described with a multi-dimensional matrix by applying the algebraic graph theory. This matrix not only describes the availability of each measurement but also indicates the usage of every measurement.

In this paper, three types of sensors are considered, including GNSS receiver, UWB, and vision sensor. In this case, a 3-dimensional 0-1 matrix, MT, with the size $N \times N \times 3$, is used to mathematically describe the measurement topology. Note that the dimension of MT depends on the specific characteristics of relative sensors rather than the number of sensor types. Specifically, MT$(:, :, 1)$ is used to describe the measurement topology of GNSS and UWB, and MT$(:, :, 2)$ and MT$(:, :, 3)$ are for the vision sensor. Table 1 shows the detailed descriptions of each element in MT, and Fig. 4 presents an illustrative example.

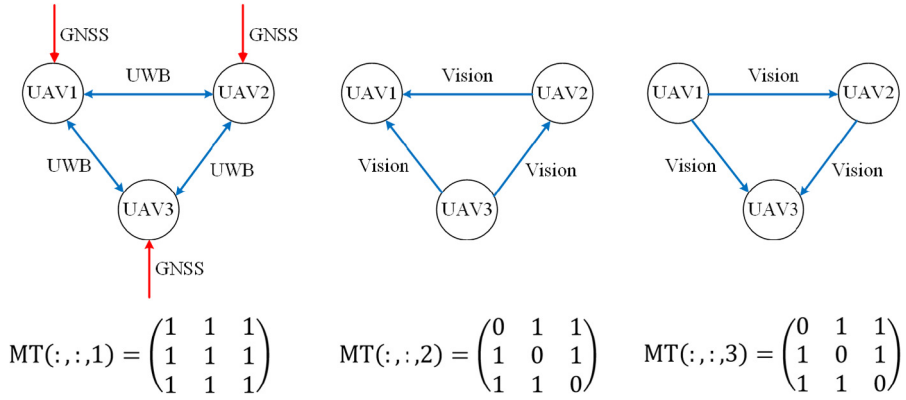

Fig. 4. An example of the whole measurement topology.

Table 1
Definitions of the measurement topology matrix.

Matrix element	Descriptions
MT(i, i, 1)=1	UAV i powers on the GNSS receiver to observe itself.
	The obtained measurements are used to estimate its own state.
MT(i, j, 1)=1 ($i \neq j$)	Both UAVs i and j power on the UWB sensors to observe each other.
	The obtained measurements are used to estimate the state of UAV j.
MT(i, j, 2)=1 ($i < j$)	UAV i powers on the vision sensor to observe UAV j.
	The obtained measurements are used to estimate the state of UAV j.
MT(j, i, 2)=1 ($i < j$)	UAV i powers on the vision sensor to observe UAV j.
	The obtained measurements are used to estimate the state of UAV i.
MT(i, j, 3)=1 ($i < j$)	UAV j powers on the vision sensor to observe UAV i.
	The obtained measurements are used to estimate the state of UAV j.
MT(j, i, 3)=1 ($i < j$)	UAV j powers on the vision sensor to observe UAV i.
	The obtained measurements are used to estimate the state of UAV i.
MT(i, i, k) ≡0 ($k = 2, 3$)	The diagonal elements in MT(:, :, 2) and MT(:, :, 3) are all 0.

3.1.3. Fusion topology (FT)

FT represents the fusion strategy of cooperative navigation. In this paper, the FT is represented by a $N \times N$ 0-1 algebraic matrix, FT. In this matrix, FT(i, j)=1 means that the state of UAV i is estimated by the filter on UAV j.

Here, an assumption is made to simplify the model [31].

Assumption 3. The state of each UAV is estimated on and only on one filter.

Under this assumption, there is no need to consider how to achieve the consensus among different estimates on the same state. And in this case, for each row of FT, only one element is 1 and the others are all 0.

3.2. Objective functions

3.2.1. Sensor cost

Sensor cost represents the overall hardware cost of the navigation system. In this paper, the quality of each sensor is discretized into four levels, and the higher level represents higher cost and higher measurement accuracy. Table 2 gives the sensor cost for different levels of each sensor. Note that the values here are only for preliminary analysis, which may not be very accurate. Then, the sensor cost of the navigation system can be easily computed by combining the sensor configuration matrix (i.e., SC) and the sensor costs in Table 2 [32].

Table 2
The sensor cost for different levels (Unit: CNY).

Sensor type	Level 0	Level 1	Level 2	Level 3
GNSS receiver	0	250	350	450
UWB	0	100	150	200
Vision sensor	0	150	200	250

3.2.2. Communication cost

Communication cost is measured by the communication bits, which indicates the energy consumption of communication. There are two modes of data transmission in cooperative navigation: transmitting a fixed-size quantized measurement [18,33], or sending the full real-valued data [16]. In this paper, we consider the latter, because the former will deteriorate the state estimation results in the case of few quantized bits [18]. For simplifying the model, we make two assumptions as follows.

Assumption 4. Communication links are completely connected between any couple of UAVs.

Assumption 5. The communication cost for transmitting each real-valued data is the same, i.e., $O(1)$.

Based on these assumptions, we can compute the overall communication cost as:

$$\text{Communication Cost} = O(1) \times (n_{measurement} + n_{state}), \tag{8}$$

where $n_{measurement}$ is the number of transmitted measurements and n_{state} is the number of transmitted states.

