Proceedings of 2016 IEEE 13th International Conference on Networking, Sensing, and Control
Mexico City, Mexico, April 28-30, 2016

Path Planning for Multi-robot Systems Using PSO and Critical Path Schedule Method

Songyang Han[1,3], Xianzhong Zhou*[1,2], Chunlin Chen[1,2]
1. Department of Control and Systems Engineering, School of Management and Engineering
2. Research Center for Novel Technology of Intelligent Equipment
Nanjing University, Nanjing, Jiangsu, China, 210093
3. University of Michigan-Shanghai Jiao Tong University Joint Institute
Shanghai Jiao Tong University, Shanghai, China, 200240
Email: zhouxz@nju.edu.cn

***Abstract*—In recent years, there are more and more situations that need robots to do dangerous and complex works instead of human, to improve the efficiency and reduce the risk. Path planning for multi-robot systems is one of the most important problems to solve in practice. As the improvements in communication and control technology of robots, an algorithm is proposed using Particle Swarm Optimization and Critical Path Schedule Method to do path planning for multi-robot systems to reach the targets without collision not only between the robots but also between the robots and environment. Different from traditional method, a schedule is proposed on the planned optimal or feasible path to enable the robots to avoid collisions and minimize the cost such as time, distance. Moreover, a simulation experiment is given to verify the validity of proposed algorithm, whereby the environment is studied in a two dimensional free space.**

Keywords—Multi-robot, Path planning, PSO, CPM

I. INTRODUCTION

Path planning for multi-robot systems is a key problem for mobile robots. Most path planning algorithms can be evaluated in terms of completeness, complexity and optimality. Path planning algorithms for multi-robot systems can be divided into two categories: coupled and decoupled.

Coupled algorithms plan the trajectories of all robots in the environment concurrently. By transferring the state of the individual robot together into a system state representation, a sequence of state transitions can be found to show how all robots reach their goals respectively. Coupled algorithms are built on a centralized architecture, where the information of all states is available. The limitation is that these algorithms need to search in a large configuration space. Using complete search methods, such as A* [1], coupled algorithms have completeness and optimality. However, since the size of the configuration space grows exponentially with the number of robots, the computational complexity of A* search also increases exponentially and quickly becomes intractable. Hopcroft et al. have shown that the general motion planning problem for multiple moving objects is PSPACE-hard [2]. One approach to reduce the size of the search space is to create probabilistic roadmaps (PRMs) through the environment; this method was shown in [3] to be probabilistically complete and demonstrated in simulation for up to five robots. Another approach is to decompose a large map into subgraphs and plan paths between subgraph segments before coordinating motion within each subgraph [4].

Decoupled algorithms plan the motion for individual robots, rather than for all robots simultaneously. In [5], the algorithm first plans a path among the stationary obstacles and then tunes the velocity along the path to avoid collisions with the moving obstacles. Such a decoupled approach has been further revisited; the prioritized planning scheme proposed in [6] assigns priorities to each robot and sequentially computes paths in a time-varying configuration space. In [7], prioritization is combined with potential fields to resolve possible conflicts. Decoupled methods may use a decentralized architecture, allowing independent planning-based methods such as maze-searching [8] or potential fields [7]. They may also use a centralized architecture planning for all robots with a single processor. Centralized decoupled planners typically determine individual trajectories sequentially and combine the plans of all robots to avoid collisions. Plans can be combined by iteratively adding new obstacles into the configuration space [9]; however, this inherently involves assigning priorities to robots to determine the order, which affects the quality of the resulting plan. This can be addressed by considering all different combinations of priorities (e.g., [10] demonstrate up to three robots), or running an optimization process on the priority assignment [11]. In a more dynamic paradigm, the plans of individual robots can be merged into the global coordination plan as new goals are assigned [12]. By planning the motion of robots sequentially, decoupled methods have lower complexity and greater scalability than a coupled plan; however, this comes with a reduction in completeness and optimality.

In this paper, a new algorithm is proposed to combine the advantage of coupled and decoupled approaches. Firstly, decoupled planning based on Particle Swarm Optimization (PSO) is utilized to generate an optimal path for individual robot in a free space rather than grid space, while the shapes of the obstacles can be chosen randomly. After that the schedules

978-1-4673-9975-3/16/$31.00 ©2016 IEEE

for every robot are obtained by Critical Path Schedule Method (CPSM) proposed in this article to avoid collisions without any priorities assigned in advance. Since the target is to minimize the total distance and total time, the priority for time is needed rather than the priority for some specified robot. Another advantage of CPSM is that it doesn't require all robots have the same performance when moving, which means they can have different maximum speed, acceleration and so on. The proposed method has a good performance in completeness, complexity and optimality.

