Task Allocation of Multi-robot Cooperative Coverage Based on Circular Coding Genetic Algorithm

Bin Xin[*1], Heng Wang[*2], Ming Li[*3]

[*1]School of Automation, Beijing Institute of Technology
Key Laboratory of Intelligent Control and Decision of Complex Systems
5 South Zhongguancun Street, Haidian District, Beijing 100081, China
E-mail: brucebin@bit.edu.cn
[*2]School of Automation, Beijing Institute of Technology
5 South Zhongguancun Street, Haidian District, Beijing 100081, China
[*3]BIT Navigation and Control Technology Co. Ltd., Beijing 102206, China

Abstract. Aiming at the problem of multi-robot cooperative area coverage, this paper adopts genetic algorithm based on circular coding to complete the allocation of robot coverage area. In this paper, the given map is divided into rectangular regions and merged into several areas to be allocated. The genetic algorithm with circular coding method is used to complete the coverage task allocation of these areas. Combined with the result of task allocation, this paper designs a navigation point set for each robot. Robots will cover the whole target area by passing through these navigation points in sequence. This paper compares the state-of-the-art algorithm and demonstrates that the algorithm used in this paper has advantages in the balance of task allocation. By running the task allocation algorithm in different situations, it is verified that the task allocation algorithm has situation scalability. This paper also carries out simulation verification on the Gazebo simulation platform which is under the Robot Operating System(ROS).

Keywords: Task allocation; Genetic algorithm; Area coverage; Multi-robot

1. INTRODUCTION

At present, only using a single robot to complete tasks can not meet the increasingly complex application environment[1]. Comparing with single robot, multi-robot system shows great advantages in task applicability, economy, robustness and scalability[2]. The research on multi-robot system has gradually become a hotspot of robot research, and its research results have also been widely used in logistics[3], industry[4], agriculture[5] and other fields.

The research of multi-robot system includes the path planning[6], task allocation[7], formation tracking[8] and so on. The problem studied in this paper is the task allocation of multiple robots. This problem requires multiple robots can fully cover all parts of the area with as high efficiency as possible. In recent years, many scholars have devoted themselves to the research of this problem. Gabriely and Rimon proposed spiral spanning tree algorithm[9]. This algorithm divides the map into grids to get a graph structure and then plans a coverage path for each robot. Kapoutsis et al. designed the DARP algorithm[10], which divides the area evenly and designs the coverage path for each robot. Kapanoglu et al. gave a solution to this problem by using a pattern-based genetic algorithm[11]. Elango et al. focused on the balance of task allocation and the distance traveled by the robot, and used the auction algorithm to complete the coverage area allocation of the robot[12].

In this paper, we realize a map divider which can divide the map into many areas. We design a task allocation method based on genetic algorithm. This method can allocate areas to robots. We design a navigation planner. Combined with the area allocation information, the navigation planner designs navigation points set for robots. Robots can cover the whole map by passing through the navigation points. Comparing with the A-STC algorithm proposed by Gao et al.[13], it is proved that the task allocation algorithm used in this paper has better performance in the efficiency and balance of task allocation. In addition, we also verify the working process of the whole design on the Gazebo simulation platform.

2. DESIGN OF WORKING PROCESS

To solve the problem, the overall working process of collaborate area coverage designed in this paper is shown in Figure 1.

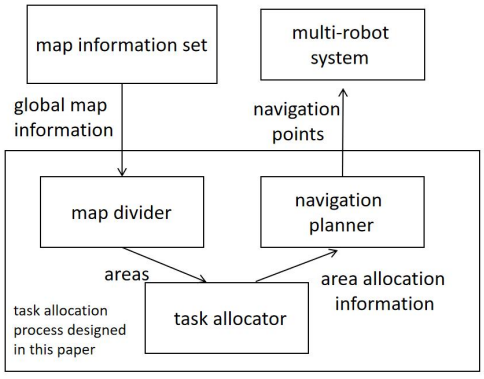


Fig. 1. Overall working process

The method designed in this paper needs global offline map information. The map divider divides the whole map into several areas according to the map information. The

task allocator allocates the divided areas according to the map area information and then navigation planner can obtain the task assignment information. After the task assignment information is processed by the navigation planner, a series of navigation points can be obtained by robots. Robots only need to navigate to these navigation points and then they can cover the whole map.

