IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 22 March 2023, accepted 17 April 2023, date of publication 26 April 2023, date of current version 3 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3270802

RESEARCH ARTICLE

Joint Task Allocation and Path Planning for Space Robot

YIFEI SUN[1], JIGANG WU[1], (Member, IEEE), AND TONGLAI LIU[2]
[1]School of Computer Science and Technology, Guangdong University of Technology, Guangzhou 510006, China
[2]College of Information Science and Technology, Zhongkai University of Agriculture and Engineering, Guangzhou 510225, China

Corresponding author: Jigang Wu (asjgwucn@outlook.com)

This work was supported in part by the Guangdong Natural Science Foundation under Grant 2023A1515011230, and in part by the Huangpu International Sci&Tech Cooperation Foundation under Grant 2021GH12.

ABSTRACT Space robots have a broad application prospect in the aerospace industry. It is difficult for space robots to keep running for a long time due to limited fuel. In addition, it is impossible to replenish the fuel for space robots at any time due to the unique working environment. Therefore, a proper path is crucial for the effective operation of the space robot. In this paper, we investigate the allocation of exploration tasks and the path planning of space robots jointly. The goal is to minimize the completion latency of exploration tasks. We propose two algorithms named subbranch insertion task allocation (SI-TA) and parallel search task allocation (PS-TA) to solve the problem. We also customize an algorithm named random path planning task allocation (RTA) as the baseline. At last, we implement extensive experiments to demonstrate that proposed algorithms can obtain lower completion latency than RTA. Compared with RTA, the proposed algorithms SI-TA and PS-TA can reduce completion latency by at least 20% and 40%, respectively. Moreover, both algorithms work more stably than RTA.

INDEX TERMS Space robot, path planning, combination algorithm, completion latency.

I. INTRODUCTION

Space robot technology is emerging in the sustainably developed space industry [1]. The space robot can perform various space tasks such as area exploration and sample collection without astronauts [2], [3]. Generally, space robots can be categorized into on-orbit robots and exploration robots. The on-orbit robots mainly provide various on-orbit services. The exploration robots usually undertake extravehicular exploration and base construction, etc.

Path planning is one of the critical techniques for space robots [4], [5]. Path planning has always been a research hotspot in many fields. As early as 1994, some scholars have already systematically studied the classical path planning problem [6]. According to the availability of information of the global environment, path planning can be simply divided into global path planning, and local path planning [7], [8]. Global path planning is usually implemented on the premise that the information of the global environment has been known. At present, the three most established global path planning algorithms can be divided into intelligent bionic algorithm, graph search algorithm, and random sampling algorithm [9]. Global path planning can provide the optimal path for space robots in most cases [10]. Local path planning, also known as online path planning, performs well in dynamic environments [11]. Local path planning can adaptively generate the real-time optimal trajectory when obstacles occur between the source and destination [12].

It is a challenge to develop the strategies of path planning for space robots due to the limited fuel and the special working environment [13]. The path planning problem is a typical complex nonlinear optimization problem. Therefore, there are many heuristic algorithms, such as simulated annealing algorithm [14], fuzzy logic algorithm [15], etc., which are widely used for path planning. Besides, the bionic algorithms have been widely adopted to solve complex optimization problems since their appearance [16]. Researchers have proposed different biological-based algorithms to tackle path

The associate editor coordinating the review of this manuscript and approving it for publication was Yingxiang Liu.

planning for space robots, such as the genetic algorithm [17]. In recent years, with the widespread application of artificial intelligence, many machine learning algorithms and deep learning algorithms have emerged in the research on path planning [18], [19]. Furthermore, some existing works have designed new algorithms to solve the path planning problem for space robots by integrating biological-based algorithms with other algorithms [20]. However, these works mainly focus on the path planning problem for a single space robot instead of multi-space robots.

In this paper, we mainly investigate joint task allocation and path planning for multiple space robots. In order to save fuel for space robots, we assume that each area only needs to be explored once, which can also reduce the overall task completion latency. Multiple space robots set out simultaneously from their respective starting areas to perform exploration tasks. These space robots assemble at a pre-defined destination area after all areas are explored. The most similar problem to this paper is the Multi-Depots Vehicle Routing Problem (MD-VRP) [21]. In MD-VRP, multiple vehicles start from different depots, serve a set of customers, and terminate their tours at the destination depots. The time spent by a vehicle moving between two customers is much longer than the time spent serving customers. However, in this paper, the moving time of a space robot is much less than the exploration time. Therefore, the algorithms used for MD-VRP can not be directly applied to our problem. Note that an area may appear on the paths of multiple space robots, but only one space robot undertakes the exploration task of this area. This makes us aware of the importance of a proper task allocation strategy. Furthermore, it is difficult to predict in advance the time spent by a space robot moving between two areas and the time spent by a space robot exploring an area. In this case, only one subsequent area that space robots move to can be determined after each action of space robots including moving and exploring. It is vital to allocate the exploration tasks considerably while optimizing path planning to minimize the exploration latency [22].

To the best of our knowledge, few existing works focus on joint optimization of path planning and task allocation for multi-space robots. We construct a joint optimization problem of path planning and allocation of exploration tasks for multi-space robots. The main contributions are as follows.

- The path planning problem is of NP-hardness. The goal of this paper is to minimize the completion latency of exploration tasks.
- We propose an algorithm for path planning, which can cover all nodes by adjusting the shortest paths of space robots from respective staring areas to the destination area. Then, an allocation algorithm based on the greedy idea is proposed to allocate the exploration tasks to space robots according to the generated paths.
- We also propose an algorithm for path planning based on the parallel search. It simultaneously searches the next areas for all space robots to move to. The proposed task allocation algorithm is utilized to allocate exploration tasks based on the generated paths.
- We customize an algorithm based on the random algorithm and the proposed allocation algorithm as the baseline. Experimental results show that our proposed algorithms outperform the baseline algorithm in terms of completion latency.

The remainder is organized as follows. We summarize the related work in Section II. We then describe the studied problem and model the problem in Section III. In Section IV, we propose two algorithms named PS-TA and ST-TA, respectively. Extensive experiments and analyses are presented in Section V. Finally, the conclusion of this paper is presented in Section VI.

II. RELATED WORKS

Path planning is one of the essential issues of the robots industry [23]. Since the path planning problem is of NP-hardness [24], it is hard for an exponentially complicated traverse algorithm to satisfy the time-limited practical applications. Path planning is also a fundamental problem in mobile robots. It is significant to generate an optimal or sub-optimal paths to enable the efficient operation of robots [25]. Path planning can be roughly divided into real-time path planning [26] and non-real-time path planning [27]. In general, path planning algorithms may be divided into three categories: biologically inspired, combinatorial, and sampling-based [28]. The biological-based algorithms for path planning is a type of intelligent algorithms that simulate the evolutionary behaviors of biology. Combinatorial algorithms for path planning usually integrate the graph search algorithms and workspace representation methods to solve the path planning problem [29]. The sampling-based algorithms are usually utilized to find a path quickly [30].