3.2.3. Computation complexity

Computation complexity refers to the amount of resource required to estimate the navigation states. In this paper, the EKF is used to estimate the state of each UAV. For an EKF, the computational complexity is mainly affected by the dimension of the state covariance matrix $\boldsymbol{P}$ and the number of fused measurements. And for the filter on a UAV, the dimension of $\boldsymbol{P}$ is equal to the dimension of the navigation state of a single UAV multiplied by the number of UAVs whose states are estimated in this filter. Then, the

computation complexity of an arbitrary filter i can be calculated as [17]:

$$\text{Computation Complexity of filter } i = O\left[(N_{state} \times N_i)^2 \times n^i_{measurement}\right], \tag{9}$$

where N_{state} is the dimension of the navigation state vector of a single UAV, N_i is the number of UAVs whose states are estimated on filter i, and $n^i_{measurement}$ is the number of measurements fused on filter i.

Then, the total complexity of the cooperative navigation system is computed by adding up the complexity of each filter:

$$\text{Computation Complexity} = \sum_i O\left[(N_{state} \times N_i)^2 \times n^i_{measurement}\right]. \tag{10}$$

3.3. Constraint

For the design of sensor configuration and integration architecture, the optimization constraint is the required navigation performance. The navigation performance of a MAS can be divided into the absolute navigation performance and the relative navigation performance. It can be expressed by a graph algebraic matrix:

$$\text{Navigation performance} = \begin{pmatrix} p_1 & p_{12} & \cdots & p_{1i} & \cdots & p_{1N} \\ & p_2 & \cdots & p_{2i} & \cdots & p_{2N} \\ & & \ddots & \vdots & \ddots & \vdots \\ & & & p_i & \cdots & p_{iN} \\ & & & & \ddots & \vdots \\ & & & & & p_N \end{pmatrix}, \tag{11}$$

where p_i is the absolute navigation performance of UAV i, and p_{ij} is the relative navigation performance between UAV i and UAV j.

In this paper, the navigation performance is measured by the theoretical positioning accuracy, which is derived from the covariance matrices output from the filters. In the following part, we will discuss how to quantify the navigation performance in different integration architectures.

As previously depicted in Fig. 2, the integration architectures can be divided into two categories [31]: (a) the architectures without filter interaction: fully-centralized and locally-centralized ones, and (b) the architectures with filter interaction: locally-decentralized and fully-decentralized architectures.

For category (a), the state estimation can be implemented with a standard EKF procedure. And thus, the theoretical navigation accuracy can be directly derived from the posterior state error covariance matrices. But this is not the case for category (b) due to the existence of filter interaction. Filter interaction can lead to the data incest problem, which means one measurement is used explicitly or implicitly for several times. It will introduce correlations among information, which is difficult to track in practical applications. In this case, if the state estimation is directly implemented using standard EKF while ignoring the correlations, the covariance matrices output from the filters will be overly optimistic (i.e., smaller than the true error covariance) and the filters may be divergent [34]. To estimate the achievable accuracy in this case, [31] provided a method to compute the upper bound on the theoretical accuracy for the integration architectures belonging to category (b). This method is adopted in this work, and its basic principle is briefly explained as follows.

Consider two filters with interaction as an example. The information used to estimate navigation states can be divided into two categories: (a) local information with no correlation, which will not lead to filter interaction, and (b) interaction information between filters that will introduce correlations. This method can be divided into three steps. First, as shown in Eq. (12), the two filters estimate navigation states with local information $\boldsymbol{z}^{local}_k$ to obtain $\hat{\boldsymbol{x}}^{local+}_k$ and $\boldsymbol{P}^{local+}_k$. As only the local information is fused, the estimation results have no correlation with states estimated by the other filter. Then, the filters fuse $\hat{\boldsymbol{x}}^{local+}_k$ and interaction information $\boldsymbol{z}^{inter}_k$ using EKF update equations to improve the estimation accuracy, which is shown in Eq. (13). Finally, as shown in Eq. (14), $\hat{\boldsymbol{x}}^{local+}_k$ and $\boldsymbol{P}^{local+}_k$ are used to perform the EKF recursive process for next epoch. That is to say, at each epoch, only the local information is propagated to next epoch, so that the priori estimate at next epoch has no correlation with interaction measurement information and a standard EKF can be applied.

Table 3
The reference variance of each sensor.

Sensor type	GNSS receiver	UWB	Vision sensor
Reference variance	1 m^2	0.05 m^2	0.001 rad^2

1) Fusion of local information:

$$\begin{aligned} \hat{\boldsymbol{x}}^{local+}_k &= \hat{\boldsymbol{x}}^{local-}_k + \boldsymbol{K}^{local}_k\left[\boldsymbol{z}^{local}_k - \boldsymbol{h}^{local}_k\left(\hat{\boldsymbol{x}}^{local-}_k\right)\right] \\ \boldsymbol{P}^{local+}_k &= \left(\boldsymbol{I} - \boldsymbol{K}^{local}_k \boldsymbol{H}^{local}_k\right)\boldsymbol{P}^{local-}_k. \end{aligned} \tag{12}$$

2) Fusion of interaction information:

$$\begin{aligned} \hat{\boldsymbol{x}}^{+}_k &= \hat{\boldsymbol{x}}^{local+}_k + \boldsymbol{K}^{inter}_k\left[\boldsymbol{z}^{inter}_k - \boldsymbol{h}^{inter}_k\left(\hat{\boldsymbol{x}}^{local+}_k\right)\right] \\ \boldsymbol{P}^{+}_k &= \left(\boldsymbol{I} - \boldsymbol{K}^{inter}_k \boldsymbol{H}^{inter}_k\right)\boldsymbol{P}^{local+}_k. \end{aligned} \tag{13}$$