3. MAP DIVIDER

The global offline map used in this paper is occupancy grid map. The function of map divider is to divide the occupancy grid map. The occupancy grid map consists of many small squares called grids. There are two states of these grids: “occupied” and “free”. Robots cannot pass through the grid in the occupied state. The occupancy grid map used in this paper is as follows:

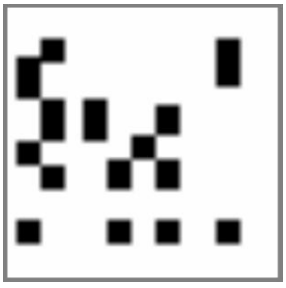


Fig. 2. Experimental map

In the above map, the black part is the obstacle and the white part is the area where the robot can pass. The map divider uses the following strategies to divide the entire occupancy grid map:

Make extension lines along each edge of the obstacle so that several division lines can be obtained. These division lines, together with the boundary of the map, divide the whole map into several small zones. These small zones can be divided into two types. One is occupied and the other is free. Traverse all zones, eliminate zones occupied by obstacles, and only keep free zones. Merge the reserved small zones according to the following rules: firstly, merge vertically. Merge all zones with the same horizontal width and directly connected to obtain the vertically merged partition set. Then, conduct horizontal merge. Merge all vertically merged partitions with the same vertical width and directly connected to obtain the divided area set.

The divided occupancy grid map is as follows:

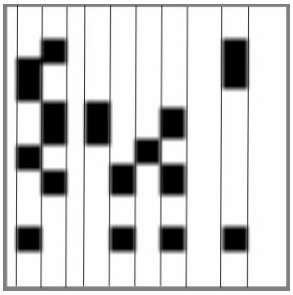


Fig. 3. Area division result

4. TASK ALLOCATION

The function of task allocator is to allocate the divided areas to robots. The allocation algorithm used by task allocator is genetic algorithm. Genetic algorithm was proposed by Professor Holland in the 1970s[14], which draws lessons from Darwin's theory of natural selection. The main idea of genetic algorithm is that in the whole population, individuals with higher fitness have a greater chance of being retained, while individuals with lower fitness have a smaller probability of being retained. The algorithm needs to select each generation of population, and carry out crossover operation and mutation operation on the retained individuals. Crossover operation and mutation operation can produce new individuals, new individuals may have better adaptability than their parents, which makes the whole population evolve. The specific steps of genetic algorithm are as follows:

1.Population initialization. A set of solutions are given under certain constraints. At the same time, a series of parameters required by genetic algorithm are given: population size, length of each chromosome, individual crossover rate, individual mutation rate, maximum iteration times, etc;

2.Calculate the fitness of all individuals. The fitness of each individual in the population is calculated according to the designed fitness function;

3.Select individuals in the population. Individuals are selected according to the fitness value calculated in the previous step. Individuals with higher fitness value have a greater chance to be retained;

4.Crossover the retained individuals. According to the crossover rate, the individuals in the population are crossed to obtain new individuals;

5.Mutate the individuals. The individuals in the population are mutated according to the mutation rate to obtain new individuals;

6.Judge whether the termination conditions of the algorithm are met. If the termination conditions of the algorithm are met, the algorithm ends. If the termination conditions are not met, jump back to step 2 and run steps 2-5 until the end of the whole algorithm.

The optimization problem model solved by genetic algorithm in this paper is as follows:

$$\min Z = \max(C) \tag{1}$$

$$\text{s.t. } \sum_{i=1}^{n} X_{i,j} = 1, j = 1, \cdots, m \tag{2}$$

$$X_{i,j} = 0 \text{ or } 1 \tag{3}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{m} X_{i,j} = m \tag{4}$$

In the above formula, Z is the variable to be optimized. C is the set of travel distances of all robots. The number of robots is n and the number of tasks is m. $X_{i,j}$ can be 0 or 1. When $X_{i,j}$ is 1, task j will be assigned to robot i. Formula 2 means that each task can only be assigned to one robot. Formula 4 means that all tasks should be fully assigned.