In Section Ⅱ, a more specific problem formulation is given. In Section Ⅲ, the algorithm of path planning for multi-robot systems is described which includes two main step. An example is given in Section Ⅳ to show how to use this algorithm to do path planning for multi-robot systems. Finally, we conclude and discuss future works in Section Ⅴ.

II. Problem Formulation

The problem we need to solve is the path planning for multi-robot systems from their start points to end points in free space. In this process, we need ensure that no collision happens not only between the robots but also between the robots and the obstacles whose positions and shapes are known in the environment. Meanwhile, the cost of distance and time should be minimized.

In terms of mathematic expression, Path planning for individual robot is to find a set of points that the robot passes when moving in the environment. $S_i(i = 1,2,\cdots,num)$ in the global coordinate system $O-XY$ represents the start points of num robots, and $G_i(i = 1,2,\cdots,num)$ represents the end points accordingly. As shown in Fig. 1, the black filled objects represent obstacles (the shapes of the obstacles can be chosen randomly). The sizes of the obstacles need to be expanded in advance to avoid the collisions between the robots and environment (the figure shows the shapes of obstacles after expansion) [13]. Taking robot 1 for example, the task is to find a point set

$$P_1 = \{S_1, p_{11}, p_{12}, \cdots, p_{1m}, G_1\} \quad (1)$$

$(p_{11}, p_{12}, \cdots, p_{1m})$ is the point set we want to find in the environment. $p_{ij}(j = 1,2,\cdots,m)$ must not be in the barrier region, and the points on the straight line joining p_{ij} and its adjacent points must not either. This point set can be used as the encoded mode in PSO.

It is required that robot 1 must move to the direction of the target in every instant. Hence, the coordinate system transformation method is used to limit the robot's range of motion, as shown in Fig. 1 [14]. In the global coordinate system $O-XY$, S_1 is the new origin of coordinates, and the straight line joining S_1 and G_1 is the X' axis. Meanwhile, the Y' axis is a line orthogonal to the X' axis while passing the point S_1. The coordinate system transformation formula is

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} cos\alpha & sin\alpha \\ -sin\alpha & cos\alpha \end{pmatrix} \begin{pmatrix} x - x_{S_1} \\ y - y_{S_1} \end{pmatrix} \quad (2)$$

$(x, y), (x', y')$ represent the coordinates in the coordinate system $O-XY$ and $S_1 - X'Y'$ respectively. α is the angle between the X axis and the X' axis. $(x_{S_1} y_{S_1})$ are the coordinates of S_1 in the coordinate system $O-XY$.

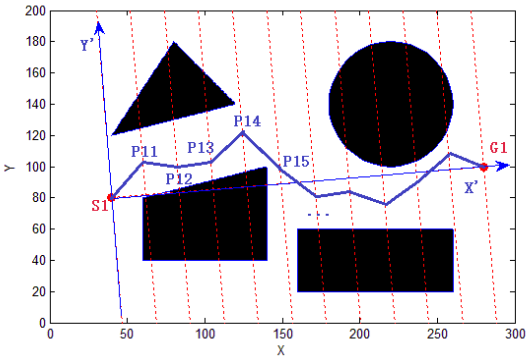


Fig. 1 Environmental model

Divide the line segment S_1G_1 into $(m+1)$ parts uniformly, and draw a line perpendicular to S_1G_1 at every equipartition point to get parallel lines $(l_{11}, l_{12}, \cdots, l_{1m})$ whose intersections with robot's path are points in the point set P_1. The robot's range of motion is the area surrounded by l_{11}、l_{1m} and the environment without the barrier region. If we define S_1 as p_{10} and G_1 as p_{1m+1}, then the length of robot 1's path L_{P_1} can be expressed as

$$L_{P_1} = L_{S_1 p_{11}} + \sum_{j=1}^{m-1} L_{p_{1j}p_{1j+1}} + L_{p_{1m}G_1} = \sum_{j=0}^{m} L_{p_{1j}p_{1j+1}} \quad (3)$$

$L_{p_{1j}p_{1j+1}}$ represents the distance between p_{1j} and p_{1j+1}. L_{P_1} can be presented in the coordinate system $S_1 - X'Y'$ as

$$\begin{aligned} L_{P_1} &= \sum_{j=0}^{m} \sqrt{(x'_{p_{1j}} - x'_{p_{1j+1}})^2 + (y'_{p_{1j}} - y'_{p_{1j+1}})^2} \\ &= \sum_{j=0}^{m} \sqrt{(\frac{L_{S_1G_1}}{m+1})^2 + (y'_{p_{1j}} - y'_{p_{1j+1}})^2} \end{aligned} \quad (4)$$