3) Propagation of local information:

$$\hat{\boldsymbol{x}}^{local-}_{k+1} = \boldsymbol{F}_k \hat{\boldsymbol{x}}^{local+}_k \; \boldsymbol{P}^{local-}_{k+1} = \boldsymbol{F}_k \boldsymbol{P}^{local+}_k \boldsymbol{F}^T_k + \boldsymbol{Q}_k. \tag{14}$$

At the end of this subsection, the relationship between the measurement accuracy and the sensor level is established, so that the measurement covariance matrix R can be determined. In this work, it is assumed that the measurement error variance of a sensor is inversely proportional to its level, as given below:

$$C = \frac{C_{ref}}{L} \tag{15}$$

in which C is the error variance of a sensor, C_{ref} is the reference variance of the level-1 sensor, and L is the level of sensor. The reference variances of each sensor are shown in Table 3 [32].

4. Solving the optimization problem by MOSA

In this section, the MOSA algorithm is tailored to solve the combinatorial optimization model above. First, the complexity of the combinatorial optimization problem is analyzed, and then the procedures of MOSA are described in detail.

4.1. Complexity analysis of the optimization problem

In the optimization problem given in Section 3, all the optimization variables are discrete. This means the number of feasible solutions for this problem is finite. A rough estimate of the number of feasible solutions is presented as follows.

In a MAS with N vehicles, the vehicles can obtain a maximum of $O(N^2)$ relative measurements at each epoch. Each relative measurement has two statuses: “on” and “off”. Therefore, the number

of integration architectures derived from these relative measurements is $O(2^{N^2})$. It can be seen that the number of integration architectures will increase exponentially with the increase of cluster scale.

Then, let us compare the optimization problem complexity with that of the well-known Travelling Salesman Problem (TSP) [35]. For the optimization problem above, the increasing rate of the number of feasible solutions with the increase of vehicles is computed by:

$$Rate = \frac{O(2^{(N+1)^2})}{O(2^{N^2})} = O\left(2^{2N+1}\right) \tag{16}$$

In contrast, the number of feasible solutions for TSP is $O(N!)$ with N being the number of nodes. Therefore, the corresponding increasing rate for TSP is:

$$Rate = \frac{O((N+1)!)}{O(N!)} = O(N+1) \tag{17}$$

It can be seen that the optimization problem formulated in this work has a much higher computation complexity than the TSP. Therefore, for the optimization problem in Eq. (3), traveling over the whole solution space is impossible. In response, we attempt to use MOSA to solve the optimization problem.

4.2. Procedure of MOSA

Multi-objective simulated annealing is a stochastic global optimization method based on solid annealing mechanism [36]. For high-complexity optimization, MOSA is more suitable than other random search methods, e.g., genetic algorithm, for its higher computation efficiency. Besides, it has been proved that MOSA can find the global optimal solution with probability 1 after a sufficient number of iterations [37]. The basic idea of MOSA is illustrated as follows. First, an initial solution is generated randomly and becomes the initial Pareto solution set. Then, the initial solution is perturbed to generate a new solution. And by comparing the new solution with the existing Pareto solution set, we can update the Pareto solution set. The above process is repeated until the termination condition is satisfied. The pseudo-codes of MOSA are shown in Algorithm 1.

Algorithm 1 MOSA.

Define
$\boldsymbol{x}$ — optimization variable
$\boldsymbol{F_o}(\boldsymbol{x})$, $\boldsymbol{F_c}(\boldsymbol{x})$ — objective functions and constraint functions
Begin
1 Generate an initial solution $\boldsymbol{x}$ randomly
2 Compute $\boldsymbol{F_o}(\boldsymbol{x})$ and $\boldsymbol{F_c}(\boldsymbol{x})$
3 **while** the constraint is not satisfied
4 Re-generate an initial solution $\boldsymbol{x}$ randomly
5 Compute $\boldsymbol{F_o}(\boldsymbol{x})$ and $\boldsymbol{F_c}(\boldsymbol{x})$
6 **end while**
7 Take the solution $\boldsymbol{x}$ as the initial Pareto solution set
8 **while** the stopping criteria of outer iteration is not satisfied
9 **while** the stopping criteria of inner iteration is not satisfied
10 Apply a disturbance to the current solution $\boldsymbol{x}$ to get a new solution $\boldsymbol{x}'$
11 Compute $\boldsymbol{F_o}(\boldsymbol{x}')$ and $\boldsymbol{F_c}(\boldsymbol{x}')$
12 Update the Pareto solution set by comparing $\boldsymbol{F_o}(\boldsymbol{x}')$ with $\boldsymbol{F_o}(\boldsymbol{x})$
13 **if** $\boldsymbol{x}'$ gets into the Pareto solution set
14 Accept $\boldsymbol{x}'$ as the current solution $\boldsymbol{x}$
15 **else**
16 Compute the Metropolis acceptance probability P_{ACC}
17 **if** $\delta < P_{ACC}$ (δ is a random number within (0, 1))
18 Accept $\boldsymbol{x}'$ as the current solution $\boldsymbol{x}$
19 **end if**
20 **end if**
21 **end while**
22 Lower the temperature T
23 Select a solution randomly from the Pareto solution set as the current solution $\boldsymbol{x}$
24 Reset the inner iteration
25 **end while**
End

4.3. Encoding

In this work, the optimization variables are expressed by three matrices, i.e., SC, MT, and FT. To solve the optimization problem, we first need to encode the optimization variables with 1-dimensional sequences. Three encoding sequences are defined below to represent the three matrices, respectively.