There are a large number of works on path planning for mobile robots. The authors in [31] and [32] propose efficient algorithms for path planning based on the artificial potential field, membrane computing, and genetic algorithm. These proposed algorithms can quickly generate a safe path for each mobile robot. The authors in [33] propose the algorithm ParaPA based on traditional particle swarm optimization algorithm and the ant colony algorithm to speed up the search speed through three-stage evolution and two fitness functions. Then, ParaPA selects the optimal path with the consideration of security and loss issues. In [34], the authors propose a algorithm based on simulated annealing for path planning of mobile robots. This algorithm adopts the initial path selection method and deletion operation to reduce computational effort. With the development of artificial intelligence, machine learning algorithms and deep learning algorithms are also widely used in path planning for mobile robots. The authors in [35] propose a Q-learning algorithm based on partial guidance of the artificial potential field, which can converge fast and performs well in path planning, both in known environment and unknown environment.

Path planning are always a hot topic in the researches of space robots. The authors in [36] propose a memetic algorithm to generate the optimal path with obstacle avoidance. The memetic algorithm mainly utilizes the genetic algorithm and the simulated annealing algorithm to perform the global search and local search, respectively. In [37], the authors propose an improved artificial bee colony algorithm to generate paths with obstacle avoidance for space robots. The authors in [38] propose a self-adaptive ant colony algorithm. It adopts adaptive selection, adaptive adjustment, and measures of keeping optimization. The algorithm can converge faster than the traditional ant colony algorithm. Except for the biology-based algorithms, an algorithm is proposed in [39] for path planning on the basis of enhanced fuzzy control to generate paths with obstacle avoidance. The algorithm can obtain higher obstacle avoidance performance through speed feedback when obstacles appear continuously. The authors of [40] design a deep deterministic policy gradient method with multi-constrained reward to find the best path while keeping path length, coupling disturbance, and safety in mind. The existing works mentioned above can solve the path planning problem for single space robots using different methods. However, to the best of our knowledge, the cooperation of multiple space robots is less considered.

III. SCENARIO DESCRIPTION AND PROBLEM DEFINITION

In our studied scenario, multiple space robots are required to collaboratively perform an exploration task with multiple discrete areas to explore. We assume that each area must be explored only once to avoid additional fuel consumption. Before starting the exploration, all space robots are at their respective starting nodes. Then, all the space robots set off simultaneously to perform the exploration task. Finally, after exploring all areas, all space robots gather at the pre-specified destination node. Without loss of generality, the destination node is set to the node with the highest serial number. We will describe the serial number later. We jointly study the execution decisions of exploration tasks and the path planning of multi-space robots to minimize total exploration task completion latency. When a space robot first arrives at a certain unexplored area, the space robot must undertake the task of exploring the area. Furthermore, a space robot does not take time to explore the areas other space robots have explored. If some space robots arrive simultaneously in a previously unexplored area, the lower serial number space robot explores this area.

Next, we will introduce the studied scenario and the proposed problem, and then formulate this problem. We define an undirected graph $G = (\mathcal{V}, E)$ to abstractly represent the entire exploration area. We denote $E = \{1, 2, \cdots, e\}$ to represent the paths between areas. Besides, let $V = \{1, 2, \cdots, v\}$ be the area set. For descriptive convenience, node and area are used interchangeably to indicate the same meaning. We can obtain the corresponding graph $G = (\mathcal{V}, E)$ in advance for an exploration task. Then, we number the nodes in a graph. We set the serial number of the starting node to start at 0, that is $0, 1, \cdots, s-1$. For example, assuming there are three starting nodes, we set the serial numbers of these nodes to 0, 1, and 2, respectively. The serial number of a space robot is set to the serial number of the corresponding starting node. Note that the destination is the node with the largest serial number. Let $\lambda_b^{i,j}$ indicate whether the space robot b moves between area i and j.

$$\lambda_b^{i,j} = \begin{cases} 1 & \text{The space robot } b \text{ moves between area } i \text{ and } j \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

When $i = j$, let $\lambda_b^{i,j} = 0$. We can use an order sequence $\langle\cdot\rangle$ to represent the path for space robots. For example, if a space robot is initially located in area 1, then, it goes through area 2 and returns to area 1, then the path of this space robot can be represented as $\langle 1, 2, 1\rangle$. Next, a discriminant operation $\ominus$ is defined as follows. If $\lambda_b^{i,j} = 0$, let $\lambda_b^{i,j} \ominus j = \{\emptyset\}$. When $\lambda_b^{i,j} = 1$, let $\lambda_b^{i,j} \ominus j = \{j\}$. Then, a splicing operation $\otimes$ between sets is also defined. The rule for $\otimes$ is as follows.

$$\{f\} \otimes \{g\} = \langle f, g\rangle. \tag{2}$$

Finally, let $\bigsqcup$ be the operation of set accumulation.

$$\bigsqcup_{a}^{v}\{\phi_a\} = \{\phi_a\} \otimes \{\phi_{a+1}\} \otimes \cdots \otimes \{\phi_v\}. \tag{3}$$

Then, let P_b be the path of space robot b, which can be obtained by the following formula.

$$P_b = \{b\} \otimes (\bigsqcup_{i=b}^{v-1}\bigsqcup_{j=0}^{v-1} \lambda_b^{i,j} \ominus j), \tag{4}$$

Our goal is to achieve minimum completion latency of the exploration task. The time spent by space robot b exploring area i is denoted as τ_b^i. If the multiple paths contain the same area, the allocation of exploration task for this area should be determined carefully. Let γ_b^i indicate whether the space robot b performs the exploration tasks of area i, where $b \in \{0, 1, \cdots, s-1\}$, $i \in \{0, 1, \cdots, v-1\}$.

$$\gamma_b^i = \begin{cases} 1 & \text{The space robot } b \text{ explore area } i \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Let T_b^{exe} denote the time spent by space robot b completing the exploration task.

$$T_b^{exe} = \sum_{i=0}^{v-1} \gamma_b^i \cdot \tau_b^i. \tag{6}$$

Let $\kappa_b^{i,j}$ indicate the time for space robot b moving between area i and area j. Then, let θ indicate the position of a node in P_b, where $1 \leq \theta < |P_b|$. As an illustration, the θ-th node on path P_b can be represented as $P_b[\theta]$. Let T_b^r be the overall moving time of space robot b.

$$T_b^r = \sum_{\theta=1}^{|P_b|-1} \kappa_b^{P_b[\theta], P_b[\theta+1]}. \tag{7}$$