The following will introduce the design method of genetic algorithm:

4.1. Chromosome coding mode

In this paper, circular coding method is used for chromosome coding[15]. Suppose there are *m* robots and *n* areas to be allocated. The length of the chromosome is *n*+*m*. A chromosome is represented by the random arrangement of integers from 1 to *n*+*m* where 1 to *m* represent *m* robots and *m*+1 to *n*+*m* represent *n* areas to be allocated. In order to indicate that the area is assigned to the robot, we make the following rules: the tasks between two adjacent robots in the chromosome are assigned to the robot on the right and the robot on the right performs the tasks in order from right to left; The tasks assigned to the leftmost robot in the chromosome include two parts. One is all the tasks on its left side and the other is all the tasks on the right side of the rightmost robot in the chromosome. The order in which the robot performs tasks is from the tasks on its left side and from right to left. Here is a specific example to show the meaning of chromosome coding.

Suppose a chromosome code is 924763815, 1 to 3 represent robots, 4 to 9 represent areas. Then according to the above coding principles, it can be obtained that for robot 1, it needs to perform task 8, for robot 2, it needs to perform two tasks 9 and 5 in turn, and for robot 3, it needs to perform three tasks 6, 7 and 4 in turn.

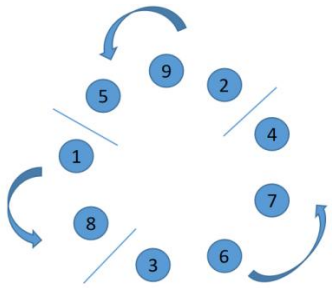


Fig. 4. Chromosome decoding

4.2. Design of fitness function

In order to reduce the overall working time of the robot system, the fitness function is designed as follows:

$$\text{fitness}(i) = \frac{1}{\max(\text{dis}(1,\cdots,n))} \tag{5}$$

In the above formula, $\text{fitness}(i)$ represents the fitness value of chromosome *i,* $\max(\text{dis}(1,\cdots,n))$ represents the distance of the robot with the longest distance in robot 1, 2,$\cdots$,*n* on the premise of adopting chromosome *i*. The calculation of the distance to be taken by each robot involves the design of navigation planner, which will be described in section 5.

4.3. Design of selection operation

The selection operation adopts the elite retention mechanism to make roulette selection for the population. Suppose there are *N* individuals in the whole population, which is recorded as $X_1, X_2, \cdots, X_N$. The fitness function of the individual i is $fitness(i)$. When selecting individuals, retain the individual with the highest fitness among all individuals in the population. And then select the remaining individuals. The probability that an individual is selected to enter the next generation of population is:

$$\text{p}(i) = \frac{\text{fitness}(i)}{\sum_{j=1}^{N}\text{fitness}(j)} \tag{6}$$

From the above formula, it can be seen that individuals with greater fitness have a greater chance of being retained.

4.4. Design of crossover operation

Combined with reference[15], this paper adopts a heuristic crossover method to complete the crossover operation.

Firstly, the distance matrix used in crossover operation is described. Suppose there are *m* robots and *n* areas to be allocated. We use integers 1 to *m* to represent *m* robots and *m*+1 to *m*+*n* to represent the areas to be allocated. We can get a symmetric matrix with the scale of (*m*+*n*)*(*m*+*n*). This matrix is called the distance matrix. The element $s_{i,j}$ in row *i* and column *j* of the matrix represents the distance between area or robot *i* and area or robot *j*. When calculating this distance, the center point of the area is used to represent the area and participates in the calculation of the distance. For the areas in Figure 3, if we use 3 robots, we can get a 29 * 29 distance matrix.