For sensor configuration, the length of the encoding sequence is $3N$, because the size of SC is $3\times N$. However, note that each UWB sensor installed on the MAS is assumed to be of the same level, and thus an addition encoding digit is needed to represent this constraint. Consequently, the sensor configuration is actually encoded by a sequence with $(3N+1)$ elements.

Specifically, the encoding criterion is given as: the $(3j$-2)th ($j = 1, 2, ..., N$) element represents the level of GNSS receiver in vehicle j; the (3j-1)th (j=1, 2, ..., N) encoding element is either 0 or 1 to indicate whether vehicle j is equipped with UWB; the $(3j)$th ($j = 1, 2, ..., N$) element represents the level of vision sensor in vehicle j; and the last element is the level of the UWB sensors. Fig. 5 gives an illustrative example of this encoding criterion.

For measurement topology, the length of the encoding sequence is $3N^2$. However, because the diagonal entries in MT$(:,:,2)$ and MT$(:,:,3)$ are all 0, $2N$ elements can be removed from the sequence. Therefore, the length of the encoding sequence for MT is actually $3N^2 - 2N$. Fig. 6 shows an example of the encoding process for a measurement topology.

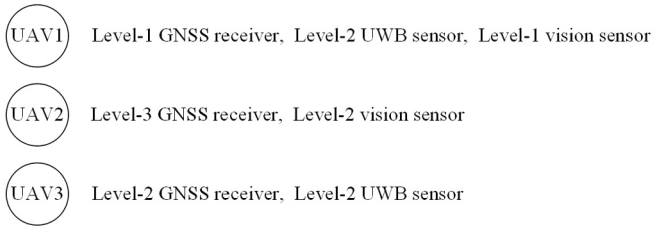

Fig. 5. The encoding sequence of sensor configuration. The color of elements in SC and encoding sequence indicates the type of sensor. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

Finally, the fusion topology is encoded with a sequence whose length is N^2. Nevertheless, it is assumed that the state of each agent is estimated on and only on one filter, leading to only N nonzero elements in FT. Consequently, there is actually only N elements in the encoding sequence, and the ith element indicates the number of the filter that estimates the states of vehicle i. For example, the encoding sequence is (1, 3, 3) for the fusion topology:

$$FT = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}. \tag{18}$$

4.4. Neighborhood transformation operator

In Algorithm 1, the disturbance (Line 10) applied to the current solution is a neighborhood transformation. It transforms the current solution to a new solution in the neighborhood based on the designed neighborhood function below:

$$f_{NBHD}(\boldsymbol{x}) = \boldsymbol{x}', \tag{19}$$

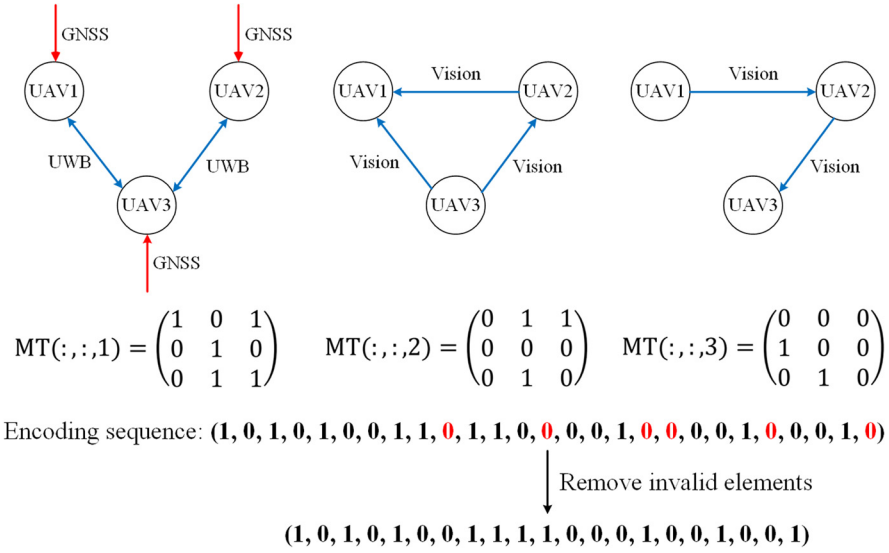

Fig. 6. The encoding sequence of measurement topology.

where f_{NBHD} is the neighborhood function, $\boldsymbol{x}$ is the current solution, and $\boldsymbol{x}'$ is the new solution after disturbance.

One of the commonly-used neighborhood transformation methods in heuristic algorithm is to transform or translocate one single element or fragment in the encoding sequence probabilistically. The basic principle of the neighborhood transformation method used in this work is briefly described as follows. First, the neighborhood transformation probability P_{NBHD} is set. Then, for each element in the sequence, generate a random number within (0, 1). If this number is smaller than P_{NBHD}, the corresponding element will be changed as follows. If this element is a 0-1 variable, it will be transformed from the original value to the other one. And if this element can take several different values, it will be randomly transformed from the original one to one of the other values.

4.5. Metropolis criterion

In Algorithm 1, the Metropolis acceptance probability (Line 16) is given by

$$P_{ACC} = \exp\left(-\frac{E(\boldsymbol{x}_{\text{new}}) - E(\boldsymbol{x}_{\text{current}})}{T}\right), \tag{20}$$

where T is the current temperature, $E(\boldsymbol{x}_{\text{new}})$ and $E(\boldsymbol{x}_{\text{current}})$ are "internal energy" corresponding to the new solution and current solution, respectively, which are determined by the values of objective functions.