Algorithm 1 Find Shortest Path

```
Input: $s, v, G$
Output: $l$
1:  Initial $l$
2:  Append $s$ to $l$        // Append start node to path $l$
3:  while $l$ does not end with $v$ do
4:      Get the last node $e$ of $l$
5:      Obtain the neighbor node set $C$ of $e$ according to $G$
6:      Calculate the distance $d$ between first node in set $C$
        and $v$
7:      $d_{e,v} = Caldis(e, v)$        // Calculate the distance
        between two nodes
8:      for each $c$ in $C$ do
            $d_{c,v} = Caldis(c, v)$
9:          if $d_{c,v} < d_{e,v}$ and $d_{c,v} < d$ then
10:             $d = d_{c,v}$
11:         end if
12:     end for
13:     Obtain the node $c^*$ corresponding to $d$
14:     Append $c^*$ to $l$
15: end while
16: return $l$
```

Let T represent the overall task completion latency.

$$T = \max_{0 \leqslant b \leqslant s-1} \{T_b^{exe} + T_b^r\}. \tag{8}$$

We can formalize the latency minimization problem as follows.

$$min\, T \tag{9}$$

$$s.t. \sum_{b=0}^{s-1} \sum_{i=0}^{v-1} \gamma_b^i = v. \tag{10}$$

Formula (10) ensures that each area must be explored.

IV. SOLUTIONS

The joint optimization problem can be decomposed into two subproblems, path planning and task allocation. Before solving the task allocation problem, we must find the working path for all space robots. The path of each space robot records the movement sequence from a starting area to the destination. The generated paths must cover all nodes in the given graph. Then, we solve the task allocation problem to minimize the overall completion latency. In particular, we propose two path planning algorithms, Sequential Search Subbranch Insertion (SSSI) and Forward Parallel Search (FPS). After that, a task allocation algorithm named Greedy-based Task Assignment (GRTA) is proposed to generate the execution decision for each exploration area. Finally, we combine SSSI with GRTA to obtain SI-TA and combine FPS with GRTA to obtain PS-TA.

A. SOLUTIONS FOR PATH PLANNING

We next introduce the SSSI and FPS, respectively. Before introducing these two algorithms, we first introduce

Algorithm 2 Sequential Search Subbranch Insertion

```
Input: $G, l_b, b \in \{1, 2, \cdots, s\}$        // $l_b$ is initial path of
       space robots $b$
Output: $l_b$
1:  $w = 0$
2:  Initialize $F_b$
3:  Denote $z$ to indicate state of nodes        // explored or
    unexplored
4:  while There exists paths $b$ where $l_b[w]$ is not destination
    do
5:      $\Phi = \{\emptyset\}$
6:      Record serial number of all space robots that meet the
        conditions in $\Phi$.
7:      for each $\varphi$ in $\Phi$ do
8:          Construct $h_\varphi$        // neighbor nodes set of $l_\varphi[w]$
9:          if There exits nodes that not explored then
10:             Insert the node $l_\varphi[w]$ into the last position of $F_\varphi$
11:             Choose an unexplored node $h'$ in $h_\varphi$
12:             Insert $h'$ into position $l_\varphi[w+1]$
13:             Update $z$
14:         end if
15:         if The successor node $g$ of $l_\varphi[w]$ are contained by $h_\varphi$
            then
16:             if The last node of $F_\varphi$ is in $g$ then
17:                 Insert $g$ into position $l_\varphi[w+1]$
18:                 Remove $g$ from $F_\varphi$
19:             end if
20:         end if
21:     end for
22:     $w = w + 1$
23: end while
24: return $l_b$
```

Algorithm 1. The Algorithm 1 is used to generate the shortest path between two nodes on a graph and usually used to generate the shortest path in many existing works. The algorithm takes the graph G, start node s and destination node v as inputs. Then, we initialize a list l and add s to l. Next, we judge the last node in l and repeat the following operations when the last node e in l is not the destination node. First, the neighbor set C is obtained according to graph G. The distance d between node v and the first node in neighbor set C is calculated. Second, for each node c in C, the distance $d_{c,v}$ between node c and destination node v is calculated. If $d_{c,v} < d_{e,v}$ and $d_{c,v} < d$, we update d by $d_{c,v}$. Last, the node c^* is selected according to the distance d. Node c^* is appended to the l.

However, the paths generated by Algorithm 1 can not cover all nodes. Next, the SSSI algorithm inserts nodes not covered into the shortest paths obtained by Algorithm 1. The details of SSSI are described as follows.

First, Algorithm 2 takes the graph G and the path l_b generated by Algorithm 1 as inputs, where b is the serial number of space robots. The path that does not contain the destination

Algorithm 3 Distance Calculation

```
Input: $G = \{\mathcal{N}, E\}$
Output: $\mathcal{D}$      // distance information
 1: $\mu = 1$
 2: Initial $\mathcal{E}$                    // indicator vector
 3: Initialize $\eta$
 4: while $Sum(\mathcal{E}) < |\mathcal{N}|$ do
 5:   Initialize $\varsigma$
 6:   for $\alpha = 0$ to $|\eta| - 1$ do
 7:     for $\beta = 0$ to $|\mathcal{N}| - 1$ do
 8:       if $\eta[\alpha]$ is a neighbor of $\beta$ then
 9:         if $\mathcal{E}[\beta] == 0$ then
10:           $D[\beta] = \mu$            // record distance
11:           $\mathcal{E}[\beta] = 1$
12:           append the node $\beta$ to $\varsigma$
13:         end if
14:       end if
15:     end for
16:   end for
17:   $\mu = \mu + 1$
18:   $\eta = \varsigma$
19: end while
20: return $\mathcal{D}$
```

node at the current search position w is recorded. Let Φ record the serial numbers of space robots corresponding to these paths. Then, let set h_φ record serial numbers of neighbor nodes of node $l_\varphi[w]$ according to the graph G, where l_φ represents the path of robot φ.

Next, we judge the type of nodes in Φ according to indicator z, where z indicates whether nodes have been explored. If the node $l_\varphi[w]$ is an unexplored node, we append node $l_\varphi[w]$ into the last position of F_φ, where F_φ is the set of bifurcation nodes. We denote a node as a bifurcation node if some nodes in its neighbors are the unexplored nodes. Then, we select an unexplored node h' in h_φ and insert h' into the position $l_\varphi[w+1]$. We next update the indicator z. If all nodes in h_φ are explored, we insert node g into position $l_\varphi[w+1]$, where node g is contained by h_φ. Node g is also the successor node of node $l_\varphi[w]$ and the last node in F_φ. Next, we remove node g from F_φ. Finally, the position w is updated by adding 1. The pseudocode of SI-TA is described in Algorithm 2.