Then it expounds the specific process of crossover operation. Firstly, an integer *z* is randomly found from 1 to *m* + *n*. Then the integer is inserted into the new chromosome *N1*, and the integer is deleted from the chromosomes of parent X and *Y*. Find out integers(X_R and Y_R) that are adjacent to the right of the selected integer in *X* and *Y*. Find element s_{z,Y_R} and element s_{z,X_R} in the distance matrix. If $s_{z,X_R} < s_{z,Y_R}$, X_R will be inserted in *N1* and deleted in *X* and *Y*. Conversely, Y_R will be inserted in *N1* and deleted in *X* and *Y*. Finally, check whether the length of *N1* reaches *m* + *n*. if so, stop inserting integers into new individual and derive the crossed chromosome. If not so, replace the integer *z* with X_R (Y_R) and repeat the whole process until the length of *N1* reaches *m*+*n*. Similarly, the generation method of another chromosome *N2* generated by crossover operation is similar to *N1*, but in the second step, it is changed to the integer adjacent to the left of integer *Z* instead of the integer adjacent to the right.

After practical verification, this method can ensure that the circular coding mode of parents will not be destroyed, and the two children also have the characteristics of circular coding.

4.5. Design of mutation operation

The mutation operation adopts the method of two-point inversion[15]. The specific operation process is to randomly find two different loci from the chromosome code to be mutated. Then reverse the chromosome code between the two loci to obtain the mutated chromosome. The specific demonstration is as follows:

Assuming that the pre-mutation chromosome is 126348759 and the loci of 2 and 7 are randomly selected as the target loci. The post mutation chromosome is 178436259.

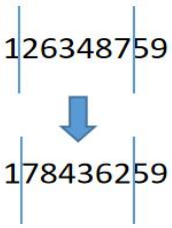


Fig. 5. Mutation

4.6. The whole process of genetic algorithm

The flow of the whole genetic algorithm is as follows:

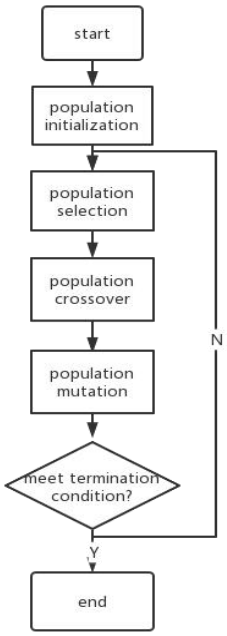


Fig. 6. Algorithm flow

Firstly, initialize the population, and set the parameters such as population size, maximum generation and mutation rate. In the second step, the population is selected by roulette selection using elite retention strategy. The third step is to cross the population using heuristic crossover operation. It should be pointed out that the best individual don't participate in crossover and mutation. The fourth step is to use two-point inversion for population chromosome mutation. Finally, judge whether the maximum number of iterations has been reached. If the maximum number of iterations of the population has been reached, stop the algorithm. On the contrary, repeat steps 2 to 4.

After iterating to the last generation, we use the individual with the highest fitness in the last generation as the final solution of task allocation. We can ensure that the selected individual is the best individual of all generation population because the best performing individuals in each generation do not participate in crossover and mutation.

It should be noted that the task allocation algorithm used in this paper is different from that used in reference paper[15]. The reference paper aims to reduce the total distance of all robots and this paper aims to balance the tasks of multiple robots. Based on this consideration, we modify the fitness function of genetic algorithm. In the iterative process, the algorithm can reduce the distance traveled by the robot with the longest search path. We modify the selection operation of the algorithm. The use condition of mutation operation and crossover operation of genetic algorithm is also modified, so that the optimal individuals in each generation of population can be preserved.

5. DESIGN OF NAVIGATION PLANNER

The navigation planner sets the navigation point set for each robot after the task allocator gives the task of each robot. Each robot needs to navigate according to the assigned navigation point set to complete the coverage of the whole map.

The shape of the area is rectangular. When we know the initial position of the robot, we can get the closest rectangular vertex to the robot. We take this vertex as the first navigation point for the robot to enter the area. After the robot enters the area, by designing the navigation point set, the robot can cover the area as shown in the figure below. We let the robot navigate back and forth along the longer side of the rectangle, which can make the robot completely cover the whole area. By moving in this way, we can clearly know the position of the robot after covering this area, which also prepares the conditions for the next area transfer of the robot.

