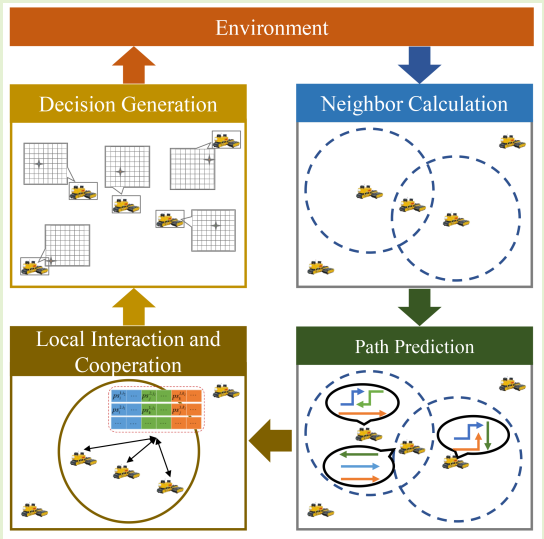
Multi-agent coverage path planning via proximity interaction and cooperation

Lei Jiao, Zhihong Peng, Lele Xi, Shuxin Ding and Jinqiang Cui

***Abstract*— In multi-agent systems, the decision of an agent will be affected by the behaviors of others. Therefore, from the perspective of an agent, the situation is uncertain and random. Inspired by the social behaviors in the biological world, a novel multi-agent coverage path planning algorithm is proposed. Based on the positions of agents, the problem is decoupled, which can effectively reduce the dimension of the decision space. The behavior-guide-point is introduced to guide agents in making decisions, and a new motion mode is presented. To avoid falling into the local optimum, a cooperation mechanism is designed, which can improve the adaptability of the system. Through proximity interaction, the prediction results obtained via the model predictive control (MPC) technology are fused, evaluated, and sorted within the neighborhood, based on which decisions are gained. The proposed algorithm can handle emergencies in unknown environments such as body damage and moving obstacles, and can also be applied to heterogeneous systems. Simulation shows that compared with other algorithms, it has advantages in terms of the makespan and the coverage repetition rate.**

***Index Terms*— multi-agent, coverage path planning, adaptive cooperation, proximity interaction**



I. Introduction

COVERAGE path planning (CPP) [1-2] mainly studies how to make agents plan their paths under the conditions of collision avoidance and obstacle avoidance to achieve the complete coverage in the given environments. There are various applications for this task [3-5], such as search and rescue, window cleaners, demining robots, automated harvesters, and the inspection of complex underwater structures, to name a few.

According to the information awareness degree, the CPP problem can be classified as offline planning [6-8] and online planning [9-11]. The former assumes the environment to be a priori known, while the latter updates the cognition of the environment in a real time based on the feedback from the sensors.

In most applications, task environments are unknown or partially observable. Even if the environmental data can be obtained in advance, the unpredictable development of the situation adds much uncertainty to the task. Therefore, it is essential to study the online CPP problem. An online coverage algorithm called BA* (A* search algorithm) is presented in [9]. A single agent can achieve complete coverage through the boustrophedon motion and the backtracking mechanism. In [10], a hierarchical, hex-decomposition-based CPP algorithm for unknown, obstacle-cluttered environments is proposed. To deal with the CPP problem in dynamic environments, an adaptive CPP approach is developed and it is efficient to respond to the changes in real time [11].

Considering that disaster areas, such as earthquake and nuclear leakage, are extremely complex [12], the cooperation among agents is crucial for the effective execution of search and rescue missions [13-14]. In multi-agent systems, agents can handle complex tasks through local interaction and cooperation [15-16]. Also, they can still complete the task when some of them are damaged [6]. In [17], the content in [9] is extended to a multi-agent version called BoB (an online complete coverage through the boustrophedon motion and

Manuscript received XX XX, 2021; revised XX XX, 2021. This work is supported in part by the Key Program of National Natural Science Foundation of China (NSFC) under Grant U2013602 and U1613225, the National Natural Science Foundation of China under Grant 62088101 and 61720106011, Shanghai Municipal Commission of Science and Technology Project (19511132101). (Corresponding author: Zhihong Peng)

Lei Jiao, Zhihong Peng and Lele Xi are with the School of Automation, Beijing Institute of Technology, Beijing 100081, China, and the State Key Laboratory of Intelligent Control and Decision of Complex System, Beijing 100081, China (e-mail: jiaolei_1029@163.com; peng@bit.edu.cn; 3120170437@bit.edu.cn)

Shuxin Ding is with the Signal and Communication Research Institute, China Academy of Railway Sciences Corporation Limited, Beijing 100081, China, and the Center of National Railway Intelligent Transportation System Engineering and Technology, China Academy of Railway Sciences Corporation Limited, Beijing 100081, China (e-mail: dingshuxin@rails.cn).

Jinqiang Cui is with the Department of Mathematics and Theories, Peng Cheng Laboratory, Shenzhen 518055, China (e-mail: cuijq@pcl.ac.cn)

the backtracking mechanism). In [18], a biologically inspired neural dynamics approach is proposed to guide agents in achieving area coverage cooperatively. With the purpose of balancing the length of the traversal path as much as possible, the workload of each agent is adjusted for several times [19]. Two approximation heuristics for solving the multi-agent coverage path planning (MACPP) problem are developed in [6]. One is a direct extension of the efficient single robot area coverage algorithm, while the other divides the area into several regions and assigns them to the agents. However, there is less discussion about the uncertainty of the situation in the above researches and the envrionments involved are somewhat ideal. Thus, in real applications, especially in disaster areas such as nuclear leakage, the applicability of these algorithms will be constrained.

Unlike traditional scenarios, disaster areas are highly dynamic and uncertain [20-21], such as the collapse caused by the earthquake and the body damage in high-risk environments, which poses a higher challenge to the robustness and adaptability of the system [22]. Agents need to dynamically adjust their decisions based on the observation to cope with the complex and changeable situation [23-24]. However, as the number of agents increases, the problems of multi-agent system have also emerged including frequent collisions, more task time, exponential growth of the decision space, etc. [25]. Hence, it is important to design efficient MACPP algorithms to deal with the uncertainties and emergencies in the process of the task execution.

Solving the online MACPP problem for environments with unexpected changes is challenging, and there are three main reasons for it. First of all, different from the single-agent CPP problem, in multi-agent systems, the decision of each agent will be affected by others. In other words, from the perspective of an agent, the situation is unstable. Secondly, the task environment is enormously uncertain and random, and each agent has only a little prior knowledge of it. Therefore, they need to gradually improve their understandings of the environment and adjust their decisions according to the changes. Thirdly, to achieve complete coverage with the minimal makespan, agents need to interact and cooperate with neighbors to minimize the movement distance, reduce the repeated paths and avoid the collisions.

This paper focuses on the multi-agent cooperation problem in emergency rescue scenarios. Because of the particularity of rescue tasks, especially the nuclear leakage rescue task, agents should be carried to the mission area by a unified platform and traverse the task environment from the same position. To deal with this problem, we propose a novel two-stage MACPP algorithm. Agents can interact and cooperate with neighbors to minimize the makespan. The proposed algorithm can achieve the complete coverage in unknown complex environments under the online condition. It can effectively cope with the emergencies such as body damage and moving obstacles, and is suitable for heterogeneous systems. The contributions and the innovations of this paper are listed below.

(1) **A novel framework for MACPP problem is proposed, which can effectively deal with the dispersion issue among agents with the same starting position.** A two-stage algorithm, called multi-agent coverage path planning based on an MPC technique (MACPP-MPC), is proposed to guide agents in achieving complete coverage in unknown complex environments with the minimum makespan. The first stage is the dispersion stage focusing on the reasonable dispersion of agents. And the second one is the search stage, mainly to achieve an effective coverage of the environment via the interaction and cooperation among agents. In MACPP-MPC, the behavior-guide-point is introduced and a new motion mode is presented.

(2) **A multi-agent interaction and cooperation mechanism is designed, which can effectively improve the efficiency of the system.** The concept of associated individual is introduced to decouple the problem based on the positions of agents, which can effectively reduce the dimension of the decision space. In each neighborhood, the model predictive control technology is adopted to predict the paths. Through proximity interaction, the prediction results are fused, evaluated, and sorted, based on which decisions are gained.

(3) **Compared with the existing algorithms, the performance of the proposed algorithm is superior.** Compared with the existing researches in various scenarios, it is proved that the proposed algorithm can better deal with the MACPP problem in the emergency rescue environments, and is superior in the makespan and the coverage repetition rate. What's more, MACPP-MPC is verified to have high adaptability in dynamic scenarios, such as body damage and moving obstacles.

The rest of this paper is organized as follows. The problem formulation of the MACPP is described in section II. In section III, the MACPP-MPC algorithm is introduced in detail. We verify the effectiveness of the proposed algorithm through simulations in section IV. Finally, concluding remarks and future works are made in section V.

II. PROBLEM FORMULATION

Considering the characteristics of the disaster areas, agents need to start from the same position, no matter for the indoor search (each agent enters into the room from the door) or the outdoor search (agents are carried to the area by a special platform). Thus, it is crucial to deal with the dispersion problem among agents. Rescue agents are mostly driven by electricity and the energy of them are limited. Therefore, it is important to design an efficient cooperation mechanism to minimize the task execution time of each agent as much as possible. For convenience, Table I provides a list of symbols and their definitions.

A. Basic Assumption

In order to simplify the MACPP problem, the following conditions are assumed.

(1) The battery of each agent is sufficient to achieve the task.

(2) Each agent can locate itself accurately.

(3) The observation of each agent has no errors.

(4) Each agent can only interact with the ones on the ground within a limited range.

(5) Agents can exchange information with a central node, such as a UAV.