Next, we introduce FPS. The main idea of FPS is to synchronously search the next node to go for each space robot that has not reached the destination. The successful execution of FPS requires preliminary knowledge of the distance from all nodes to the destination. We adapt Algorithm 3 to calculate the distance from each node to the destination node. The distance between two adjacent nodes is set to 1. First, μ represents the distance. The initial value of μ is set to 1. Then, let $\mathcal{E}$ be a list $[\mathcal{E}[0], \mathcal{E}[1], \cdots, \mathcal{E}[t]]$ to indicate whether the distances between a node to the destination node has been calculated. The value of each element in $\mathcal{E}$ is set to 1 or 0, where 1 (0) indicates the distance of node to destination node

Algorithm 4 Forward Parallel Search

```
Input: $G = \{\mathcal{N}, \mathcal{E}\}, D$
Output: $l_i$
 1: Initialize $\rho$
 2: Initialize $l_i$ to record path for $i$
 3: while There are robot $r \in S$ that has not reached destination node do
 4:   for each $r$ in $S$ do
 5:     Get neighbor nodes set $C_r$ of last node $p_r$ of path $l_r$ according to the $G$
 6:     Get unexplored nodes set $W_r$ of $i$ according to $C_r$ and $\rho$
 7:     Calculate the number $\eta_r$ of nodes in $W_r$
 8:     if $\eta_r == 1$ then
 9:       Append this node to path $l_r$
10:       Update $\rho$
11:     end if
12:     if $\eta_r > 1$ then
13:       Construct set $D^r_{max}$ of nodes that are farthest to destination node according to $D$, $W_r$
14:       if $D^r_{max}$ only contains one node then
15:         Append this node to path $l_r$
16:       else
17:         Select a node from $D^r_{max}$ randomly to append to $l_r$
18:         Append the penultimate node on $l_r$ to $l_r$ again
19:       end if
20:       Update $\rho$
21:     end if
22:     if $\eta_r \le 0$ then
23:       while $true$ do
24:         Find node with minimal distance to destination node in $C_r$
25:         Append this node to $l_r$
26:         if $r$ has reached the destination node then
27:           break
28:         end if
29:         Repeat line 5 to 20
30:         break;
31:       end while
32:     end if
33:   end for
34: end while
35: return $l_i$
```

has (not) been calculated. Initially, $\mathcal{E}[t]$ is set to 1. Then, the list η is initialized, and the destination node is first put into η.

If the sum of elements in $\mathcal{E}$ is less than the number of nodes, it means that there exist some nodes, their distances to destination node have not been calculated yet. Next, the list ς is initialized to record the nodes whose distance information will be calculated in the next round. For node α in η, we select neighbor node β whose distance to destination node has not been calculated. Then, the distance between these neighbor

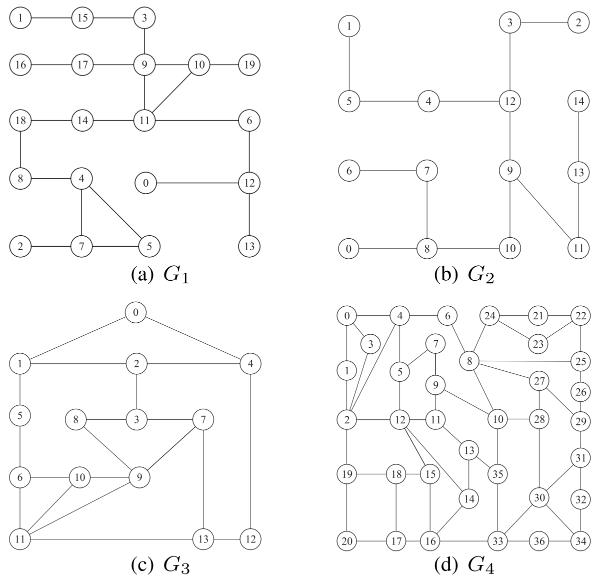


FIGURE 1. Topological graphs for experiment.

nodes to the destination node is set to μ. The corresponding value in the list $\mathcal{E}$ is set to 1 for these neighbors. These neighbors are next added to ς. After that, η is updated by ς and μ is updated by $\mu + 1$ for the next round of calculation. Next, we obtain the path of each space robot by using Algorithm 4. The primary process is summarized as follows.

Step 1: Find all space robots that have not reached the destination node. Then, a set S is constructed to record the serial number of these space robots.

Step 2: Construct a neighbor node set C_r of the last node p_r for each space robot r according to the given G. Each node in C_r may be an explored or unexplored node. We denoted W_r to represent the set of unexplored nodes.

Step 3: Calculate the number η_r of nodes in W_r.

Step 4: Append the only node in W_r to l_r when the value of η_r is 1, where l_r is the path of space robot r.

Step 5: Find all nodes in W_r farthest to the destination node according to distance D. A set D^r_{max} is constructed to record the corresponding serial numbers of nodes.

Step 6: Choose a node from D^r_{max} to add to the last position of l_r. After that, we append the penultimate node on l_r to l_r again.

Step 7: Find out the node in C_r with minimal distance to the destination node and append this node to l_r. If r reaches the destination, we terminate the algorithm. Otherwise, repeat Step4 - Step6.

For the pseudocode of FPS, see Algorithm 4

B. GREEDY-BASED TASK ASSIGNMENT

In this subsection, we propose GRTA based on greedy policy to generate the execution decisions of exploration tasks. The details of GRTA are described in Algorithm 5.

Algorithm 5 Greedy-Based Task Allocation

```
Input: $p_b$
Output: $T$
1: $w = 1, \gamma = p_b[0]$
2: $T = \tau_b^{\gamma}$
3: Let vector $z$ indicate whether nodes are explored
4: Let $m_b$ represent the length of $p_b$
5: while There exists $w \leq m_b$ do
6:     $U = \{\emptyset\}$
7:     for $b = 0$ to $s - 1$ do
8:         if $w \leq m_b$ then
9:             $U = U \cup b$
10:        end if
11:    end for
12:    for each $u$ in $U$ do
13:        if Node $p_u[w]$ is an explored node then
14:            $T_u = T_u + \kappa_u^{p_u[w-1],p_u[w]}$
15:        else
16:            Construct set $\Omega$ to record paths whose last node
               is $p_u[w]$
17:            Select space robot $a$ whose $T_a$ is minimum.
18:            $T_a = T_a + \tau_a^{p_a[w]} + \kappa_a^{p_a[w-1],p_a[w]}$
19:            for Robot $\alpha$ in $\Omega - \{a\}$ do
20:                $T_\alpha = T_\alpha + \kappa_\alpha^{p_\alpha[w-1],p_\alpha[w]}$
21:            end for
22:        end if
23:        Update $z$
24:    end for
25:    $w = w + 1$
26: end while
27: Calculate $T$ utilizing formula (8)
28: return $T$
```