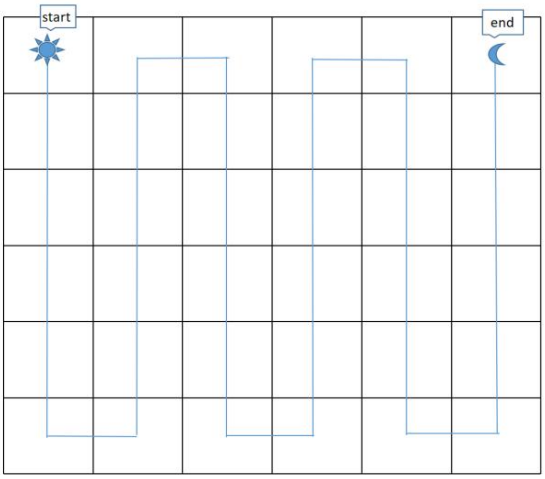


Fig. 7. Path within area

When the robot have completed the coverage of area. It needs to enter the next area in the order assigned to it by the task allocator. After covering the previous area, it is easy to get the current position of the robot. We can complete the area transfer of the robot by taking the vertex closest to the robot in the next area as the next

navigation point. After reaching the next area, the robot continues to design the navigation points according to the navigation point design method described above until the navigation point design of all areas is completed.

Then, the estimation method of the total distance is described. Combined with the planning method of the above navigation points, the distance to be taken by each robot is roughly composed of two parts: one is the distance to be taken by the robot in the area, and the other is the distance to be taken by the robot during area transfer. The distance to be traveled in the area can be easily estimated by calculating the grid number of the area, and the distance between areas can be estimated by calculating the distance between the last navigation point of the previous area and the first navigation point of the next area. The overall distance estimation can be obtained by adding the distance estimation in the area to the distance estimation between areas.

6. ANALYSIS OF EXPERIMENTAL RESULTS

In this paper, Python3.9 is used to compile the above procedure. After obtaining the navigation points information, the simulation verification is carried out by using the Gazebo platform under the ROS system. The results are as follows:

6.1. Efficiency analysis

This part analyzes the efficiency of multi-robot task allocation results. The efficiency of task allocation can be measured by makespan. Makespan refers to the time taken for all robots to complete the area coverage. This index can indicate whether the amount of tasks between different robots are balanced. In this paper, we use the area covered by the robot with the largest coverage area among all robots (in grid as the basic unit) to represent makespan. Obviously, on the premise of the number of robots and the area of environmental feasible area are determined, we can get that the lower bound of the makespan is the area of feasible area divided by the number of robots.

We use the map in Figure 2 to carry out the calculation experiment. In the experiment, the size of the population is 50. The probability of chromosome mutation is 20%. The maximum iterations of the population is 1000. The crossover rate is 40%. After repeated experiments, in the case of using three robots, the average makespan is 718.1(standard deviation: 52.9), and the feasible area of the whole map is 1725. In the most ideal case, the makespan is 575. The algorithm proposed in this paper is about 24.9% higher than the ideal case. Combined with the experimental results of Gao et al.[13], the task allocation of A-STC algorithm to the outdoor environment is about 30% higher than the ideal case, It shows that the algorithm proposed in this paper is more efficient.

6.2. Scalability analysis

In order to verify that the algorithm can still have good allocation performance in many different situations, we rearranged the experimental situation to obtain a series of new situations, as shown in Figure 8.