TABLE I: Nomenclature

Notation	Definition
c_i^k	a decision variable
n	the number of agents
m	the number of points to be covered
o_s	the starting point of agents
r_o	the observation radius
x_i^t	the abscissa of agent i at step t
y_i^t	the ordinate of agent i at step t
o_i^t	the position of agent i at step t, $o_i^t = (x_i^t, y_i^t)$
ξ^t	positions of all the agents at step t
ω_i	the angular velocity of agent i
v_i	the linear velocity of agent i
d_i^k	the distance between agent i and point k
θ_i^{jik}	the turning angle of agent i to cover point k
κ_{pre}	the set of behavior-guide-points of agents
$\mathscr{M}^{\text{vm}}$	the set of grid value matrixes
O^u	the set of uncovered points
O^c	the set of covered points
O^o	the set of points occupied by obstacles
$N_{\text{urt}}^t(i)$	the uncovered reachable points of agent i at step t
$R_i^d(o_k)$	the directional coverage reward associated with point k
$R_i^s(o_k)$	the smoothness reward associated with point k
$R_i^b(o_k)$	the boundary reward associated with point k
ω_d	the weight of the directional coverage reward
ω_s	the weight of the smoothness reward
ω_b	the weight of the boundary reward
psn^t	number of steps for path prediction at step t
ps_i	the prediction result of agent i
pre_a	the index set of the agents belong to Type (c)
pa_i	the path of agent i
$\mathcal{P}$	the set of paths, $\mathcal{P} = \{pa_i \vert i = 1, 2, \cdots, n\}$
smm_i^t	the stop flag of agent i at step t

B. Problem Statement

$S = [x\ y\ \theta]^T$ denotes the state of each agent. Assume that the motion of each agent is discrete, and there are four directions in total including north, south, west and east. The kinematic equation of the agent is described as follows.

$$S' = \begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} v\cos(\theta + \Delta\theta) \\ v\sin(\theta + \Delta\theta) \\ \Delta\theta \end{bmatrix} \tag{1}$$

where (x, y) and (x', y') are the previous and the current coordinates of the agent. θ and $\Delta\theta$ are the heading angle and the change of it. We set speed v as a constant.

In order to avoid unnecessary path repetition, a pause sign smm is set for each agent. While in the dead-point status, if the path repair scheme does not meet the condition shown in (2), the agent will stop moving until its repair scheme meets the restart condition. The restart condition is that all the agents are paused and the repair path of agent i is the shortest. The interval between pause and restart is a pause-restart cycle. Due to the diversity of the environments, agents may have none or multiple pause-restart cycles. The waiting time wt_i of agent i is the sum of the intervals.

$$rpl_i^t \leq |O^u| \,/\, \sum_{i=1}^{n} (1 - smm_i^t) \tag{2}$$

where rpl_i^t is the lenght of the repair path at step t. If agent i is paused at step t, $smm_i^t = 1$. Otherwise, $smm_i^t = 0$.

The task execution time of each agent is affected by many factors such as the path distance, the number of turns, the waiting time, etc. The mathematical expression is shown as follows.

$$T_i = \sum_{j=0}^{m} \sum_{k=1}^{m} ((\frac{d_i^k}{v_i} + \frac{\theta_i^{\text{jik}}}{\omega_i}) \cdot c_i^k \cdot nc_i^k) + wt_i, \quad i = 1, 2, \cdots, n \tag{3}$$

where "0" is the index of the starting point and we call it depot in the following contant. nc_i^k discribes the number of times agent i covers the point k. c_i^k is a decision variable. If agent i covers the point k, then $c_i^k = 1$. Otherwise, $c_i^k = 0$. θ_i^{jik} is the turning angle of agent i to cover point k, as expressed in (4).

$$\theta_i^{\text{jik}} = \begin{cases} 0 & \left|\varepsilon_i^j - \varepsilon_k^i\right| = 0 \\ \pi/2 & \left|\varepsilon_i^j - \varepsilon_k^i\right| = 1 \ or \ \left|\varepsilon_i^j - \varepsilon_k^i\right| = 3 \\ \pi & \left|\varepsilon_i^j - \varepsilon_k^i\right| = 2 \end{cases} \tag{4}$$

where ε_i^j is the motion direction of agent i from point j to the current position, as shown in (5). Similarly, ε_k^i is that of agent i from the current position to point k.

$$\varepsilon_i^j = \begin{cases} 1 & y_i - y_j = 1 \ and \ x_i - x_j = 0 \\ 2 & y_i - y_j = 0 \ and \ x_i - x_j = 1 \\ 3 & y_i - y_j = -1 \ and \ x_i - x_j = 0 \\ 4 & y_i - y_j = 0 \ and \ x_i - x_j = -1 \end{cases} \tag{5}$$

In summary, the makespan of the task is defined as follows.

$$T_{\text{ms}} = \max\{T_1, T_2, \cdots, T_n\} \tag{6}$$

The goal of our research is to minimize the makespan under the condition of detection constraint which means that each point should be covered at least once.

$$\begin{aligned} \min \quad & T_{\text{ms}} \\ \text{s.t.} \quad & \sum_{i=1}^{n} c_i^k \geq 1, k = 1, 2, \cdots, m \\ & c_i^k \in \{0, 1\} \end{aligned} \tag{7}$$

C. Some Properties

In this subsection, we introduce some conceptions.

Definition 1 (Grid Division): A set $\mathcal{A}$ is called a complete segmentation of the environment if its elements, called cells, meet the following two conditions. (1) The area of each element is not redundant, that is, $\tau_i \cap \tau_j = \phi$, $\forall i \neq j$, where $i, j \in \{1, \cdots, |\delta|\}$. (2) The sum of the sub-regions represented by each element is the total coverage of the whole domain, that is, $\mathcal{A} \subseteq \cup_{i=1}^{|\delta|} \tau_i$.

In this paper, $\mathcal{A} = \{\tau_i \subset R^2, i = 1, \cdots, |\delta|\}$ is divided into two types. The one is the unreachable area ψ^o, namely obstacle, and the other is the reachable area $\psi^a = \{o_k \in \mathcal{A} | o_k \cap \psi^o = \phi\}$. ψ^a is composed of two parts, including the searched part ψ_s^a and the unsearched part ψ_u^a. Elements in $\psi^o, \psi_s^a, \psi_u^a$ will be changed during the coverage.

Definition 2 (Grid Value Matrix): Each element in the grid value matrix is the value of the Hamilton distance between each point in the environment and the reference point o_{rp},

as can be seen in Fig. 1. We call the reference point as the behavior-guide-point. It is a virtual one that can be either inside or outside the environment.

Fig. 1: A grid value matrix with the reference point located at the orange graphics.

Definition 3 (R_c-directly-interactive Set): For agent j, only if $\left\|o_j^t - o_i^t\right\| \leq R_c$, we can say agent j belongs to the R_c-directly-interactive set of agent i, where R_c is the interaction radius. ${N_{\mathrm{di}}}^t(i)$ denotes the R_c-directly-interactive set of agent i. As can be seen in Fig. 2(a), agent i_1, agent i_2 and agent i_3 are R_c-directly-interactive to each other.

Definition 4 (R_c-indirectly-interactive Set): If the positions of agent i and agent j satisfy the following two conditions: (1) $\left\|o_j^t - o_i^t\right\| > R_c$, and (2) $\exists\ \tau_1, \tau_2, \cdots, \tau_n \in \mathcal{A}$, satisfy $\left\|o_j^t - o_{\tau_1}^t\right\| \leq R_c, \left\|o_{\tau_1}^t - o_{\tau_2}^t\right\| \leq R_c, \cdots, \left\|o_{\tau_n}^t - o_i^t\right\| \leq R_c$, we can say agent j belongs to the R_c-indirectly-interactive set of agent i. ${N_{\mathrm{ii}}}^t(i)$ denotes the R_c-indirectly-interactive set of agent i. As shown in Fig. 2(b), due to the existence of agent i_1, agent i_2 and agent i_3 are R_c-indirectly-interactive to each other.

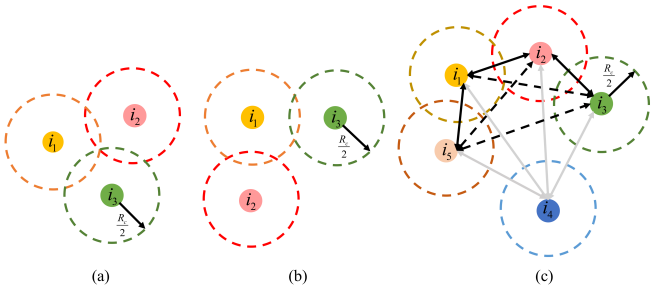


Fig. 2: Schematic diagrams of the position relationship of agents. (a) Agents are R_c-directly-interactive to each other. (b) i_2 and i_3 are R_c-indirectly-interactive to each other. (c) The interactive relationship among agents.

Definition 5 (Associated Individual): Agent j is called the associated individual of agent i if the relationship between agent i and agent j satisfies *Definition* 3 or 4. Agent i can interact with associated individuals directly or indirectly and its decisions will be affected by the prediction results of them. In the following presentation, associate individuals are called neighbors, the set of which is expressed as φ_i. Fig. 2(c) shows the interactive relationship among agents. The black solid and dotted lines indicate the direct interaction and indirect influence among agents, respectively. The gray solid lines indicate non-interaction.

Definition 6 (Complete Coverage): If the point o_k covered by the agent satisfies $\bigcup_k o_k = \psi^a \cup \psi^{o_{\mathrm{m}}}$, then the complete coverage is achieved. Here, $\psi^{o_{\mathrm{m}}}$ represents the points occupied by moving obstacles. The proposed algorithm can deal with the MACPP problem in the presence of moving obstacles. Even if some of the agents are destroyed, the remaining ones can still complete the task.