For multi-objective optimization problems, $E(\boldsymbol{x})$ is usually expressed as the weighted sum of multiple objective functions. In this paper, a normalized method based on sampling is developed to determine the weights for sensor cost, communication cost, and computation complexity. First, generate twenty feasible solutions that satisfy the constraint in a random way. Then, for each objective function, compute the standard deviation with these 20 samples. Finally, the weights of each objective function can be set to the reciprocal of the corresponding standard deviation. This normalization method can balance the magnitude of the three objective functions in $E(\boldsymbol{x}_{\text{new}}) - E(\boldsymbol{x}_{\text{current}})$, avoiding the situation that the acceptance probability is too low due to the excessive magnitude of one objective function and the search of solution space falls into the local optimum.

Table 4
Simulation settings.

Parameter	Value (SI unit)
Number of UAVs	3
Sampling interval	0.1
Dynamic model	2-dimensional Constant Acceleration (CA) model [38]
State vector	$\boldsymbol{x}_i = [p_x^i, V_x^i, a_x^i, p_y^i, V_y^i, a_y^i]^{\mathrm{T}}$ $(i = 1, 2, 3)$, where p, v, a and i denote position, velocity, acceleration and UAV number, respectively.
Initial state (truth)	$[1, 0, 1, 1.5, 1, 1, 2, 0, 1, 0, 1, 1, 3, 0, 1, 2, 1, 1]^{\mathrm{T}}$
Initial state (estimated)	$[3, 2, 3, 3.5, 3, 3, 5, 2, 3, 2, 3, 3, 5, 2, 3, 4, 3, 3]^{\mathrm{T}}$
Initial state error covariance	$5 \times \boldsymbol{I}_{18}$, where $\boldsymbol{I}_n$ denotes a $n \times n$ identify matrix
Acceleration noise covariance	$q_{ax,i} = q_{ay,i} = 0.01$, UAV $i = 1, 2, 3$.
Navigation performance requirement	Baseline scenario 1: $0.23 \times \boldsymbol{I}_6$;
	Baseline scenario 2: $0.17 \times \boldsymbol{I}_6$
Weights of two baseline scenarios	Baseline scenario 1: 0.7;
	Baseline scenario 2: 0.3
P_{NBHD}	0.35
Initial temperature	1000
Minimum temperature	0.5
Cooling strategy	$T(k) = 0.95T(k-1)$

5. Simulation results

Simulations are carried out in this section to validate the proposed model and algorithm. The simulations are based on an example case, where the MAS is composed of three UAVs. The flight mission profile consists of two baseline scenarios, in which the navigation performance requirement is different. Table 4 gives the detailed simulation settings.

The optimization problem and its solver are established with the settings in Table 4. And after solving this problem using the MOSA algorithm, 38 Pareto optimal solutions are found. Table 5 shows the values of the three objective functions for each solution. It can be seen that all the solutions satisfy the Pareto dominance relationship, which proves the reasonability of the results.

Then, as shown in Fig. 7, the 38 solutions are plotted in a 3D space, and they are fitted to form a surface in the 3D space (i.e., Pareto front). It is obvious that the shape of the fitted sur-

Table 5
The values of the objective functions for the Pareto solution set.

Number	Sensor cost (CNY)	Communication cost	Computation complexity
1	1750	9.0	820.8
2	1550	6.0	1231.2
3	1800	3.2	1209.6
4	1950	9.0	500.4
5	1750	5.3	1159.2
6	1750	5.0	1234.8
7	1800	7.5	820.8
8	1700	7.8	907.2
9	1750	12.2	486.0
10	1850	10.9	486.0
11	1950	4.1	820.8
12	2050	5.9	468.0
13	1950	9.8	424.8
14	1800	5.7	979.2
15	1550	5.6	1332.0
16	1650	8.5	1134.0
17	1850	6.2	770.4
18	1900	9.4	669.6
19	1750	10.6	756.0
20	1950	5.8	720.0
21	2350	14.4	342.0
22	1500	10.6	1458.0
23	1800	11.5	453.6
24	1950	2.9	1108.8
25	1850	5.0	1159.2
26	1650	5.0	2268.0
27	1750	8.1	900.0
28	2150	4.1	644.4
29	1700	8.4	871.2
30	1700	7.1	1159.2
31	1550	5.3	1407.6
32	1700	11.5	518.4
33	1900	5.6	979.2
34	1650	12.0	1004.4
35	2200	8.7	424.8
36	2100	19.8	417.6
37	1650	11.0	1018.8
38	1650	18.4	633.6

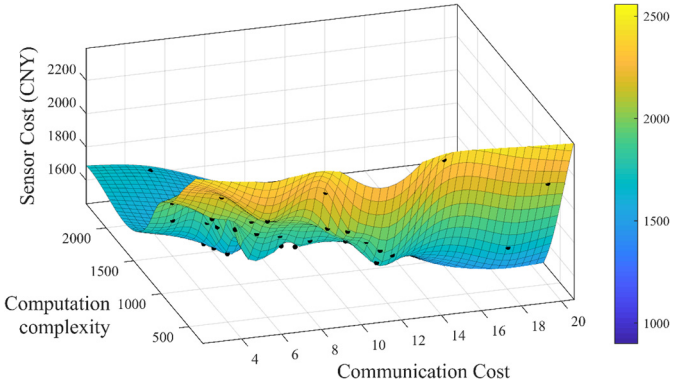

Fig. 7. Pareto front of sensor configuration and integration architecture. The black points are Pareto solutions. The color of the surface indicates the value of sensor cost.

face conforms to the basic properties of a Pareto front. Because the heuristic algorithm (e.g., MOSA) cannot guarantee to find all of the Pareto optimal solutions, the Pareto front can be used to predict the other possible solutions.