If we define $\frac{L_{S_1G_1}}{m+1}$ as d_1 to represent the length of every part of the line segment S_1G_1 after division and simplify $y'_{p_{1j}}$ to y'_{1j}, then

$$F(x) = L_{P_1} = \sum_{j=0}^{m} \sqrt{d_1^2 + (y'_{1j} - y'_{1j+1})^2} \quad (5)$$

Finally, the path planning of robot 1 can be understood as finding the optimal value of the function 5 or finding the minimum L_{P_1} in the solution space of $y'_{1j}(j = 1,2,\cdots,m)$. L_{P_1} can be used as the fitness function in PSO. Equation (1) can be changed to

$$P_1 = \{0, y'_{11}, y'_{12}, \cdots, y'_{1m}, 0\} \quad (6)$$

III. Path Planning for Multi-Robot Systems

Given the task is to make all the robots move from their start points to end points, the total time is the longest time costed by each robot rather than the addition of the time costed by all robots. Collisions can be avoided by good schedules rather than

changes in paths. What's more, the same parameters will not be urged for all robots, if just concentrate on time requirement for each robot. Based on this idea, the problem can be solved with the method described as follows:

Step 1 Plan a shortest path for each robot from its start point to end point separately. This step ensures the lower complexity and completeness for each robot.

Step 2 Use CPSM to get the time arrangements so that all robots can move on their paths designed before without collisions. This step ensures the optimality and will not influence the completeness for each robot.

These two main steps are introduced in detail respectively as follows.

A. Path planning for each robot individually

Also taking robot 1 for example, the path planning for robot 1 can use the algorithm proposed in [14], described as follows:

Step 1 Transform the coordinate system according to the robot's start point S_1 and end point G_1. Divide S_1G_1 into $(m+1)$ parts uniformly, and compute $y_{1j}'^{min}$、$y_{1j}'^{max}$ and speed extremum v_{1j}^{max}.

Step 2 Repeat the following steps when m ranges from 2 to 22 and find the best path for robot 1.

Step 3 Initialize the speed v_{1j}^{k0} and position x_{1j}^{k0} of particle $(k = 1,2,\cdots,n)$, and compute them fitness value. Initialize $pBest$ and $gBest$.

Step 4 Update the speed v_{1j}^{kt} and position x_{1j}^{kt} of particle $(k = 1,2,\cdots,n)$, and compute them fitness value. Update $pBest$ and $gBest$.

Step 5 Turn to step 4 until the number of iterations comes to the maximum $time_{max}$ or the algorithm converges.

An example can be showed to illustrate the influence of m on the result. The coordinates of S_1 are $(0,180)$ in the environment shown in Fig. 1, and G_1 is $(280,20)$. The number of particles in the particle swarm $n = 100$, and $time_{max} = 500$. Fig. 2 shows the path length of the results when m ranges from 2 to 22.

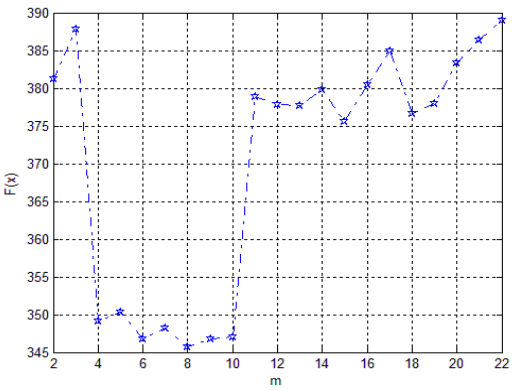


Fig. 2. The relationship between the optimal path length and particle dimension

When $m = 8$, the path length obtains the minimum $F(x) = 345.78$. The figure shows that the higher the dimension of the particles is, the shorter the path length cannot be ensured. In fact, the higher the dimension is, more freedom the line segments after division will get. The parts, that can be connected directly with a straight line when the dimension is low, must be connected with polygonal line on a high degree of freedom. It is easier to fall into local optimum in this case. Also, we need to utilize polygonal line instead of straight line at some time to get a shorter path length. So the optimal m needs to be found if using this algorithm .

B. Critical Path Schedule Method

An activity on arc (AOA) network is used to get the schedule for each robot moving on the path planned by PSO [15-16]. By using a dummy activity (indicated by a dotted arc), we can get a project diagram which only has one start node and one finish node. The time at which all robots reach their target marks the end of the whole project. Before using this project diagram to get the schedule for each robot, some concepts must be given in advance. Despite CPSM is developed from CPM, some concepts are totally different.