Definition 7 (Dead-point Status): If all the nearby points of the agent have been covered or occupied by obstacles, the agent is deemed to have entered into the dead-point status. In this case, agent applies the A* algorithm to search for the shortest repair path.

III. Multi-agent Coverage Path Planning Algorithm

MACPP problem can be simplified to the Traveling Salesman problem (TSP) without returning to the initial position. Since the TSP is NP-hard, the MACPP problem in complex environments is also NP-hard [26]. Therefore, we design a heuristic algorithm to solve the problem effectively.

The framework of the proposed algorithm is shown in Fig. 3. According to the distribution of agents, the algorithm is mainly composed of two stages, including the dispersion stage and the search stage.

(1) The former mainly avoids potential conflicts among agents by dispersing their positions. In this stage, each agent make decisions through its own observation, the behavior-guide-point and the reward function, the flow chart of which is shown in the upper part of Fig. 3. After completing the dispersion, the final positions of agents are served as the input of the next stage.

(2) The latter mainly focuses on reducing the redundant paths and shortening the makespan through the interaction and cooperation between neighbors. In this stage, agents in different types make decisions according to different rules, as shown in the lower part of Fig. 3.

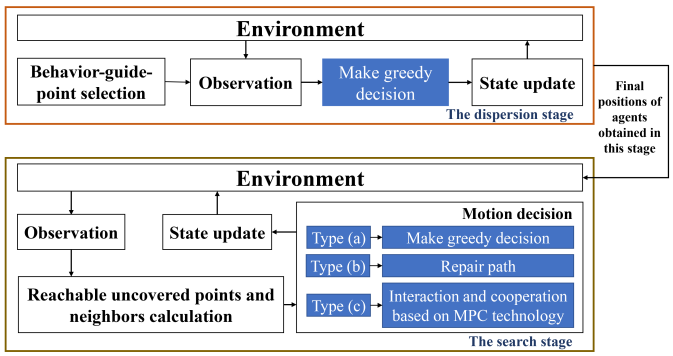


Fig. 3: The framework of the MACPP-MPC algorithm.

A. The Dispersion Stage of the MACPP-MPC

In this stage, each agent iteratively optimizes its decision according to the observation, the behavior-guide-point and the

reward function to achieve a reasonable dispersion. Algorithm 1 illustrates the pseudocode of the dispersion stage.

In line 2 of Algorithm 1, the grid value matrix corresponding to each agent is calculated. Lines 4 to 9 describe the decision-making process of each agent. The function $\mathrm{URPC}(o_i^t, O_i^u)$ calculates the set of uncovered reachable points of each agent. The function $\mathrm{DG}(o_i^{t-1}, o_i^t, N_{\mathrm{urt}}^t(i), \mathscr{M}_i^{\mathrm{vm}}, \omega_d, \omega_s, \omega_b)$ decides the next motion of each agent based on its behavior-guide-point and the reward function (please refer to Section III-D for the mathematical expression of the reward function). For ease of description, we set the same behavior-guide-point for agents in this stage, such as the depot, etc. For odd-numbered agents, the goal is to stay far away from the depot. For even-numbered ones, the goal is opposite. Lines 10 to 13 are the data update processes. Taking into account the discrete nature of the motion, to achieve dispersion of agents, $(n-1)$ iterations are required in this stage.

Algorithm 1 The dispersion stage of MACPP-MPC

Input: $n, m, o_s, r_o, \kappa_{pre}, \omega_d, \omega_s, \omega_b, O^u$;
Output: $\xi^t, \mathcal{P}, \mathscr{M}^{\mathrm{vm}}, O^u, O^c, O^o$;
1: Initialize $t \leftarrow 1, o_i^1 \leftarrow o_s, pa_i \leftarrow o_s (i = 1, 2, \cdots, n)$
$O^c \leftarrow o_s, O^o \leftarrow \phi$;
2: $\mathscr{M}_{\mathrm{vm}} \leftarrow \mathrm{GridValueMatrix}(\kappa_{\mathrm{pre}}, o_s, m)$;
3: **for** $t = 1$ *to* $n-1$ **do**
4: **for** $i = 1$ *to* n **do**
5: $O_i^o \leftarrow \mathrm{UpdateMapData}(o_i^t, O^o, r_o)$;
6: $O_i^u \leftarrow O^u \backslash (O_i^o \cap O^u)$;
7: $N_{\mathrm{urt}}^t(i) \leftarrow \mathrm{URPC}(o_i^t, O_i^u)$;
8: $o_i^{t+1} \leftarrow \mathrm{DG}(o_i^{t-1}, o_i^t, N_{\mathrm{urt}}^t(i), \mathscr{M}_i^{\mathrm{vm}}, \omega_d, \omega_s, \omega_b)$;
9: **end for**
10: $\xi^{t+1} \leftarrow \{o_i^{t+1} | i = 1, \cdots, n\}$;
11: $O^c \leftarrow O^c \cup \zeta^{t+1}, O^o \leftarrow \bigcup_{i=1}^{n} O_i^o$;
12: $O^u \leftarrow O^u \backslash (\zeta^{t+1} \cup O^o)$;
13: $pa_i \leftarrow pa_i \cup o_i^{t+1} (i = 1, 2, \cdots, n)$;
14: **end for**

B. The Search Stage of the MACPP-MPC

In this stage, the MPC technology is adopted to predict the paths of both the agent and the ones belongs to its R_c-directly-interactive set. Algorithm 2 illustrates the pseudocode of the search stage.

Lines 3 to 15 of Algorithm 2 describe the decision rules of the agents, which is mainly divided into the following three types. (1) If $N_{\mathrm{di}}^t(i) = \phi \ \& \ N_{\mathrm{urt}}^t(i) \neq \phi$, agent i belongs to Type (a) and it can directly combine the observation, the behavior-guide-point and the reward function to make decisions, as can be seen in line 9. (2) If $N_{\mathrm{di}}^t(i) = \phi \ \& \ N_{\mathrm{urt}}^t(i) = \phi$, agent i belongs to Type (b) and the A* algorithm is applied to repair the path of it, as shown in line 11. (3) If $N_{\mathrm{di}}^t(i) \neq \phi \ \& \ \mathrm{N^t_{urt}(i)} \neq \phi$, agent i belongs to Type (c) and the index of the agent is stored in the pre_a matrix. The function $\mathrm{DII}(o_i^t, o_j^t (j = 1, \cdots, n \ \& \ j \neq i), r_c)$ obtains the R_c-directly-interactive set of each agent. Lines 16 to 22 describe the decision process of the agents belong to Type (c) (please see Section III-E for detail). The function $\mathrm{PP}(\xi_{\psi_{\mathrm{i}}}^{t-1}, \xi_{\psi_{\mathrm{i}}}^t, O_{\psi_{\mathrm{i}}}^u, O_{\psi_{\mathrm{i}}}^o, \mathscr{M}_{\mathrm{vm}}^{\psi_{\mathrm{i}}}, \omega_d, \omega_s, \omega_b, psn^t)$ returns the prediction results of each agent. The function $\mathrm{PRS}(tp_{\psi_{\mathrm{i}}}^t, \xi_{\psi_{\mathrm{i}}}^{t-1}, \xi_{\psi_{\mathrm{i}}}^t, O_{\psi_{\mathrm{i}}}^u, \mathscr{M}_{\mathrm{vm}}^{\psi_{\mathrm{i}}}, \omega_d, \omega_s, \omega_b)$ evaluates the rewards of the fusion results. Lines 23 to 26 are the data update processes.

Algorithm 2 The search stage of MACPP-MPC

Input: $\xi^t, \mathcal{P}, \mathscr{M}^{\mathrm{vm}}, O^u, O^c, O^o, t, n, r_o, r_c, \omega_d, \omega_s, \omega_b$;
Output: $\mathcal{P}$;
1: Initialize $smm_i \leftarrow 0 (i = 1, 2, \cdots, n), pre_a \leftarrow \phi$;
2: **while** $O^u \neq \phi$ **do**
3: **for** $i = 1$ *to* n **do**
4: $O_i^o \leftarrow \mathrm{UpdateMapData}(o_i^t, O^o, r_o)$;
5: $O_i^u \leftarrow O^u \backslash (O_i^o \cap O^u)$;
6: $N_{\mathrm{urt}}^t(i) \leftarrow \mathrm{URPC}(o_i^t, O_i^u)$;
7: $N_{\mathrm{di}}^t(i) \leftarrow \mathrm{DII}(o_i^t, o_j^t (j = 1, \cdots, n \ \& \ j \neq i), r_c)$;
8: **if** $N_{\mathrm{di}}^t(i) = \phi \ \& \ N_{\mathrm{urt}}^t(i) \neq \phi$ **then**
9: $o_i^{t+1} \leftarrow \mathrm{DG}(o_i^{t-1}, o_i^t, N_{\mathrm{urt}}^t(i), \mathscr{M}_i^{\mathrm{vm}}, \omega_d, \omega_s, \omega_b)$;
10: **else if** $N_{\mathrm{di}}^t(i) = \phi \ \& \ N_{\mathrm{urt}}^t(i) = \phi$ **then**
11: $o_i^{t+1} \leftarrow \mathrm{AstarRepair}(o_i^{t-1}, o_i^t, O_i^u, O_i^o)$;
12: **else**
13: $pre_a = pre_a \cup i$;
14: **end if**
15: **end for**
16: **for** $i = 1$ *to* n **do**
17: **if** $pre_a \neq \phi \ \& \ i \in pre_a \ \& \ |\psi_{\mathrm{i}}| > 1$ **then**
18: $pre_a = pre_a \backslash \psi_{\mathrm{i}}$;
19: $tp_{\psi_{\mathrm{i}}}^t \leftarrow \mathrm{PP}(\xi_{\psi_{\mathrm{i}}}^{t-1}, \xi_{\psi_{\mathrm{i}}}^t, O_{\psi_{\mathrm{i}}}^u, O_{\psi_{\mathrm{i}}}^o, \mathscr{M}_{\mathrm{vm}}^{\psi_{\mathrm{i}}}, \omega_d, \omega_s, \omega_b, psn^t)$;
20: $o_{\psi_i}^t \leftarrow \mathrm{PRS}(tp_{\psi_{\mathrm{i}}}^t, \xi_{\psi_{\mathrm{i}}}^{t-1}, \xi_{\psi_{\mathrm{i}}}^t, O_{\psi_{\mathrm{i}}}^u, \mathscr{M}_{\mathrm{vm}}^{\psi_{\mathrm{i}}}, \omega_d, \omega_s, \omega_b)$;
21: **end if**
22: **end for**
23: $\xi^{t+1} \leftarrow \{o_i^{t+1} | i = 1, \cdots, n\}$;
24: $O^c \leftarrow O^c \cup \zeta^{t+1}, O^o \leftarrow \bigcup_{i=1}^{n} O_i^o$;
25: $O^u \leftarrow O^u \backslash (\zeta^{t+1} \cup O^o)$;
26: $pa_i \leftarrow pa_i \cup o_i^{t+1} (i = 1, 2, \cdots, n)$;
27: **end while**