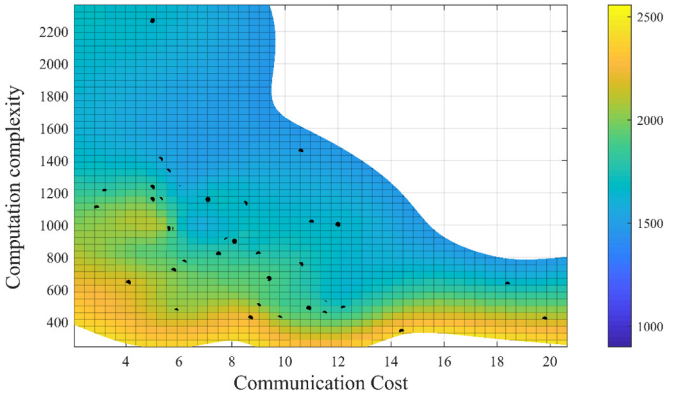

Fig. 8. X-Y view of the Pareto front. The color of the surface indicates the value of sensor cost.

Table 6
The number of different integration architectures.

Baseline scenario	1	2
Fully-decentralized architecture	16	2
Locally-decentralized architecture	15	20
Locally-centralized architecture	6	0
Fully-centralized architecture	1	16

In order to make the results clearer, the X-Y view of Fig. 7 is shown in Fig. 8. From Fig. 8, it can be seen that the higher the sensor cost, the less the communication and computation resources are consumed. This is because the higher sensor cost indicates that the quality of the sensors is better, and in this case meeting the navigation performance requirement needs less sensors, thereby reducing the computation and communication costs. By analyzing the integration architecture of the solutions, we know that only the solution in the upper left region adopts the fully-centralized integration architecture in both two baseline scenarios while the other solutions use a decentralized fusion strategy. Obviously, compared to the centralized integration architecture, the decentralized integration architecture can reduce computation complexity greatly. Besides, the variety of decentralized integration can make it easier for cluster to adapt to different scenarios.

The results shown in the two figures above are obtained by weighting the two baseline scenarios, as explained in Section 2. For a fixed sensor configuration, the integration architectures will be different in different scenarios. To show the difference between the integration architectures in two baseline scenarios, Fig. 9 separately presents the Pareto solution sets for these two scenarios. Besides, Table 6 counts the number of different types of integration architectures for the two scenarios. Two conclusions can be drawn from Fig. 9 and Table 6.

Remark 1. As the navigation performance requirement becomes higher, the communication cost and computation complexity tend to be larger.

Remark 2. The integration architecture for cooperative navigation will transition from a fully-decentralized one to a fully-centralized one with the improvement of required navigation performance.

This is because that using a centralized fusion topology is beneficial to improving the navigation performance as compared to the decentralized integration architectures. This finding also inspires us to investigate the online design method of integration architectures for a fixed sensor configuration, which will be shown in our future work.