Definition 1: The node in the project diagram represents an actual point in the environment.

Definition 2: The arc represents an activity which means the motion of a robot moving from one point to another.

Activities don't have the precedence relationships shown in the project diagram. Because each robot has its own path, an activity does not need to wait to begin until all predecessors are completed. The beginning of an activity will be only determined by the predecessors belonging to the same robot. The nodes in the diagram don't need to be numbered as they don't have the precedence relationships. The nodes can be named as the points in the environment.

Definition 3: The duration of activity (i,j), represented by t_{ij}, is the shortest time in which the robot moves from point i to point j.

Note that different robots may have different maximum moving speed, so the duration may be different for each robot in the same activity. The nodes, arcs, and durations determine the structure of the project diagram completely. Then we can obtain some time parameters from this diagram.

Definition 4: The early event time for node i, represented by $ET(i)$, is the earliest time at which the last robot that need to pass the point passes this point.

Definition 5: The late event time for node i, represented by $LT(i)$, is the latest time at which the last robot pass the point without delaying the completion of the project.

Definition 6: The total float, represented by $TF(i,j)$, of the activity (i,j) is the amount by which the moving time of the robot from i to j can be delayed without delaying the completion of the project (assuming no other activities are delayed).

Although the meaning of $ET(i), LT(i), TF(i,j)$ is different from CPM, the values are the same compared with the values

calculated by CPM using the same duration values in definition 3. That is to say, we can use CPM to calculate $ET(i), LT(i), TF(i,j)$ after determining the structure of the project diagram.

Definition 7: The critical path is the longest path in all the moving paths for each robot planned before.

The critical path is only determined by the total path distance of each robot, so it will not be influenced by the time parameters mentioned above. And sometimes it's different from the path calculated by CPM using the same project diagram, e.g., when the path obtained by CPM is the combination of activities belonging to several robots.

Each robot has a start node and a finish node. But only the start node and the finish node of the robot on the critical path represent the start node and the finish node of the diagram. All the finish nodes of the robots not on the critical path means the time when they reach their end points and wait for the robot on the critical path arriving at the finish node of the diagram.

Definition 8: The safe time, represented by ST, is the shortest time interval between two robots passing a same point without collisions, considering the size of the robot.

Definition 9: The modified late event time for node i, represented by $MLT(i)$, is the latest time at which a specific robot pass the point without delaying the completion of the project.

$MLT(i)$ may not equal $LT(i)$ because the activities don't have the precedence relationships.

Definition 10: If we have arranged the robot to pass node i before time t, then the modified total float, represented by $MTF(i,j)$, will be $TF(i,j) + ET(i) - t$

After having a knowledge of the basic concepts, CPSM can be used to get schedule for each robot. The algorithm is described as follows:

Step 1 Use dummy activity to get a project diagram which only has one start node and one finish node.

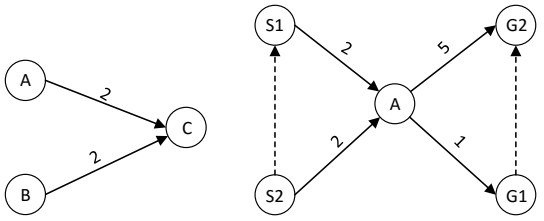
Step 2 Compute time parameters of nodes and arcs in this diagram.

Step 3 Find the critical path.

Step 4 Give the schedule of the robots. The robot on the critical path must move at its maximum speed, so the time that this robot pass the points on its path is $ET(i)$ or $LT(i)$. For other robots, the robot just needs to pass the node i before $LT(i) - ST$ when there will be only one robot to pass the node. If num robots will pass a same node i, the one whose predecessor have a longer total float will pass the node first. The time before which a specific robot pass node i equals the time, which is the earliest time at which the next robot in the sequence passes this node, minus ST, if the total float permits. Meanwhile, the last robot in this sequence must pass the node i before $LT(i) - ST$. All robots not on the critical path need to reach their finish nodes before the end of the project.

Sometimes we will have some trouble with step 4, and the settlements for these difficult cases are given below:

Case 1: As shown in Fig. 3(a), it's a part of a project diagram, and $ET(A) = LT(A) = ET(B) = LT(B) = 4$, $ET(C) = LT(C) = 6$. We can know that $TF(A,C) = TF(B,C) = 0$, so we can't get the priority between A and B in step 4. But if we have arranged the robot to pass B before 2 and $ST = 1$, then $MTF(B,C) = 2$. So the robot from B must pass the node C before 5, and the robot from A must pass the node C at time 6.