First, let m_b be the length of path p_b and w be the index to indicate the current search position of all paths, where b is the serial number of the space robot. We define U to record the serial numbers of space robots. If $w \leq m_b$, we append b to U. Then, we judge whether the node $p_u[w]$ has been explored for each space robot u in U. If node $p_u[w]$ has been explored, we just compute the moving delay for u arriving at this node. The current working time T_u of space robot u is updated by adding the time $\kappa_u^{p_u[w-1],p_u[w]}$ spent by u moving from node $p_u[w-1]$ to node $p_u[w]$. If node $p_u[w]$ is unexplored, we record the path whose node locate in position w is this node. Then, we construct Ω to record the robot serial number corresponding to these paths. We next choose a space robot a from Ω with a minimum time T_a and compute the exploring time and moving time of a. Then, the working time T_a is updated by adding the time $\tau_a^{p_a[w]}$ spent by a exploring $p_a[w]$ and the time $\kappa_a^{p_a[w-1],p_a[w]}$ spent by a moving from node $p_a[w-1]$ to node $p_a[w]$. After that, we remove a from Ω. For other space robot α in Ω, we update the working time T_α by adding the time $\kappa_\alpha^{p_\alpha[w-1],p_\alpha[w]}$ spent by α moving from node $p_\alpha[w-1]$ to node $p_\alpha[w]$. Finally, we update the

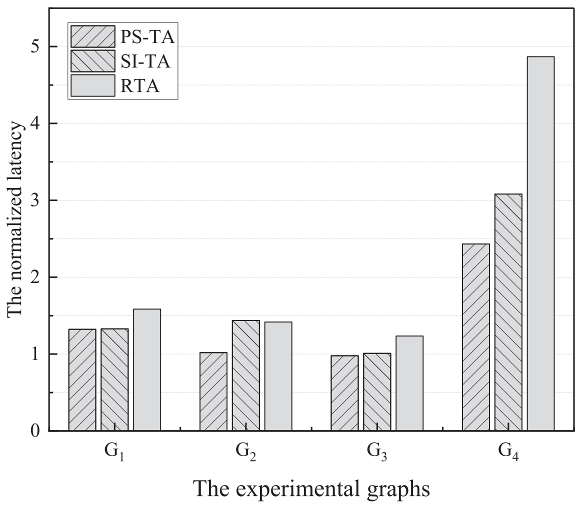


FIGURE 2. Completion latency.

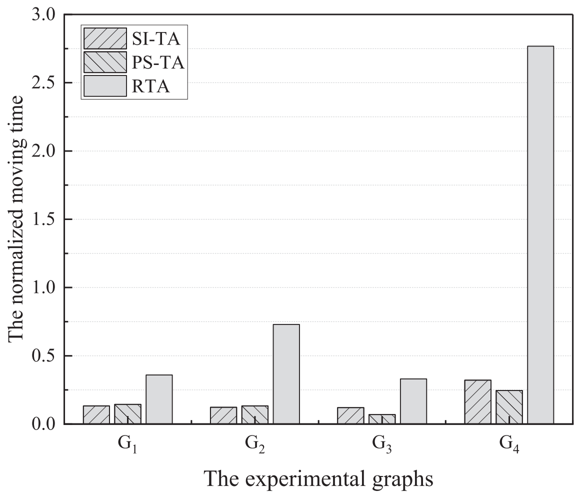


FIGURE 3. The moving time of space robots.

state of nodes in z. We calculate the T utilizing formula (8) when $h > max(S_i)$. GRTA can be formally described by Algorithm 5.

C. COMBINATION ALGORITHM

We combine Algorithm 2 and Algorithm 5 to obtain the SI-TA algorithm. Besides, we also obtain a combination algorithm named PS-TA by combining Algorithm 4 and Algorithm 5. The proposed algorithms SI-TA and PS-TA solve the path planning problem through Algorithm 2 and Algorithm 4, respectively. After that, SI-TA and PS-TA obtain the execution decision of the exploration task by using the GRTA algorithm.

V. SIMULATION RESULTS AND ANALYSIS

A series of experiments are implemented in this section to demonstrate the performance of SI-TA and PS-TA. Furthermore, the algorithm RTA is customized for comparison. The RTA selects nodes randomly for each space robot and uses GRTA to allocate tasks.

A. SIMULATION SETUP

As shown in FIGURE 1, we conduct a series of experiments on four areas with different graph structures to verify the performance of our proposed algorithms. The space robots are homogeneous, i.e. they have the same configuration. It takes the same time for different robots to explore the same area and the same time to move the same distance. The structures of graph G_1 [41], G_2, and G_3 are relatively simple, while the structure of G_4 [42] is relatively complex. Besides, the area corresponding to the node with the largest serial number is selected as the destination area. The time consumed by space robots moving between two areas is uniformly distributed in [10] and [20]s. The time consumed by a space robot exploring an area is distributed in [200, 500]s.

B. EXPERIMENTAL RESULTS AND PERFORMANCE ANALYSIS

FIGURE 2 shows the completion latency of exploration task generated by algorithms on different experimental graphs. The default number of space robots is set to 3. Initially, the space robots locate in the areas corresponding to nodes number 0,1 and 2. In order to facilitate observation, we normalize the latency. From FIGURE 2, it is clearly observed that the completion latency generated by proposed algorithm PS-TA is always minimum. The main reason is that PS-TA can balance the load among space robots. We also observe that the task completion latency generated by proposed SI-TA is always lower than that of RTA on G_1, G_3 and G_4. However, the task completion latency generated by baseline algorithm RTA is less than that of SI-TA on G_3. The main reason is that the high randomness of RS-TA in path planning may generate relatively short task completion latency in some cases.

FIGURE 3 shows the moving time of space robots by utilizing different algorithms. Because the fuel of space robot is limited and refueling for space robots is difficult, it is significant to consider the energy consumption of space robots. Due to the homogeneity of space robots, the exploration time spent by each space robot to explore the same area is the same. The differences in the completion latency generated by different algorithms are dominated by the moving time of space robots. Therefore, we can compare the performances of algorithms in terms of energy efficient according to the moving time of space robots. For the sake of simplicity, we normalize the moving time. From FIGURE 3, it is observed that the moving time of space robots by utilizing RTA is the largest in most cases. Due to the high randomness, RTA works better than SI-TA in terms of the performance of moving time on G_1 in some cases. Besides, when the experiments are carried out on G_1 and G_2, SI-TA works better than PS-TA in terms of the performance of moving time. However, on G_3 and G_4, PS-TA works better than SI-TA. The main reason is that SSSI in

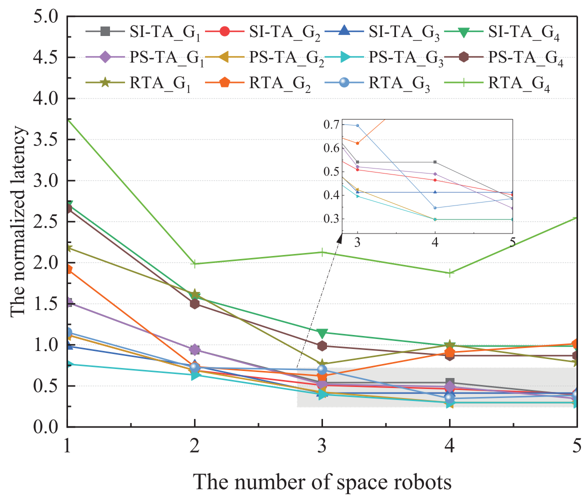


FIGURE 4. The latency vs. the number of space robots for different algorithms.