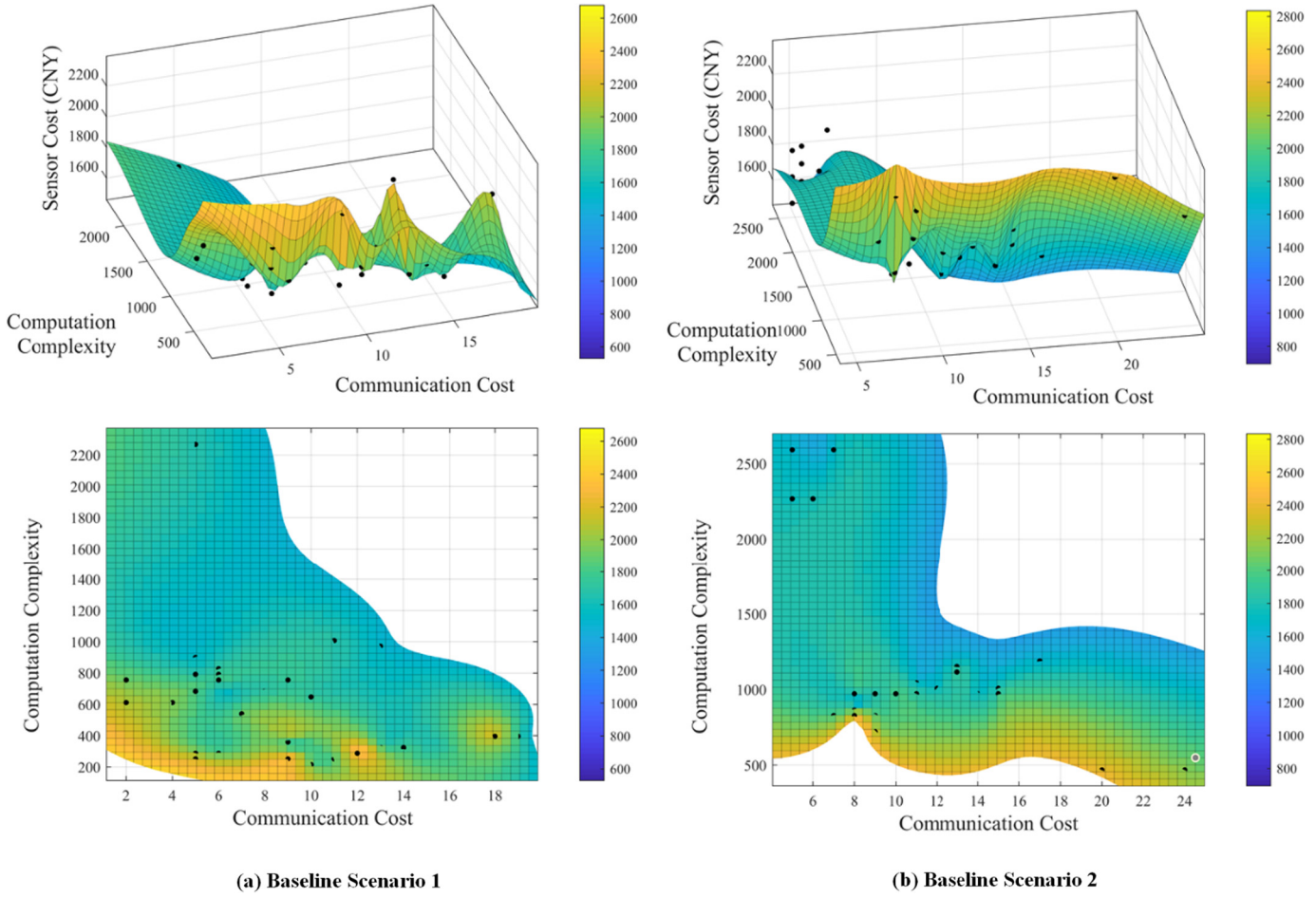

Fig. 9. Pareto solutions for two baseline scenarios. The bottom two subfigures are the X-Y view of the top two subfigures, and the color indicates the sensor cost.

To conclude, the results shown above prove the effectiveness of the proposed model and algorithm. This solution framework is beneficial to the practical applications of multi-agent systems. This is because designing the sensor configuration can cut down the hardware cost, and designing the integration architecture is helpful to reduce the communication load and computation complexity. This work also takes into account the variation in the required navigation performance in different phases of a flight mission, and thus it can help the MAS be adapted to dynamic operation scenarios.

6. Conclusions

This paper provides a method to realize the offline design of sensor configuration and integration architecture for cooperative navigation in multi-agent systems. The motivation is to reduce sensor cost, communication cost, and computation complexity of the cooperative navigation system while ensuring that the navigation performance requirement is satisfied. This goal is achieved by solving a multi-objective combinational optimization problem via multi-objective simulated annealing. The mathematical model of this optimization problem is established by algebraic graph theory and state estimation method, and the algorithm to solve it is tailored. Finally, simulations are carried out. Through random search in the discrete solution space, a Pareto solution set based on three objective functions is found. And the Pareto solutions reveal the characteristics and application scenarios of different integration architectures.

This work is beneficial to the practical applications of multi-agent systems. It reflects the relationship between sensor cost, communication/computation load and different integration architectures. Therefore, it can help to select a proper integration architecture to reduce navigation cost and unnecessary energy consumption under navigation performance constraint. In the future, our continuous efforts will be paid to the following topics: (a) refining the optimization models, (b) investigating the online design of integration architectures, and (c) developing a more efficient optimization algorithm.

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.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grant Number: 62173227) and Laboratory of Science and Technology on Marine Navigation and Control, China State Shipbuilding Corporation (Grant Number: 2021010108).

References

[1] M. Pipattanasomporn, H. Feroze, S. Rahman, Multi-agent systems in a distributed smart grid: design and implementation, in: 2009 IEEE/PES Power Syst. Conf. Expo. PSCE 2009, 2009, pp. 1–8.

[2] A. Altan, R. Hacıog, Model predictive control of three-axis gimbal system mounted on UAV for real-time target tracking under external disturbances, Mech. Syst. Signal Process. 138 (2020) 106548, https://doi.org/10.1016/j.ymssp.2019.106548.