C. Cooperation Mechanism Based on the Model Predictive Control

In real applications, the propagation medium has an influence on the communication distance between agents. It is assumed that each agent can communicate with a central node, such as a UAV, but can only interact with others on the ground within a certain range.

Inspired by social behaviors in the biological world, we introduce a cooperation mechanism to improve the adaptability of the system. Before modeling the problem, the following propositions are given.

Proposition 1: If agent $j \in \varphi_i$, agent $i \in \varphi_j$. That is, associated individuals are symmetric.

Proof: If agent $j \in \varphi_i$, the relationship of them may be in two cases, including agent $j \in {N_{\mathrm{di}}}^t(i)$ and agent $j \in {N_{\mathrm{ii}}}^t(i)$. For the first case, since it is obvious that $\left\| o_i^t - o_j^t \right\| \leq R_c$, agent $i \in {N_{\mathrm{di}}}^t(j)$. According to *Definition* 5, it is easy to

get agent $i \in \varphi_j$. For the second one, as can be seen from *Definition* 4, $\exists\ \tau_1, \tau_2, \cdots, \tau_n \in \mathcal{A}$ satisfy $\left\|o_i^t - o_{\tau_n}^t\right\| \leq R_c, \left\|o_{\tau_n}^t - o_{\tau_{n-1}}^t\right\| \leq R_c, \cdots, \left\|o_{\tau_1}^t - o_j^t\right\| \leq R_c$, and agent $i \in N_{\text{ii}}^t(j)$. Then, we can get agent $i \in \varphi_j$. In summary, associated individuals are symmetric. ■

Proposition 2: If agent $j \in \varphi_k$ and agent $k \in \varphi_i$, we can get agent $j \in \varphi_i$. That is, associated individuals are transitive.

Proof: Synthesizing *Definition* 3 and *Definition* 4, it is easy to get that if agent $j \in \varphi_k$, $\exists\ p_1, p_2, \cdots, p_{g_1} \in \mathcal{A}$ satisfy $\left\|o_j^t - o_{p_1}^t\right\| \leq R_c, \left\|o_{p_1}^t - o_{p_2}^t\right\| \leq R_c, \cdots, \left\|o_{p_{g_1}}^t - o_k^t\right\| \leq R_c$. For agent $k \in \varphi_i$, we can get that $\exists\ q_1, q_2, \cdots, q_{g_2} \in \mathcal{A}$ satisfy $\left\|o_k^t - o_{q_1}^t\right\| \leq R_c, \left\|o_{q_1}^t - o_{q_2}^t\right\| \leq R_c, \cdots, \left\|o_{q_{g_2}}^t - o_i^t\right\| \leq R_c$. Thus, at least one set of points $\{p_1, p_2, \cdots, p_{g_1}, q_1, q_2, \cdots, q_{g_2}\}$ satisfy $\left\|o_j^t - o_{p_1}^t\right\| \leq R_c, \left\|o_{p_1}^t - o_{p_2}^t\right\| \leq R_c, \cdots, \left\|o_{p_{g_1}}^t - o_k^t\right\| \leq R_c, \left\|o_k^t - o_{q_1}^t\right\| \leq R_c, \left\|o_{q_1}^t - o_{q_2}^t\right\| \leq R_c, \cdots, \left\|o_{q_{g_2}}^t - o_i^t\right\| \leq R_c$, and agent $j \in \varphi_i$. It proves that associated individuals are transitive. ■

Proposition 3: If agent $j \in \varphi_i$, $\psi_i = \{i, \varphi_i\}$ and $\psi_j = \{j, \varphi_j\}$, we can get $\psi_i = \psi_j$.

Proof: Assume that $\exists\ k \in \psi_i$ satisfies $k \notin \psi_j$. Agent k may be in two cases, including $k = i$ and $k \in \varphi_i$. For the former, according to *Proposition 1*, easy to get $k \in \varphi_j$ and $k \in \psi_j$. This contradicts the assumption. For the latter, since $k \in \varphi_i$ and $i \in \varphi_j$, we can get $k \in \varphi_j$ and $k \in \psi_j$ from *Proposition 2*. This also contradicts the assumption. To sum up, the proposition holds. ■

According to *Proposition 3*, based on the associative relationship among agents, the MACPP problem SP can be divided into g disjoint subproblems, shown as $SP = \{SP_1, SP_2, \cdots, SP_p, \cdots, SP_g\}$, $p = 1, 2, \cdots, g$. Here, $|\vartheta_p|$ denotes the number of agents involved in the subproblem SP_p and $\sum_{p=1}^{g} |\vartheta_p| = n$.

For the subproblem with $|\vartheta_p| = 1$, the agent involved makes decisions by combining its observation and the reward function. For the subproblem with $|\vartheta_p| > 1$, to avoid conflict or local optimization caused by greedy decisions, the next psn^t steps of the agents involved are predicted via the MPC technology. And the next one-step motion of each agent is obtained via fusing, evaluating, and sorting the prediction results. The mathematical expression of subproblem SP_p is shown in (8).

$$\begin{aligned} SP_p : \max_{q \in [1, \xi_p]} & J_p(tp_{\text{fusion}}^q), \quad p = 1, 2, \cdots, g \\ \text{s.t.} \quad & \eta_q^i[t+1] = f(\eta_q^i[t], u_q^i[t]) \\ & \eta_q^i[t] \in \psi^a, \quad t \in [st_{\text{cur}} + 1, st_{\text{cur}} + psn^t] \\ & i \in \vartheta_p \end{aligned} \tag{8}$$

where tp_{fusion}^q is the fusion result q in the subproblem SP_p, the total number of elements in which is $psn^t * |\vartheta_p|$ (please refer to Fig. 6 for the representation of tp_{fusion}^q). ξ_p is the number of the fusion results. $\eta_q^i[t]$ and $\eta_q^i[t+1]$ are the states of agent i at step t and $t+1$, respectively. $u_q^i[t]$ is the decision variable of agent i at step t. st_{cur} is the label of the current step.

The optimization objective of MPC is to maximize the number of the non-repeated uncovered points in the fusion results. For the same agent, its next decision will be determined by the current position and the decision. What's more, the points contained in tp_{fusion}^q must be in the reachable area.

To sum up, the proposed algorithm has a multi-group distributed architecture, as shown in Fig. 4. As mentioned above, the problem studied is decoupled into several subproblems, and each of them corresponds to a group where the intra-group decision is made independently without centralized intervention. After receiving the updated positions and observations from agents, a central node only updates and feeds back the global information without decisions.

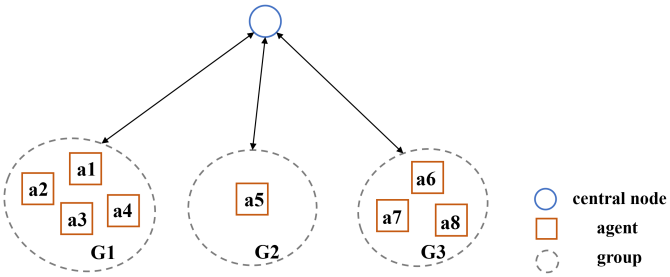


Fig. 4: A multi-group distributed architecture.

D. Reward Function

In the following content, the mathematical expression of the reward function is described, which is composed of three parts, including directional coverage reward, smoothness reward and boundary reward.

1) Directional Coverage Reward: In order to achieve rational distribution of agents and avoid collisions, we introduce the behavior-guide-point to guide agents in making decisions. In each iteration, each agent tries to maximize this reward by selecting one of the uncovered reachable points.

$$R_i^d(o_k) = \frac{g(o_k) - g_{\min}(o_j)}{g_{\max}(o_j) - g_{\min}(o_j)}, \quad j, k \in N_{\text{urt}}^t(i) \tag{9}$$

where $g(o_k)$ is the value of point k in the grid value matrix obtained by taking κ_i as the behavior-guide-point. $g_{\max}(o_j)$ is the maximum one among the values of the points in the uncovered-reachable-point set of agent i. Similarly, $g_{\min}(o_j)$ is the minimum one. Obviously, the value of $(g_{\max}(o_j) - g_{\min}(o_j))$ is a constant, and $R_i^d(o_k) \in [0, 1]$.