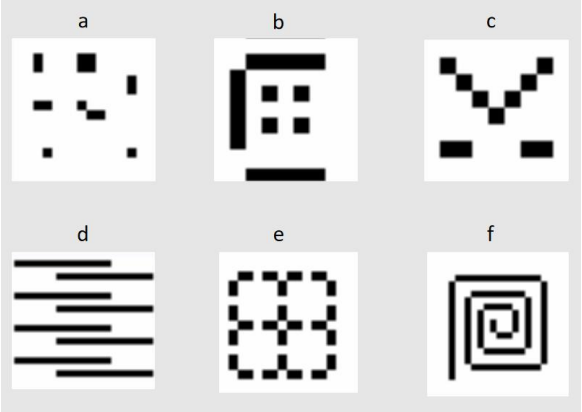


Fig. 8. Rearranged situation map

In different situations, we still use three robots for testing. The probability of chromosome mutation is 15% and other algorithm parameters are the same as those in Section 6.1. The final test results are shown as follows:

Table 1 Experimental test results under different experimental situations

Map	Feasible area	Makespan lower bound	Makespan given by GA	Excess percentage
a	1890	630	752.4(55.5)	19.4%
b	1595	532	666.5(52.7)	25.2%
c	1750	584	655(41)	12.2%
d	1545	515	621.9(40.7)	20.8%
e	1665	555	614.7(31.2)	10.8%
f	1605	535	636.5(44.7)	19%

In the above table, the makespan given by the GA we use is the average makespan after repeated experiments. The data in brackets is the standard deviation of makespan. It shows that the index is basically stable, which reflects the situation adaptability of the algorithm proposed in this paper.

6.3. Gazebo simulation

This paper uses the Gazebo platform in the ROS system to do the simulation and experimental results test. The simulation scene in Gazebo is shown in the figure below. The simulation uses the Gmapping function package provided by ROS to build the map, and obtains the global offline map. The AMCL(Adaptive Monte Carlo Localization) function package is used to realize the global positioning of robots. On the basis of the global positioning of robots, Move_Base function

package is used to complete the navigation of all robots. All of the robots can navigate according to the navigation points obtained by the navigation planner.

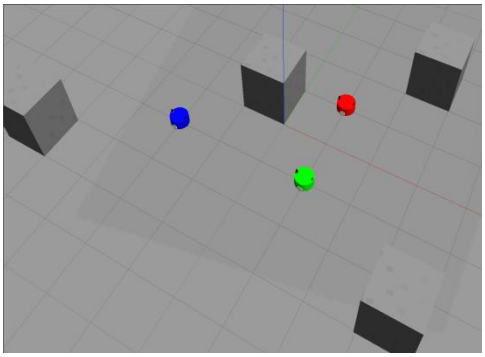


Fig. 9. Simulation scene in Gazebo

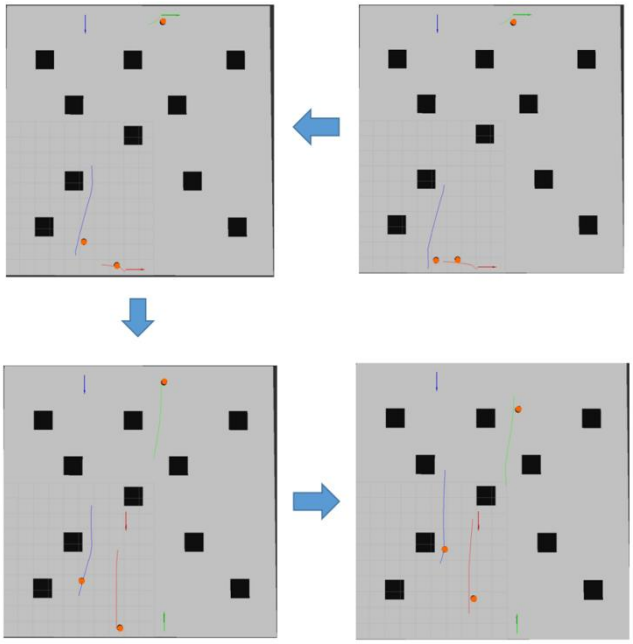


Fig. 10. Robots are navigating to navigation points

In Figure 10, lines are the paths planned by robots, and the arrows represent the navigation target points. The yellow dots are robots. Different colored lines and arrows correspond to different robots.