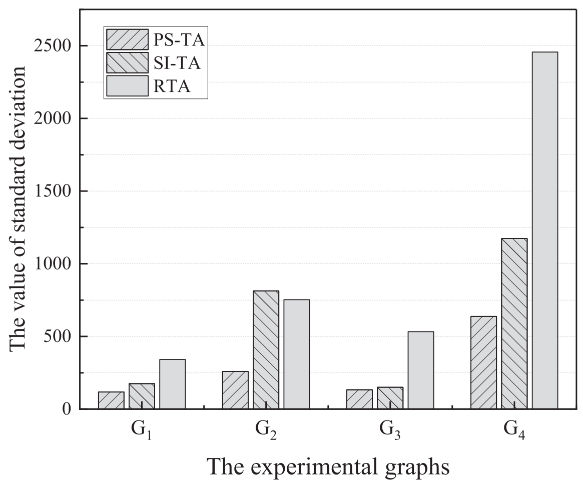


FIGURE 6. The value of standard deviation on different graphs.

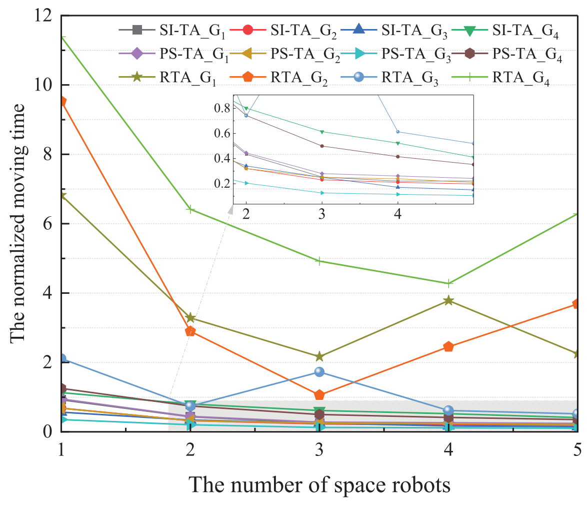


FIGURE 5. The moving time vs. the number of space robots.

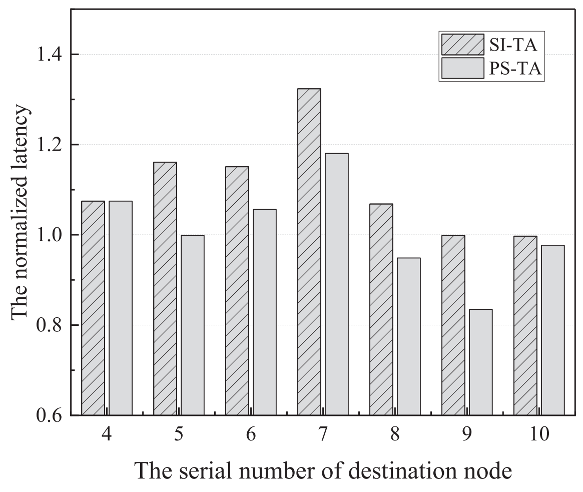


FIGURE 7. The latency vs. serial numbers of destination nodes.

SI-TA may generate more repeated paths among space robots when the initial position of space robots is relatively close.

FIGURE 4 shows the variation of task completion latency generated by different algorithms as the number of space robots increases. The number of space robots is set from 1 to 5. From FIGURE 4, we can clearly observe that the task completion latencies generated by SI-TA and PS-TA decrease as the number of space robots increases. The main reason is that as the number of space robots participating in the exploration task, the burden a single space robot undertakes is reduced. Finally, the overall task completion latency is reduced. Furthermore, the completion latency generated by PS-TA is always less than that generated by SI-TA as the number of space robots increases. Besides, due to the high randomness of RTA, the performance of RTA in terms of completion latency fluctuates greatly.

FIGURE 5 shows the moving time of space robots by utilizing different algorithms with the increasing number of space robots. From FIGURE 5, it is obviously observed that the moving time of space robots by utilizing SI-TA and PS-TA decreases with the increase in the number of space robots. This is because the number of areas a space robot explores decreases as the number of space robots increases, the length of the path of each space robot is reduced. Furthermore, when experiments are carried out in G_1 and G_2, the moving time of space robots by utilizing SI-TA is always smaller than that by utilizing PS-TA as the number of space robots increases. The performances of SI-TA and PS-TA in terms of moving time are reversed when the experiments are performed on G_3 and G_4. Besides, the performance of RTA in terms of moving time fluctuates greatly due to high randomness.

We evaluate the performance in terms of load balancing for algorithms on different graphs by calculating the standard deviation of task completion latency. The default number of space robots is set to 3. The node with the maximum serial number in a graph is set as the destination node. The results

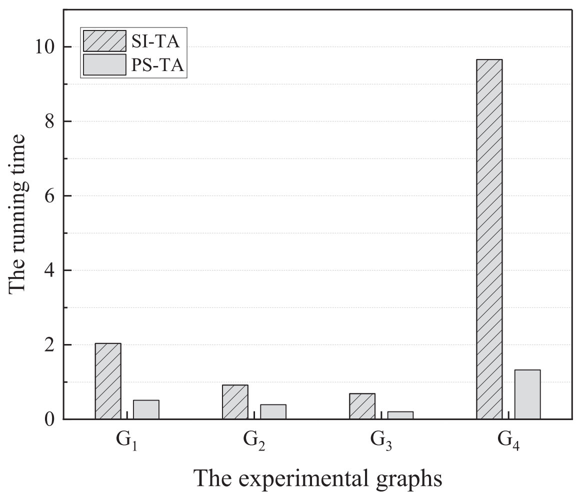


FIGURE 8. The running time of SI-TA and PS-TA on different graphs.

are shown in FIGURE 6. PS-TA consistently outperforms SI-TA and RTA in terms of load balancing. The performance of RTA is better than that of SI-TA when the experimental graph is G_2. The primary factor is that the reduced number of neighbors for each node in G_2 mitigates the impact of the high level of randomness of RTA.