2) Smoothness Reward: Reducing the number of turns on the path can shorten the execution time of each agent. Smoothness reward makes the agent continue in the direction of its motion and decide to turn only when it encounters the boundary, the dead-point status, or its neighbors. The expression of $R_i^s(o_k)$ is shown in (10), and $R_i^s(o_k) \in \{0, 0.5, 1\}$.

$$R_i^s(o_k) = \frac{|180 - \theta_i^{\text{ijk}}|}{180} \tag{10}$$

3) Boundary Reward: In order to reduce the redundant path caused by missing boundary points, agents will preferentially select boundary points during the coverage process. We set the boundary reward to increase the agent's attention to the boundary points.

$$R_i^b(o_k) = \frac{n_b^{\max} - n_b^{o_k}}{n_b^{\max}} \tag{11}$$

where $n_b^{\max}$ is the maximum number of the adjacent points, and $n_b^{o_k}$ is the number of that of point k. Easy to get $R_i^b(o_k) \in [0,1]$.

4) Total Reward: In summary, the total reward for agent i moving to an uncovered reachable point k is calculated as (12). Taguchi method [27] is applied to adjust the values of ω_d, ω_s and ω_b in Section IV-A, which will affect the performance of the proposed algorithm.

$$R_i(o_k) = \omega_d \cdot R_i^d(o_k) + \omega_s \cdot R_i^s(o_k) + \omega_b \cdot R_i^b(o_k) \tag{12}$$

For agents without neighbors, that is $N_{\mathrm{di}}^t(i) = \phi \ \& \ N_{\mathrm{urt}}^t(i) \neq \phi$, the next motions of them are decided via selecting the point with the maximum value, as shown in (13). Also, agents belong to Type (c) adopt it to predict paths.

$$k^* = \underset{k \in N_{\mathrm{rut}}^t(i)}{\arg\max}(R_i(o_k)) \tag{13}$$

E. Decision Generation

Next, the decision rules are introduced in detail. While making decisions, agent i are divided in three types, as shown in Algorithm 2, including $N_{\mathrm{di}}^t(i) = \phi \ \ \& \ N_{\mathrm{urt}}^t(i) \neq \phi$, $N_{\mathrm{di}}^t(i) = \phi \ \& \ N_{\mathrm{urt}}^t(i) = \phi$ and $N_{\mathrm{di}}^t(i) \neq \phi \ \& \ \mathrm{N}_{\mathrm{urt}}^{\mathrm{t}}(\mathrm{i}) \neq \phi$.

(1) **Type (a).** Agent i satisfies $N_{\mathrm{di}}^t(i) = \phi \ \& \ \mathrm{N}_{\mathrm{urt}}^{\mathrm{t}}(\mathrm{i}) \neq \phi$. Since there are no neighbors, the decision-making of agent i is only influenced by the observation, the behavior-guide-point and the reward function.

(2) **Type (b).** Agent i satisfies $N_{\mathrm{di}}^t(i) = \phi \ \& \ N_{\mathrm{urt}}^t(i) = \phi$. Since it has entered into the dead-point status, A* algorithm should be adopted to repair its path. To avoid collision during the path repairing process, the **one-step repair** method is adopted, and the repair scheme will be affected by the latest situation.

(3) **Type (c).** Agent i satisfies $N_{\mathrm{di}}^t(i) \neq \phi \ \& \ N_{\mathrm{urt}}^t(i) \neq \phi$. The mutual influence among agents needs to be considered. To calculate the optimal solution efficiently, the path prediction method of the agent is designed as follows.

The path prediction process is explained by taking the three agents i, j, k in the same R_c-directly-interactive set as an example, and the interactive relationship among them is shown in Fig. 2(a). Fig. 5 shows the prediction results of agent i. Since the prediction order of the agents may affect the result, we divided the prediction process into two cases, *individual based* and *step based*. Both of them start from the agent who predicts.

For the first case, as shown in the upper part in Fig. 5, agent i first makes predictions in the order of $i \to j \to k$, and then in the order of $i \to k \to j$. The prediction results are stored in sp_i^1 and sp_i^2, respectively. Here, psn is the number of the prediction step.

For the second case, as shown in the lower part in Fig. 5, agent i first makes predictions in the order of $i \to j \to k \to i \to j \to k \to \cdots$, and then in the order of $i \to k \to j \to i \to k \to j \to \cdots$. Similarly, the prediction results are stored in sp_i^3 and sp_i^4, respectively.

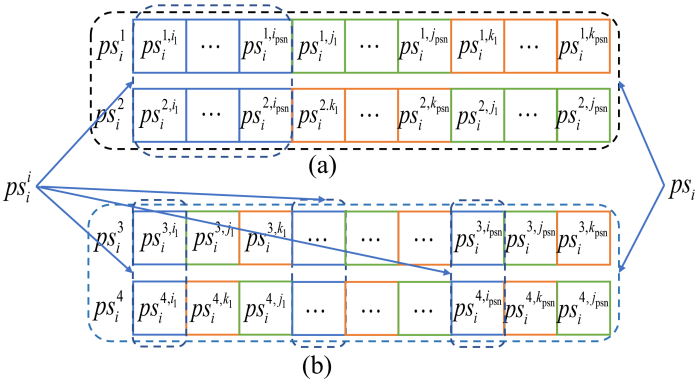


Fig. 5: Prediction results of agent i.

After the predicting process, agents spread the prediction results within their neighborhoods. Taking subproblem SP_p as an example, the fusion result tp_{fusion}^q is shown in Fig. 6. By evaluating and ranking the cumulative rewards of all the fusion results, the one with the maximum reward is selected, and agents involved can obtain the next motions.

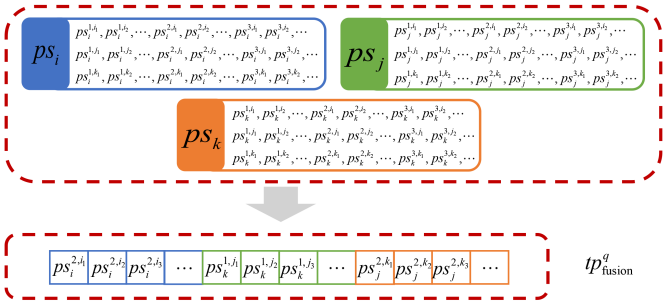


Fig. 6: Local interaction and result fusion.

IV. Simulations and Analysis

This section is devoted to the performance investigation of the proposed algorithm. Firstly, a variety of test cases are generated and the parameter setting of the proposed algorithm is introduced. Next, the effectiveness of some main modules is verified. Finally, the performance of the MACPP-MPC algorithm is validated by comparing it with several existing algorithms in terms of both the makespan and the coverage repetition rate.

We discretize the environment and decompose it into several $1 * 1$ grids. For the convenience of calculation and representation, the motion step of each agent is equal to the length of the grid, that is, the step size of the agent is 1. For simulations in heterogeneous systems, the step size of the slower agent is 1, while that of the faster agent is r_{sp}, where r_{sp} is the speed ratio between agents.

All simulations are carried out in Matlab R2020b environment on a PC with 64-bit operating system, Intel(R) Core(TM) i7-10875H CPU @2.30 GHz.

A. Test Case Generation and Parameter Ajustment

1) Test Case Generation: To the best of our knowledge, there is no recognized benchmark for the MACPP problem. Therefore, the test case generator is used to generate data set for the performance test, as can be seen in Table II. We divide the scope of the environment into three types, including large, medium, and small, and express them as L, M, S, which is the first place in the name of each case. As for the number of agents, we divide it into two types, expressed as M, L, corresponding to the second place. What's more, the proportion of obstacles consists of two grades, that is, less than 30% and more than 30%, denoted as U, B, corresponding to the last digit. For example, the test case $\mathrm{S-L-B_1}$ means that the environment is small, the number of agents is no more than 5, and the proportion of obstacles is less than 30%. Subscript numbers mark different cases of the same type.

TABLE II: Test cases

Index	Data instance	Zone size	Agent num	Obstacle percentage(%)
1	$\mathrm{S-L-B_1}$	25	3	26.08
2	$\mathrm{S-L-U_1}$	50	4	37.56
3	$\mathrm{S-L-U_2}$	50	5	38.32
4	$\mathrm{S-M-B_1}$	40	6	26.31
5	$\mathrm{S-M-B_2}$	50	8	19.20
6	$\mathrm{S-M-U_1}$	50	7	31.76
7	$\mathrm{M-L-B_1}$	85	5	29.87
8	$\mathrm{M-L-B_2}$	60	3	29.83
9	$\mathrm{M-L-U_1}$	70	4	33.06
10	$\mathrm{M-M-B_1}$	70	8	26.45
11	$\mathrm{M-M-B_2}$	90	6	29.11
12	$\mathrm{M-M-B_3}$	80	7	28.56
13	$\mathrm{L-L-U_1}$	100	5	37.06
14	$\mathrm{L-L-U_2}$	150	3	36.97
15	$\mathrm{L-L-B_1}$	120	4	21.92
16	$\mathrm{L-M-B_1}$	120	6	23.57

2) Parameter Adjustment: Since parameters have an impact on the performance of the proposed algorithm, Taguchi method is adopted to adjust them, such as $\omega_d, \omega_s, \omega_b, psn^t$. The method is implemented with orthogonal arrays containing all the information about the factors that affect the performance. The factors involved are divided into two types, including controllable or signal factors and noise factor. Through preliminary tests, we set three alternative values for each parameter, that is, $\omega_d \in \{0.5, 0.75, 1\}$, $\omega_s \in \{0.25, 0.5, 0.75\}$, $\omega_b \in \{0.5, 0.75, 1\}$, $psn^t \in \{2, 4, 6\}$. Thus, L9(3^4) orthogonal table is adopted to adjust them. Our goal is to minimize the makespan. Results and the value of each parameter are shown in Fig. 7 and Table III, respectively.