7. CONCLUSON

In this paper, genetic algorithm is used to realize the task allocation of multiple robots, so that robots can completely cover the whole target area. At the same time, by comparing with the A-STC algorithm proposed by Gao et al, it is verified that the algorithm is more efficient. By changing the experimental situation, it is verified that the algorithm has situation scalability. This paper also does simulation on Gazebo platform, which shows the multi-robot coverage process.

Acknowledgements

This work was supported in part by the National Outstanding Youth Talents Support Program under Grant 61822304 and in part by the Basic Science Center Programs of NSFC under Grant 62088101.

REFERENCES:

[1] Danfeng Wu,GuangPing Zeng,Lingguo Meng,Weijian Zhou,Linmin Li,“Gini Coefficient-based Task Allocation for Multi-robot Systems With Limited Energy Resources”, IEEE/CAA Journal of Automatica Sinica, 5(01), pp.155-168, 2018.

[2] Jun Wu,Xin Xu,Chuanqiang Lian,Hangen He, “A survey of recent advances in cooperative multi-robot systems”, CAAI Transactions on Intelligent Systems, 6(01), pp. 13-27, 2001.

[3] Haikang Robot Technology Co., Ltd, “Intelligent mobile robot system helps the intelligent upgrading of factory logistics”, Automation Panorama, 2020(07), pp. 36-40, 2020.

[4] Zhipeng Wu,Anan Zhao,Yi Zheng,Wei Tian,Haihua Cui, “Research Progress on motion planning of multi robot manufacturing system for complex large part assembly”, Mechanical Science and Technology for Aerospace Engineering, 40(06), pp. 969-978, 2021.

[5] Wei Wang,Yanfei Zhang,Jinliang Gong, “Research on full area coverage of agricultural robot in complex environment based on ant colony-BFS algorithm”, Journal of South China Agriculture University, 42(04), pp. 119-125, 2021.

[6] Yunfeng Fan,Fang Deng,Xiang Shi, “Multi-robot Task Allocation and Path Planning System Design”, Proc. of the 39th Chinese Control Conference, pp.178-183, 2020.

[7] Yulin Mei,Sheng Li,Chen Chen,Andong Han, “A Multi-robot Task Allocation and Path Planning Method for Warehouse System”, Proc. of the 40th Chinese Control Conference, pp.151-156, 2021.

[8] Lu Yue,Airong Wei,Rui Mu, “Formation tracking for multi-agent systems based on dynamic event-triggered”, Proc. of the 40th Chinese Control Conference, pp.136-141, 2021.

[9] Gabriely, Rimon, “Spanning-tree based coverage of continuous areas by a mobile robot”, Proc. Of the 2001 IEEE International Conference on Robotics and Automation, pp. 1927-1933, 2001.

[10] Kapoutsis, Chatzichristofis, Kosmatopoulos, “DARP: Divide Areas Algorithm for Optimal Multi-Robot Coverage Path Planning”, Journal of Intelligent and Robotic Systems, 86(3/4), pp. 663-680, 2017.

[11] Kapanoglu,Alikalfa,Ozkan,al, “A pattern-based genetic algorithm for multi-robot coverage path planning minimizing completion time”, Journal of Intelligent Manufacturing, 23(04), pp.1035-1045, 2012.

[12] Murugappan Elango,Subramanian Nachiappan,Manoj Kumar Tiwari, “Balancing task allocation in multi-robot systems using K-means clustering and auction based mechanisms”, Expert Systems With Applications, 38(06), pp. 6486-6491, 2011.

[13] Guanqiang Gao,Bin Xin, “A-STC: auction-based spanning tree coverage algorithm for motion planning of cooperative robots”, Frontiers of Information Technology & Electronic Engineering, 20(01), pp. 18-31, 2019.

[14] Shuaiyong Zhang, “Research on Task Scheduling Algorithm for High Performance Computing Based on Genetic-Ant Colony”, Master,Zhengzhou University, Henan, 2021.

[15] Wenhui Hu, “Research on Multi-robot Path Planning and Task Assignment in Obstacle Environment”, Master,Hunan University, Hunan, 2018.