FIGURE 7 shows the variation in completion latency of tasks as the position of the destination node changes. We set the start nodes to 0, 1, and 2, respectively. The serial numbers of destination node are set from 4 to 10. We carry out experiments on G_1 with proposed algorithms SI-TA and PS-TA. FIGURE 7 shows that PS-TA always works better than SI-TA in terms of latency. Last, FIGURE 8 shows the running time of proposed algorithms PS-TA and SI-TA on different graphs. From FIGURE 8, we can see that the running time of SI-TA is always larger than that of PS-TA on different graphs. We can also conclude that the running time of algorithms will increase as the number of nodes in a graph increases.

VI. CONCLUSION

In this paper, we have investigated the joint path planning and task allocation problem for space robots and formalized the problem to minimize the completion latency of exploration task. In this problem, multiple space robots set out simultaneously from different starting areas to perform exploration tasks and assemble at a single destination after exploring all areas. We have proposed algorithms SI-TA and PS-TA by integrating a task allocation algorithm and a path planning algorithm to solve the completion latency minimization problem. In particular, we have proposed two algorithms for path planning named SSSI and FPS, which can generate paths for all space robots moving from their respective starting nodes to the destination node. In addition, each node in the given graph must be contained by at least one path. Furthermore, we have proposed GRTA as the task allocation algorithm based on the generated paths with the greedy idea to obtain minimum completion latency. To further prove the performance of the proposed algorithms, the RTA algorithm has been customized as a baseline, which generates paths for space robots by randomly selecting nodes. Then, RTA allocates tasks with GRTA. We have implemented extensive experiments to demonstrate the performance of the proposed algorithms. Experimental results demonstrate the effectiveness of the proposed algorithm. Compared with RTA, the proposed algorithms SI-TA and PS-TA reduce completion latency at least by 20% and 40%, respectively. Regarding energy efficiency, SI-TA and PS-TA outperform RTA in the most cases. Furthermore, SI-TA and PS-TA can obtain more stable performance than RTA in the most cases. On the other hand, PS-TA runs faster than SI-TA, while SI-TA can generate better paths than PS-TA when the distance between starting nodes is large.

Our future work will consider the heterogeneity of space robots and real-time exploration task, as the heterogeneity produces new challenges for static path planning, while the real-time exploration task leads us to deal with dynamic path planning.

DECLARATION OF INTERESTS

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

REFERENCES

[1] X. Hou, Li Zhang, Y. Su, G. Gao, Y. Liu, Z. Na, and Q. Zhang, "A space crawling robotic bio-paw (SCRBP) enabled by triboelectric sensors for surface identification," *Nano Energy*, vol. 105, pp. 1–12, Jan. 2023.

[2] Z. Xing, Y. Zhao, and S. Zhu, "Path planning method design and dynamic model simplification of free-flying space robot," in *Proc. 15th IEEE Conf. Ind. Electron. Appl. (ICIEA)*, Nov. 2020, pp. 1501–1505.

[3] Y. Lin, "Current status and analysis of space robot," *Spacecr. Eng.*, vol. 24, no. 5, pp. 1–10, 2015.

[4] L. E. X. Youlun and D. Han, *Robotics*. Beijing, China: Mech. Ind. Press, 1993.

[5] A. N. Atiyah, N. Adzhar, and N. I. Jaini, "An overview: On path planning optimization criteria and mobile robot navigation," *J. Phys., Conf. Ser.*, vol. 1988, no. 1, pp. 1–11, 2021.

[6] L. Chen, Y. Huang, H. Zheng, H. Hopman, and R. Negenborn, "Cooperative multi-vessel systems in urban waterway networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3294–3307, Aug. 2020.

[7] P. B. Kumar, C. Sahu, D. Parhi, K. Pandey, and A. Chhotray, "Static and dynamic path planning of humanoids using an advanced regression controller," *Scientia Iranica*, vol. 26, no. 1, pp. 375–393, 2019.

[8] Z. Tang and H. Ma, "An overview of path planning algorithms," *IOP Conf. Ser., Earth Environ. Sci.*, vol. 804, no. 2, pp. 1–11, 2021.

[9] K. Cai, C. Wang, J. Cheng, C. W. De Silva, and M. Q.-H. Meng, "Mobile robot path planning in dynamic environments: A survey," 2020, *arXiv:2006.14195*.

[10] G. Klancar, A. Zdesar, S. Blazic, and I. Skrjanc, *Wheeled Mobile Robotics: From Fundamentals Towards Autonomous Systems*. Oxford, U.K.: Butterworth-Heinemann, 2017.

[11] M. M. Costa and M. F. Silva, "A survey on path planning algorithms for mobile robots," in *Proc. IEEE Int. Conf. Auto. Robot Syst. Competitions (ICARSC)*, Apr. 2019, pp. 1–7.

[12] H. S. Dewang, P. K. Mohanty, and S. Kundu, "A robust path planning for mobile robot using smart particle swarm optimization," *Proc. Comput. Sci.*, vol. 133, pp. 290–297, Jan. 2018.

[13] M. Tipaldi, R. Iervolino, and P. R. Massenio, "Reinforcement learning in spacecraft control applications: Advances, prospects, and challenges," *Annu. Rev. Control*, vol. 54, pp. 1–23, Jan. 2022.

[14] B. Basbous, "2D UAV path planning with radar threatening areas using simulated annealing algorithm for event detection," in *Proc. Int. Conf. Artif. Intell. Data Process. (IDAP)*, Sep. 2018, pp. 1–7.

[15] A. Bakdi, A. Hentout, and H. Boutami, "Optimal path planning and execution for mobile robots using genetic algorithm and adaptive fuzzy-logic control," *Robot. Auto. Syst.*, vol. 89, no. 1, pp. 95–109, 2016.

[16] N. Chayaamor-Heil, "From bioinspiration to biomimicry in architecture: Opportunities and challenges," *Encyclopedia*, vol. 3, no. 1, pp. 202–223, Feb. 2023.

[17] A. Seddaoui and C. M. Saaj, "Collision-free optimal trajectory for a controlled floating space robot," in *Towards Autonomous Robotic Systems*. Cham, Switzerland: Springer, 2019, pp. 248–260.

[18] P. Yan, H. Xiao, J. Guo, C. Bai, and H. Zheng, "Adaptive cooperative detection method for unmanned planetary vehicles based on deep reinforcement learning," in *Proc. IEEE Int. Conf. Unmanned Syst. (ICUS)*, Oct. 2019, pp. 714–719.

[19] M. Pflueger, A. Agha, and G. S. Sukhatme, "Rover-IRL: Inverse reinforcement learning with soft value iteration networks for planetary rover path planning," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 1387–1394, Apr. 2019.