It can be seen from Table III that the value of psn^t is 6. Each agent belongs to Type (c) predicts the next 6 steps for itself and its neighbors, that is, the total number of elements in tp^q_{fusion} is 6*$|\vartheta_p|$. By evaluating and sorting the fusion results, each agent obtains the next motion.

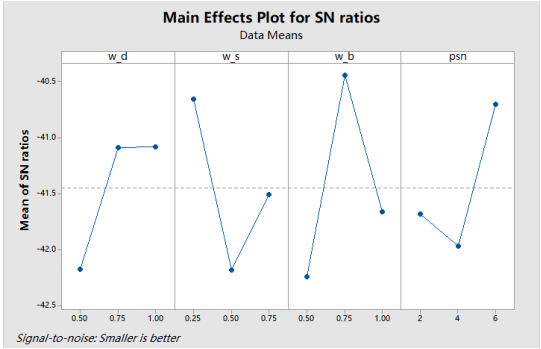


Fig. 7: Parameters gained by Taguchi method.

TABLE III: Parameters

Notation	Value
ω_d	1
ω_s	0.25
ω_b	0.75
psn^t	6

B. Validity Test

In this part, the effectiveness of each main module is tested. Except for the part involved in the comparison, the remaining parts of the comparison algorithms are the same as those of the proposed one. In large-scale problems, the lack of $R_i^s(o_k)$ will lead to a sharp increase in the makespan, which has already shown the importance of $R_i^s(o_k)$. Therefore, there are three variants in total and they are regarded as the ones different from the proposed algorithm.

1) MACPP-I: In this algorithm, the influence of the behavior-guide-point on the decision-making is not considered, that is, the item $R_i^d(o_k)$ is removed from (12).

2) MACPP-II: In this algorithm, the influence of the boundary reward on the decision-making is not considered, that is, the item $R_i^b(o_k)$ is removed from (12).

3) MACPP-III: In this algorithm, each agent only makes greedy decision for the next motion according to the reward function.

Based on test cases in Table II, the makespan T_{ms} and the coverage repetition rate r_{cr} obtained via MACPP-MPC are compared with the results from the above algorithms. The expression of r_{cr} is shown as follows.

$$r_{\text{cr}} = \frac{\sum_{i=1}^{n}\sum_{t=1}^{T_{ms}}((1-\max(0,1-2\cdot rpl_i^t))\cdot(1-smm_i^t))}{\sum_{i=1}^{n} tpl_i} \tag{14}$$

where tpl_i is the total path length of agent i.

As shown in Table IV and Table V, it is easy to get that these three sub-modules all contribute to the improvement of the overall performance.

(1) Since agents start from the same position, the distribution among them is crucial to the coverage task. Reasonable dispersion can effectively enhance the cooperation among agents, thereby avoiding the collision and reducing the repeated path.

TABLE IV: The impact of the main modules on the makespan

Index	MACPP-I	MACPP-II	MACPP-III	MACPP-MPC
1	237.40	259.17	222.17	**212.00**
2	554.10	**455.59**	486.50	**455.59**
3	448.33	419.75	385.02	**365.07**
4	371.05	301.75	284.00	**253.00**
5	327.62	321.44	305.67	**294.33**
6	343.30	361.67	356.17	**320.54**
7	1309.20	1189.00	1302.00	**1186.70**
8	1191.80	1068.50	1099.70	**1048.40**
9	1012.90	988.33	968.74	**944.06**
10	574.94	617.71	590.33	**515.28**
11	1194.90	1152.90	1105.20	**1098.10**
12	871.00	800.33	835.00	**735.83**
13	1520.60	1502.40	1455.50	**1403.80**
14	5740.70	5371.70	5409.90	**5223.00**
15	3333.10	3131.20	3139.70	**3050.30**
16	2245.70	2161.20	2147.00	**2025.40**

TABLE V: The impact of the main modules on the coverage repetition rate(%)

Index	MACPP-I	MACPP-II	MACPP-III	MACPP-MPC
1	10.29	11.52	3.14	**2.74**
2	9.69	1.48	3.02	**0.70**
3	9.33	7.37	6.67	**1.07**
4	11.60	7.89	4.26	**2.15**
5	12.61	6.05	2.46	**2.21**
6	7.39	5.22	6.22	**2.13**
7	6.01	6.36	4.38	**1.92**
8	8.41	6.00	3.81	**1.26**
9	5.39	5.62	3.43	**1.48**
10	10.40	18.32	6.96	**1.61**
11	6.05	4.19	3.87	**2.92**
12	4.60	4.66	8.28	**0.72**
13	4.27	3.59	**1.07**	1.38
14	4.81	4.15	2.52	**1.22**
15	4.56	4.39	2.14	**0.48**
16	4.87	4.58	2.34	**1.23**

(2) Focusing on boundary points can reduce the repeated path caused by missing the corner points.

(3) The interaction among agents and the path prediction mechanism can effectively avoid the decisions falling into the local optimum. The decisions of the agents are forward-looking and global, and they can maximize the overall reward through negotiation. As we can see from the data, the efficiency of the negotiation is particularly prominent when the agent number is large.

C. Algorithms for Comparison

Next, we introduce some well-known algorithms to be compared with.

1) CAC (Coverage with Area Clusting) [6]: This algorithm is an offline one based on the area clusting. Agents adopt the Efficient Complete Coverage (ECC) algorithm [7] while covering the assigned areas.

2) BOB [17]: This algorithm is an online one based on the boustrophedon motion and the backtracking mechanism. Agents construct a global memory matrix and cover the task area in an incremental manner. In this algorithm, the starting positions of the agents do not overlap, so the first stage of the proposed algorithm is applied in it to effectively disperse the agents.

3) Dec-PPCPP [23]: This algorithm is an online one aiming to achieve complete coverage under the condition of moving obstacles. Similar to BOB, the first stage of the proposed algorithm is introduced into it. Since parameters affect the performance of this algorithm, which are not clearly given in the original version, Taguchi method is adopted to adjust them.

D. Results and Analysis

In this part, we test the performance of the proposed algorithm in different scenarios, and compare the results with those obtained by the algorithms mentioned in Section IV-C.

(1) For the complex static environments, we mainly investigate the performance of the proposed algorithm in terms of the makespan and the coverage repetition rate in both homogeneous and heterogeneous systems. As for heterogeneous systems, we select part of the cases in Table II to test whether the proposed algorithm can achieve balancing the workload and the capability of each agent.

(2) For the environments with uncertainties, we verify the adaptability and robustness of the proposed algorithm in homogeneous systems. Uncertain cases mainly involve accidental body damage and the disturbance from moving obstacles. Performance change under the disturbance is investigated.

1) Comparisons in Complex Static Environments: First, we conduct simulations in homogeneous systems. The makespans and the coverage repetition rates are shown in Table VI and VII, respectively. We take case 4 as an example to show the coverage scheme obtained by each algorithm, as shown in Fig. 8. The red star is the starting position and the blue triangles are the destinations of the paths. The black lines are the repeated paths.

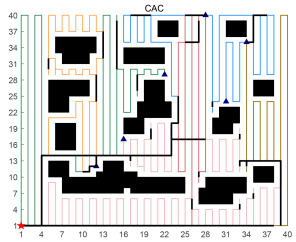
TABLE VI: Comparisons against other algorithms on the makespan.

Index	CAC	BOB	Dec-PPCPP	MACPP-MPC
1	261.50	259.83	230.33	**212.00**
2	573.17	575.00	546.45	**455.59**
3	627.17	479.33	439.50	**365.07**
4	425.83	332.00	295.00	**253.00**
5	505.17	399.33	364.87	**294.33**
6	764.00	411.67	340.67	**320.54**
7	1545.80	1302.50	1266.80	**1186.70**
8	1242.30	1140.70	1108.20	**1048.40**
9	1209.50	1116.30	1031.00	**944.06**
10	1049.50	723.33	633.00	**515.28**
11	1756.00	1215.30	1209.70	**1098.10**
12	889.33	920.67	849.77	**735.83**
13	2478.50	1691.80	1615.20	**1403.80**
14	5815.50	5463.30	**5191.50**	5223.00
15	4005.80	3349.30	3408.50	**3050.30**
16	2739.30	2378.50	2233.60	**2025.40**

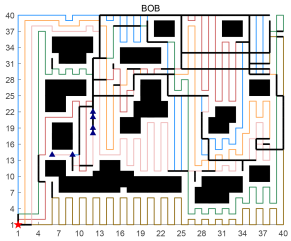
While using CAC in multi-agent systems, the workload may not evenly assigned. There are idle agents in some cases, as a result of which the makespans are large. Since boundary points are not considered in BOB while designing the point

TABLE VII: Comparisons against other algorithms on the coverage repetition rate(%)

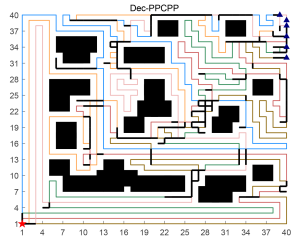
Index	CAC	BOB	Dec-PPCPP	MACPP-MPC
1	21.10	25.16	8.17	**2.74**
2	11.23	20.72	17.06	**0.70**
3	14.45	19.84	16.27	**1.07**
4	16.04	25.52	16.06	**2.15**
5	13.28	28.02	18.22	**2.21**
6	12.98	23.04	17.41	**2.13**
7	6.73	12.89	8.87	**1.92**
8	13.93	13.88	8.31	**1.26**
9	7.94	15.33	10.48	**1.48**
10	6.81	20.51	14.50	**1.61**
11	6.91	12.60	8.29	**2.92**
12	9.61	18.78	10.55	**0.72**
13	9.29	17.05	12.60	**1.38**
14	4.80	6.46	**0.86**	1.22
15	4.38	9.37	8.94	**0.48**
16	4.97	14.88	9.56	**1.23**



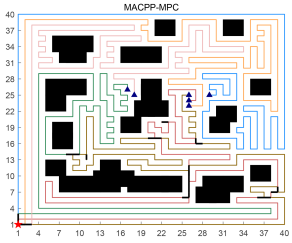
(a) Paths obtained by CAC



(b) Paths obtained by BOB



(c) Paths obtained by Dec-PPCPP



(d) Paths obtained by MACPP-MPC

Fig. 8: Comparisons of the coverage scheme obtained by each algorithm in case 4.

selection rules, agents need to backtrack due to the missing of the boundary ones. As a result, the makespans and the coverage repetition rates are high using BOB. Compared with the former two algorithms, Dec-PPCPP considers the influence of the boundary points, and introduces the dynamic predator avoidance reward to avoid collisions among agents. However, agents are not forward-looking while selecting points, as a result of which this algorithm may fall into local optimum in some status. As shown in these two tables, the proposed algorithm is superior in both the makespan and the coverage repetition rate in complex static environments.