[3] A. Altan, Performance of metaheuristic optimization algorithms based on swarm intelligence in attitude and altitude control of unmanned aerial vehicle for path following, in: 4th Int. Symp. Multidiscip. Stud. Innov. Technol. ISMSIT 2020 - Proc., 2020, pp. 9–14.
[4] A. Altan, Ö. Aslan, R. Hacıoğlu, Real-time control based on NARX neural network of hexarotor UAV with load transporting system for path tracking, in: 6th Int. Conf. CEIT., Department of Electrical and Electronics Engineering, Zonguldak Bülent Ecevit University, 2018, pp. 1–6.
[5] A. Altan, R. Hacioglu, Modeling of three-axis gimbal system on unmanned air vehicle (UAV) under external disturbances, in: 25th SIU, IEEE, 2017, pp. 1–4.
[6] 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, https://doi.org/10.1016/j.ast.2017.09.024.
[7] H. Mokhtarzadeh, D. Gebre-Egziabher, Cooperative inertial navigation, J. Inst. Navig. 61 (2) (2014) 77–94, https://doi.org/10.1002/navi.60.
[8] J. Hyams, M.W. Powell, R. Murphy, Cooperative navigation of micro-rovers using color segmentation, Auton. Robots 9 (1) (2000) 7–16, https://doi.org/10.1023/A:1008963932386.
[9] 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 (2) (2017) 504–513, https://doi.org/10.1109/JSEN.2016.2631478.
[10] B. Allotta, A. Caiti, R. Costanzi, F. Di Corato, D. Fenucci, N. Monni, L. Pugi, A. Ridolfi, Cooperative navigation of AUVs via acoustic communication networking: field experience with the Typhoon vehicles, Auton. Robots 40 (7) (2016) 1229–1244, https://doi.org/10.1007/s10514-016-9594-9.
[11] L. Zhang, D. Xu, M. Liu, W. Yan, Cooperative navigation of multiple AUVs using moving long baseline, Robotica 31 (6) (2009) 581–585+593, https://doi.org/10.13973/j.cnki.robot.2009.06.017.
[12] 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, https://doi.org/10.1177/1729881420932717.
[13] 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) 106068, https://doi.org/10.1016/j.ast.2020.106068.
[14] X. Liu, S. Xu, Multi-UAV cooperative navigation algorithm based on federated filtering structure, in: 2018 IEEE CSAA Guid. Navig. Control Conf. CGNCC 2018, IEEE, 2018, pp. 18–22.
[15] S.I. Roumeliotis, G.A. Bekey, Distributed multirobot localization, IEEE Trans. Robot. Autom. 18 (5) (2002) 781–795, https://doi.org/10.1109/TRA.2002.803461.
[16] M. Ouimet, D. Iglesias, N. Ahmed, S. Martínez, Cooperative robot localization using event-triggered estimation, J. Aerosp. Inform. Syst. 15 (7) (2018) 427–449, https://doi.org/10.2514/1.I010600.
[17] L.C. Carrillo-Arce, E.D. Nerurkar, J.L. Gordillo, S.I. Roumeliotis, Decentralized multi-robot cooperative localization using covariance intersection, in: IEEE Int. Conf. Intell. Robot. Syst., 2013, pp. 1412–1417.
[18] N. Trawny, S.I. Roumeliotis, G.B. Giannakis, Cooperative multi-robot localization under communication constraints, in: Proc. - IEEE Int. Conf. Robot. Autom., IEEE, 2009, pp. 4394–4400.
[19] 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) (2017) 120–136, https://doi.org/10.1017/S0373463316000436.
[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 (12) (2018) 3964–3980, https://doi.org/10.1109/TITS.2018.2794405.
[21] G.M. Hoang, Cooperative multisensor localization for connected vehicles, Ph.D. Thesis, TELECOM ParisTech, 2018.
[22] H. Ko, B. Kim, S.H. Kong, GNSS multipath-resistant cooperative navigation in urban vehicular networks, IEEE Trans. Veh. Technol. 64 (12) (2015) 5450–5463, https://doi.org/10.1109/TVT.2015.2481509.
[23] 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 (4) (2020) 403–416, https://doi.org/10.6125/JoAAA.202012_52(4), p. 05.
[24] 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) (2021) 21–36, https://doi.org/10.1049/rsn2.12023.
[25] 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, https://doi.org/10.1109/TAES.2017.2691921.
[26] M.A. Caceres, F. Sottile, R. Garello, M.A. Spirito, Hybrid GNSS-ToA localization and tracking via cooperative unscented Kalman filter, in: IEEE Int. Symp. Pers. Indoor Mob. Radio Commun, PIMRC, IEEE, 2010, pp. 272–276.
[27] Z. Song, K. Mohseni, FACON: a flow-aided cooperative navigation scheme, in: IEEE Int. Conf. Intell. Robot. Syst., 2017, pp. 6251–6256.
[28] G. Zhang, H.F. Ng, W. Wen, L.T. Hsu, 3D mapping database aided GNSS based collaborative positioning using factor graph optimization, IEEE Trans. Intell. Transp. Syst. 22 (10) (2020) 1–13, https://doi.org/10.1109/TITS.2020.2988531.
[29] C. Tang, L. Zhang, Y. Zhang, H. Song, Factor graph-assisted distributed cooperative positioning algorithm in the GNSS system, Sensors 18 (11) (2018) 3748, https://doi.org/10.3390/s18113748.
[30] Y. Liu, B. Lian, T. Zhou, Gaussian message passing-based cooperative localization with node selection scheme in wireless networks, Signal Process. 156 (2019) 166–176, https://doi.org/10.1016/j.sigpro.2018.10.023.
[31] S. Wang, X. Zhan, Y. Zhai, J. Shen, H. Wang, Performance estimation for Kalman filter based multi-agent cooperative navigation by employing graph theory, Aerosp. Sci. Technol. 112 (2021) 106628, https://doi.org/10.1016/j.ast.2021.106628.
[32] S. Huang, Q. Guan, C. Fan, Airborne single point kinematic positioning accuracy evaluation for low-cost GNSS receivers, J. Navig. Position. 10 (1) (2022) 97–102, https://doi.org/10.16547/j.cnki.10-1096.20220114.
[33] E.J. Msechu, S.I. Roumeliotis, A. Ribeiro, G.B. Giannakis, Decentralized quantized Kalman filtering with scalable communication cost, IEEE Trans. Signal Process. 56 (8) (2008) 3727–3741, https://doi.org/10.1109/TSP.2008.925931.
[34] M.E. Liggins, D.L. Hall, J. Llinas, Handbook of Multisensor Data Fusion: Theory and Practice, CRC Press, 2009.
[35] G. Reinelt, The Traveling Salesman: Computational Solutions for TSP Applications, Springer, 1994.
[36] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680, https://doi.org/10.1126/science.220.4598.671.
[37] V. Granville, M. Krivanek, J.P. Rasson, Simulated annealing: a proof of convergence, IEEE Trans. Pattern Anal. Mach. Intell. 16 (6) (1994) 0, https://doi.org/10.1109/34.295910.
[38] Y. Bar-shalom, X.R. Li, T. Kirubarajan, Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software, John Wiley & Sons, 2001.