[20] J. Wang, X. Shang, T. Guo, J. Zhou, S. Jia, and C. Wang, "Optimal path planning based on hybrid genetic-cuckoo search algorithm," in *Proc. 6th Int. Conf. Syst. Informat. (ICSAI)*, Nov. 2019, pp. 165–169.

[21] Y. E. M. Vieira, R. A. de Mello Bandeira, and O. S. da Silva Júnior, "Multi-depot vehicle routing problem for large scale disaster relief in drought scenarios: The case of the Brazilian northeast region," *Int. J. Disaster Risk Reduction*, vol. 58, pp. 1–12, Jan. 2021.

[22] Y. Sun, Z. Zhang, Y. Li, and J. Wu, "Joint optimization of path planning and task assignment for space robot," in *Proc. 12th Int. Symp. Parallel Archit., Algorithms Program. (PAAP)*, Dec. 2021, pp. 47–51.

[23] L. Wu, X. Huang, J. Cui, C. Liu, and W. Xiao, "Modified adaptive ant colony optimization algorithm and its application for solving path planning of mobile robot," *Expert Syst. Appl.*, vol. 215, pp. 1–22, Jan. 2023.

[24] B. Chen and G. Quan, "NP-hard problems of learning from examples," in *Proc. 5th Int. Conf. Fuzzy Syst. Knowl. Discovery*, Oct. 2008, pp. 182–186.

[25] K. Karur, N. Sharma, C. Dharmatti, and J. E. Siegel, "A survey of path planning algorithms for mobile robots," *Vehicles*, vol. 3, no. 3, pp. 448–468, 2021.

[26] Y. Zhuang, Y. Sun, and W. Wang, "Mobile robot hybrid path planning in an obstacle-cluttered environment based on steering control and improved distance propagating," *Int. J. Innov. Comput. Inf. Control*, vol. 8, pp. 4095–4109, Jan. 2012.

[27] M. A. Contreras-Cruz, V. Ayala-Ramirez, and U. H. Hernandez-Belmonte, "Mobile robot path planning using artificial bee colony and evolutionary programming," *Appl. Soft Comput.*, vol. 30, pp. 319–328, May 2015.

[28] S. K. Debnath, R. Omar, and N. B. A. Latip, "A review on energy efficient path planning algorithms for unmanned air vehicles," *Comput. Sci. Technol.*, vol. 10, pp. 523–532, Jan. 2019.

[29] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2006.

[30] F. Yan, E. Xia, Z. Li, and Z. Zhou, "Sampling-based path planning for high-quality aerial 3D reconstruction of urban scenes," *Remote Sens.*, vol. 13, no. 5, pp. 1–23, 2021.

[31] U. Orozco-Rosas, K. Picos, and O. Montiel, "Hybrid path planning algorithm based on membrane pseudo-bacterial potential field for autonomous mobile robots," *IEEE Access*, vol. 7, pp. 156787–156803, 2019.

[32] U. Orozco-Rosas, O. Montiel, and R. Sepúlveda, "Mobile robot path planning using membrane evolutionary artificial potential field," *Appl. Soft Comput. J.*, vol. 77, pp. 236–251, Apr. 2019.

[33] S. Lin, A. Liu, J. Wang, and X. Kong, "An intelligence-based hybrid PSO-SA for mobile robot path planning in warehouse," *J. Comput. Sci.*, vol. 67, pp. 1–11, Jan. 2023.

[34] K. Shi, Z. Wu, B. Jiang, and H. R. Karimi, "Dynamic path planning of mobile robot based on improved simulated annealing algorithm," *J. Franklin Inst.*, vol. 360, no. 6, pp. 4378–4398, Apr. 2023.

[35] U. Orozco-Rosas, K. Picos, J. J. Pantrigo, A. S. Montemayor, and A. Cuesta-Infante, "Mobile robot path planning using a QAPF learning algorithm for known and unknown environments," *IEEE Access*, vol. 10, pp. 84648–84663, 2022.

[36] F. Jin, "Path planning of free-flying space robot using memetic algorithm," in *Proc. IFOST*, vol. 2, Jun. 2013, pp. 19–22.

[37] F. Jin and G. Shu, "Path planning of free-flying space robot based on artificial bee colony algorithm," in *Proc. 2nd Int. Conf. Comput. Sci. Netw. Technol.*, Dec. 2012, pp. 505–508.

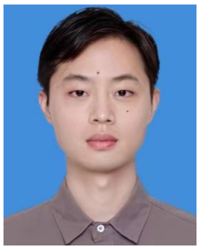
[38] W. Ye, D. Ma, and H. Fan, "Path planning for space robot based on the self-adaptive ant colony algorithm," in *Proc. 1st Int. Symp. Syst. Control Aerosp. Astronaut.*, 2006, pp. 30–33.

[39] B. Shi and H. Wu, "Space robot motion path planning based on fuzzy control algorithm," *J. Intell. Fuzzy Syst.*, vol. 2021, pp. 1–8, May 2021.

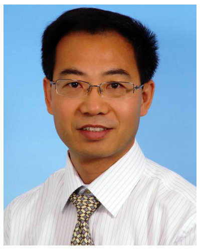
[40] X. Hu, X. Huang, T. Hu, Z. Shi, and J. Hui, "MRDDPG algorithms for path planning of free-floating space robot," in *Proc. IEEE 9th Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Nov. 2018, pp. 1079–1082.

[41] *Mastermind—2021 National Algorithm Design Challenge*. Accessed: Jul. 20, 2022. [Online]. Available: https://www.shenjims.com/

[42] Y. He, X. Zhang, Z. Xia, Y. Liu, K. Sood, and S. Yu, "Joint optimization of service chain graph design and mapping in NFV-enabled networks," *Comput. Netw.*, vol. 202, Jan. 2022, Art. no. 108626.



YIFEI SUN is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Guangdong University of Technology, Guangzhou, China. His current research interests include edge computing and resource allocation.



JIGANG WU (Member, IEEE) received the B.Sc. degree from Lanzhou University and the Ph.D. degree from the University of Science and Technology of China. He was with Nanyang Technological University, Singapore, from 2000 to 2010. He was a Tianjin Distinguished Professor with Tianjin Polytechnic University, China, from 2010 to 2015. He is currently a Distinguished Professor with the School of Computer Science and Technology, Guangdong University of Technology. He has published more than 300 papers. His research interests include network computing and machine learning.

TONGLAI LIU was born in 1982. He received the B.E. and M.E. degrees from the Guilin University of Electronic Technology, China, in 2007 and 2010, respectively, and the Ph.D. degree from the Guangdong University of Technology, in 2021. After pursued the M.E. degree, he joined the Guilin University of Electronic Technology. He is currently an Associate Professor with the Zhongkai University of Agriculture and Engineering. His current research interests include data mining, blockchain technology, edge computing, and database.

• • •