Next, we test the efficiency of the proposed algorithm in heterogeneous systems and the number of the agent is two. It is easy to get from Table VIII, the workload ratio between agents is approximately equal to the speed ratio in each case. The effectivenesses of the behavior-guide-point and the cooperative mechanism are further verified. Even in two-agent scenarios, cooperative mechanism has advantages. It confirms that the proposed algorithm can be applied to heterogeneous systems and the makespan can be minimized through the interaction and the cooperation among agents.

TABLE VIII: Comparisons in heterogeneous systems

Index	r_{sp}	r_w^P	r_m^I	r_m^{III}	r_m^P	$r_{\mathrm{cr}}^I(\%)$	$r_{\mathrm{cr}}^{\mathrm{III}}(\%)$	$r_{\mathrm{cr}}^P(\%)$
3	2	2.00	717.83	639.50	**636.50**	7.53	**1.41**	**1.41**
7	2	2.00	2260.30	**2075.70**	**2075.70**	6.18	3.21	**3.17**
10	4	4.04	978.50	915.08	**912.42**	2.84	1.80	**1.59**
12	3	3.05	1641.70	1586.50	**1563.50**	6.73	2.81	**2.74**
14	3	3.02	909.50	824.67	**822.33**	5.36	1.33	**1.20**

[1] r_{sp} is the speed ratio between agents.
[2] r_w^P is the workload ratio between agents obtained by the proposed algorithm. The workload refers to the number of points traversed by each agent.
[3] r_m^I, r_m^{III}, r_m^P are the makespans obtained by algorithm MACPP-I, algorithm MACPP-III and the proposed algorithm, respectively. Please refer to Section IV-B for algorithm MACPP-I and algorithm MACPP-III.
[4] r_{cr}^I, $r_{\mathrm{cr}}^{\mathrm{III}}$, r_{cr}^P are the coverage repetition rates obtained by algorithm MACPP-I, algorithm MACPP-III and the proposed algorithm, respectively.

2) Comparisons in Uncertain Scenes: Considering that disaster environments, such as fire, earthquake, and nuclear radiation, pose a threat to agents, they may be damaged at any time. Therefore, accidental destruction is the next scenario to be tested. What's more, since the moving objects may interfere with the agent's decisions, the performance of the algorithm under the disturbance condition is tested as well. Simulations in dynamic scenarios are all carried out for homogeneous systems.

First, we select some cases in Table II as the environment inputs, and randomly generate the damage steps of the agents with different indexes, as shown in the second column in Table IX. For example, 2(167) indicates that agent 2 is damaged at $t = 167$. It demonstrates that body damage does not have a significant impact on the overall performance, which further proves the robustness of the proposed algorithm. The necessities of the behavior-guide-point and the cooperative mechanism are proved again.

TABLE IX: Comparisons under the condition of body damage

Index	DAIs	r_m^I	r_m^{III}	r_m^p	$r_{\mathrm{cr}}^I(\%)$	$r_{\mathrm{cr}}^{\mathrm{III}}(\%)$	$r_{\mathrm{cr}}^P(\%)$
2	2(167),3(343)	683.33	644.33	**634.50**	4.54	2.33	**2.03**
3	5(78),1(309),3(365)	663.00	513.33	**469.50**	14.07	2.72	**1.53**
9	2(678),3(793)	1123.00	1086.50	**1037.80**	4.68	1.50	**0.70**
11	4(478),3(704),6(809)	1483.70	1502.00	**1426.20**	3.41	3.20	**1.53**
16	2(565),4(904)	2095.50	1883.50	**1809.50**	6.80	1.90	**0.62**

[1] DAIs stands for the information of the damaged agents.
[2] r_m^I, r_m^{III}, r_m^P are the makespans obtained by algorithm MACPP-I, algorithm MACPP-III and the proposed algorithm under the condition of body damage, respectively.
[3] r_{cr}^I, $r_{\mathrm{cr}}^{\mathrm{III}}$, r_{cr}^P are the coverage repetition rates obtained by algorithm MACPP-I, algorithm MACPP-III and the proposed algorithm under the condition of body damage, respectively.

Next, we test the MACPP-MPC algorithm in moving-obstacle scenarios, and mark the number and the size of the moving obstacles in each scenario, as shown in Table X. Assume that each agent can predict the motions of the obstacles

within its observation range, and the moving obstacles move clockwise [11], as shown in Fig. 9.

TABLE X: Description of moving obstacles

Data instances	Number of moving obstacles
I	2 (1×1)
II	3 (1×1)
III	4 (2×2)
IV	3 (3×3)
V	4 (3×3)

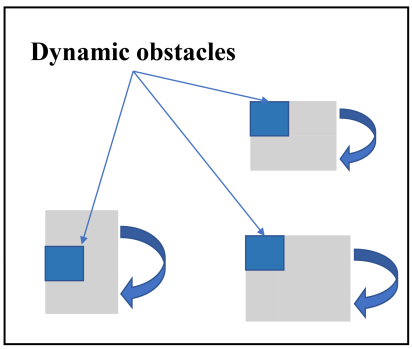


Fig. 9: The schematic diagram of moving obstacles.

Table XI shows the performances of the Dec-PPCPP algorithm and the MACPP-MPC algorithm in moving-obstacle environments. The results are compared with those obtained by these two algorithms in the cooresponding static environments. We take case 1 as an example to show the coverage scheme obtained by each algorithm with and without moving obstacles, as shown in Fig. 10. The black areas are static obstacles and the green areas are moving ones. It proves that the proposed algorithm has high robustness under the disturbance of the moving obstacles due to the introduction of the cooperation mechanism.

TABLE XI: Comparisons in moving-obstacle environments

Index	r_m^{Dec}	r_m^P	δ_m^{Dec}	δ_m^P	$r_{\mathrm{tr}}^{\mathrm{Dec}}(\%)$	$r_{\mathrm{tr}}^P(\%)$	$r_{\mathrm{tr}}^{\mathrm{wd}}(\%)$
1-I	235.50	220.33	**2.24**	3.90	12.95	4.39	**2.74**
4-II	298.67	257.33	**1.20**	1.70	15.81	3.74	**2.15**
9-III	1216.50	977.17	17.99	**3.50**	21.54	1.86	**1.48**
13-IV	1811.50	1468.50	12.15	**4.60**	18.06	2.41	**1.38**
15-V	3605.00	3151.30	5.76	**3.30**	10.40	1.23	**0.48**

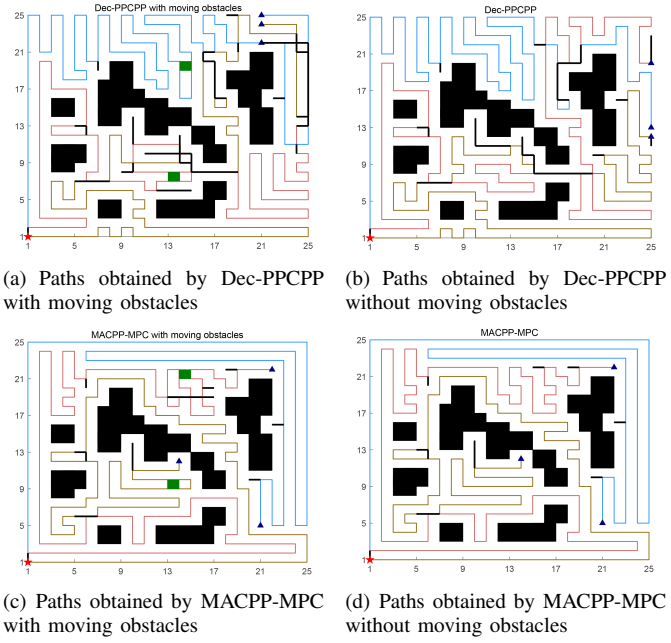
[1] r_m^{Dec}, r_m^P are the makespans obtained by the Dec-PPCPP algorithm and the proposed algorithm with moving obstacles, respectively. δ_m^{Dec} and δ_m^P are the change rates of the makespan.

[2] $r_{\mathrm{tr}}^{\mathrm{Dec}}$, r_{tr}^P are the coverage repetition rates obtained by the Dec-PPCPP algorithm and the proposed algorithm with moving obstacles, respectively. $r_{\mathrm{tr}}^{\mathrm{wd}}$ is the coverage repetition rate obtained by the proposed algorithm without moving obstacles.

V. Conclusion

This paper proposes a novel multi-agent coverage path planning algorithm, MACPP-MPC, which can improve the adaptability and the robustness of the system in unknown complex environments via interaction and cooperation among agents. Agents in the same neighborhood make decisions by fusing, evaluating and ranking the prediction results. Simulations show that the proposed algorithm is superior in the makespan and the coverage repetition rate. Moreover, it can handle sudden changes, such as body damage and moving obstacles, and has better performance than other algorithms. What's more, it can also cope with the cooperation problem in heterogeneous systems and achieve the matching of the workload and the capability of the agent.

In the future, we will focus on the problem of multi-agent cooperative decision-making under the energy constraint. At the same time, it is important to cope with the cooperation problem among heterogeneous agents with different types of capabilities and volumes.



(a) Paths obtained by Dec-PPCPP with moving obstacles

(b) Paths obtained by Dec-PPCPP without moving obstacles

(c) Paths obtained by MACPP-MPC with moving obstacles

(d) Paths obtained by MACPP-MPC without moving obstacles

Fig. 10: Comparisons of the coverage scheme obtained by each algorithm in case 1.

References

[1] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," Robotics & Autonomous Systems, vol. 61, no. 12, pp. 1258-1276, Dec. 2013.

[2] Y. Zhang, S. Li, "Distributed Biased Min-Consensus With Applications to Shortest Path Planning," IEEE Transactions on Automatic Control, vol. 62, no. 10, pp. 5429-5436, Oct. 2017.

[3] J. P. Queralta, J. Taipalmaa, B. C. Pullinen, V. K. Sarker and T. Westerlund, "Collaborative multi-robot systems for search and rescue: coordination and perception," pp. 1-28, Aug. 2020.

[4] T. Miyake and H. Ishihara, "Mechanisms and basic properties of window cleaning robot," IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), pp. 1372-1377, Jul. 2003.

[5] K. Mukherjee, S. Gupta, A. Ray and S. Phoha, "Symbolic analysis of sonar data for underwater target detection," IEEE Journal of Oceanic Engineering, vol. 36, no. 2, pp. 219-230, Apr. 2011.

[6] N. Karapetyan, K. Benson, C. Mckinney, P. Taslakian and L. Rekleitis, "Efficient multi-Robot coverage of a known environment," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1846-1852, Sept. 2017.

[7] A. Xu, C. Viriyasuthee and I. Rekleitis, "Efficient complete coverage of a known arbitrary environment with applications to aerial operations," Autonomous Robots, vol. 36, no. 4, pp. 365-381, Apr. 2014.

[8] L. Zhang, Y. Zhang and Y. Li, "Mobile Robot Path Planning Based on Improved Localized Particle Swarm Optimization," IEEE Sensors Journal, vol. 21, no. 5, pp. 6962-6972, Mar. 2021.

[9] H. H. Viet, V. H. Dang, M. N. U. Laskar and T. C. Chung, "BA*: an online complete coverage algorithm for cleaning robots," Applied Intelligence, vol. 39, no. 2, pp. 217-235, 2013.

[10] X. Kan, H. Teng and K. Karydis, "Online exploration and coverage planning in unknown obstacle-cluttered environments," IEEE Robotics and Automation Letters, vol. 5, no. 4, pp. 5969-5976, Oct. 2020.

[11] M. Hassan and D. Liu, "PPCPP: A predator-prey-based approach to adaptive coverage path planning," IEEE Transactions on Robotics, vol. 36, no. 1 pp. 284-301, Feb. 2020.

[12] S. Singh, A. S. Nandan, A. Malik, N. Kumar and A. Barnawi, "An Energy-Efficient Modified Metaheuristic Inspired Algorithm for Disaster Management System Using WSNs," IEEE Sensors Journal, vol. 21, no. 13, pp. 15398-15408, Jul. 2021.

[13] W. Yuan, N. Ganganath, C. T. Cheng, G. Qing and F. C. M. Lau, "Semi-Flocking-Controlled Mobile Sensor Networks for Dynamic Area Coverage and Multiple Target Tracking," IEEE Sensors Journal, vol. 18, no. 21, pp. 8883-8892, Nov. 2018.

[14] Y. Ji, F. Chen, B. Chen, Y. Wang, X. Zhu and H. He, "Multi-robot collaborative source searching strategy in large-scale chemical clusters," IEEE Sensors Journal, Apr. 2021.

[15] X. Li, Z. Peng, L. Jiao, L. Xi, J. Cai and S. O. Automation, "Online adaptive Q-learning method for fully cooperative linear quadratic dynamic games," Sci China Inf Sci, vol.62, no.12, pp.148-161, 2019.

[16] X. Li, Z. Peng, L. Liang and W. Zha, "Policy iteration based Q-learning for linear nonzero-sum quadratic differential games," Sci China Inf Sci, vol. 62, no. 5, pp. 191-209, 2019.

[17] H. H. Viet, V. H. Dang, S. Y. Choi and T. C. Chung, "BoB: an online coverage approach for multi-robot systems," Applied Intelligence, vol. 42, no. 2, pp. 157-173, Aug. 2015.

[18] C. Luo, S. X. Yang, X. Li and Max Q. -H. Meng, "Neural-dynamics-driven complete area coverage navigation through cooperation of multiple mobile robots," IEEE Transactions on Industrial Electronics, vol. 64, no. 1, pp.750-760, Jan. 2017.

[19] Y. Tang, R. Zhou, G. Sun, B. Di and R. Xiong, "A novel cooperative path planning for multi-robot persistent coverage in complex environments," IEEE Sensors Journal, vol. 20, no. 8, pp. 4485-4495, Apr. 2020.

[20] S. V. Sibanyoni, D. T. Ramotsoela, B. J. Silva and G. P. Hancke, "A 2-D Acoustic Source Localization System for Drones in Search and Rescue Missions," IEEE Sensors Journal, vol. 19, no. 1, pp. 332-341, Jan. 2019.

[21] T. Latif, E. Whitmire, T. Novak and A. Bozkurt, "Sound Localization Sensors for Search and Rescue Biobots," IEEE Sensors Journal, vol. 16, no. 10, pp. 3444-3453, May. 2016.

[22] S. Ding, C. Chen, Q. Zhang, B. Xin and P. M. Pardalos, *Metaheuristics for resource deployment under uncertainty in complex systems*, 1st ed. Boca Raton FL, USA: CRC Press, 2021.

[23] M. Hassan, D. Mustafic and D. Liu, "Dec-PPCPP: A Decentralized Predator–Prey-based Approach to Adaptive Coverage Path Planning Amid Moving Obstacles," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 11732-11739, Jan. 2021.

[24] L. Xi, Z. Peng, L. Jiao and B. Chen, "Smooth trajectory generation of quadrotor for tracking a moving target in cluttered environment," Sci China Inf Sci, vol. 64, no. 7, pp. 1-16, 2021.

[25] M. M. Almasri, A. M. Alajlan and K. M. Elleithy, "Trajectory Planning and Collision Avoidance Algorithm for Mobile Robotics System," IEEE Sensors Journal, vol. 16, no. 12, pp. 5021-5028, Jun. 2016.

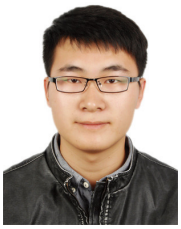
[26] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, pp. 53-66, Apr. 1997.

[27] S. Ding, C. Chen, B. Xin and P. M. Pardalos, "A bi-objective load balancing model in a distributed simulation system using NSGA-II and MOPSO approaches," Applied Soft Computing, vol. 63, pp. 249-267, Feb. 2018.

Lei Jiao received the B.S. degree from Shandong University, Jinan, China, in 2016 and the M.Eng. degree from Beijing Institute of Technology, Beijing, China, in 2018. She is currently working towards the Ph.D. degree in control science and engineering with the School of Automation, Beijing Institute of Technology, Beijing, China. Her current research interests include evolutionary computation, multi-agent cooperation and decision.

Zhihong Peng (M'12) received the B.S. degree from the Hunan University of Science and Technology, Xiangtan, China, in 1995, and the Ph.D. degree from Central South University, Changsha, China, in 2000. She held one post-doctoral appointment at the Beijing Institute of Technology, Beijing, China.

Since 2012, she has been a Professor with the Beijing Institute of Technology. Her current research interests include intelligent information processing, multi-agent cooperation, and optimization and decision.



Lele Xi received the B.E. degree and the M.Eng. degree from Hebei University of Science and Technology, China, in 2014 and 2017, respectively. He is currently working towards the Ph.D. degree in control science and engineering with the School of Automation, Beijing Institute of Technology, Beijing, China.

He was involved in research works as a Visiting Ph.D. Student with the Department of Mechanical and Automation, Chinese University of Hong Kong. His current research interests include motion planning, target tracking, and reinforcement learning, specifically in the area of aerial robotics.

Shuxin Ding received the B.E. degree in automation and the Ph.D. degree in control science and engineering from the Beijing Institute of Technology, Beijing, China, in 2012 and 2019, respectively.

From 2016 to 2017, he was a Visiting Scholar of Industrial and Systems Engineering at the University of Florida, Gainesville, FL, USA. He is currently an Associate Researcher with the Signal and Communication Research Institute, China Academy of Railway Sciences Corporation Limited. His current research interests include railway scheduling, evolutionary computation, multiobjective optimization, and optimization under uncertainty.

Jinqiang Cui received the B.E. and M.Eng. degree in mechatronic engineering from Northwestern Polytechnical University, Xi'an, China, in 2005 and 2008, respectively. He obtained his Ph.D. degree in electrical and computer engineering in National University of Singapore in 2015.

He was a sensor design engineer in HSG-IMIT in Germany from 2008 to 2010. He also served as the chief technology officer for a UAV startup company from 2016 to 2019 before joining Peng Cheng Laboratory. He serves as an Associate Editor for the journal Unmanned Systems. His current research interests include SLAM, 3D reconstruction, intelligent multi-robots systems.