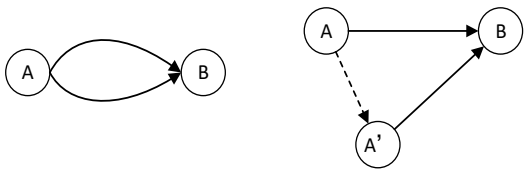


(a) Case 1 (b) Case 2

Fig. 3. Special case 1 and 2

Case 2: We can also utilize successors' total float to adjust the time. The robot on the critical path must move at the maximum speed, so the time that this robot pass the points on its path is $ET(i)$ or $LT(i)$. For other robots, the robot just needs to pass the node i before $LT(i) - ST$ when there will be only one robot to pass the node. If num robots will pass a same node i, the one whose successor have a longer total float will pass the node later. The time after which a specific robot pass node i equals $MLT(i)$ of the previous robot in the sequence plus ST, if the total float permits. Of course, each robot must pass node i before its own $MLT(i) - ST$ simultaneously. All robots not on the critical path need to get to their finish nodes before the completion of the project. A complete project diagram is shown in Fig. 3(b), and $ET(S_1) = LT(S_1) = ET(S_2) = LT(S_2) = 0$, $ET(A) = LT(A) = 2$, $LT(G_1) = ET(G_2) = LT(G_2) = 7$, $ET(G_1) = 3$, $ST = 1$. Since $MLT(A)$ of robot 2 equals 6, robot 1 should pass node A at 2 and robot 2 should pass node A from 3 to 5. Usually we hope all robots to reach their target as soon as possible, so we won't use this method if possible.

Case 3: When more than one robot moves on the same path like Fig. 4(a), we can use Fig. 4(b) to determine the time when they pass node B. $ET(A) = LT(A) = ET(A') = LT(A')$.



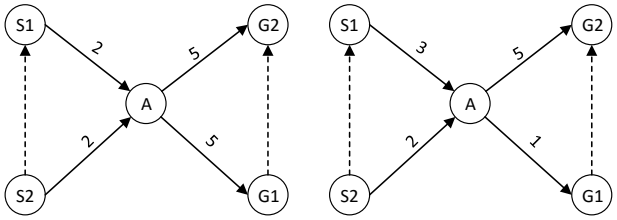
(a) (b)

Fig. 4. Special case 3

Case 4: When the total float is too short to meet the requirements, we can enlarge the duration of the activity which has the maximum duration in the predecessors before. An example shows in Fig. 5(a), $ET(S_1) = LT(S_1) = ET(S_2) = LT(S_2) = 0$, $ET(A) = LT(A) = 2$, $ET(G_1) = LT(G_1) = ET(G_2) = LT(G_2) = 7$, $ST = 1$. After enlarging $TF(S_1, A)$ to 3, $ET(S_1) = LT(S_1) = ET(S_2) = LT(S_2) = 0$, $ET(A) = LT(A) = 3$, $ET(G_1) = LT(G_1) = ET(G_2) =$

$LT(G_2) = 8$. Then, robot 1 should pass node A at 3, and robot 2 should pass node A at 2. This is the only situation when the project's time is longer than the longest time that the robot costs from its start point to its end point on its own way.

Case 5: If the critical path computed by the CPM is different from the longest path in all the moving paths for the robots planned before, we need to modify some time parameters to make sure that the critical path is we desired. As shown in Fig. 5(b), $ET(S_1) = LT(S_1) = ET(S_2) = 0$, $LT(S_2) = 1$, $ET(A) = LT(A) = 3$, $LT(G_1) = ET(G_2) = LT(G_2) = 8$, $ET(G_1) = 4$, $ST = 1$. We can modify $ET(A) = LT(A) = 2$, $LT(G_1) = ET(G_2) = LT(G_2) = 7$, $ET(G_1) = 3$. Then we can use the method mentioned in special case 2. Robot 1 should pass node A from 3 to 5 and reach node G_1 before 7. Robot 2 should pass node A at 2 and reach node G_2 at 7.



(a) Case 4 (b) Case 5

Fig. 5. Special case 4 and 5

By CPSM, we can get a schedule according to which all robots can reach their target without collisions and the time of the project is minimized. Because we just give a schedule to all robots, it needs better communication and control technique to support in implement.

IV. EXAMPLE

A complete example is given in this section to solve the path planning of autonomous mobile multi-robot. The experimental environment's size is 300×200 as shown in Fig. 1. There are 3 robots in the environment, whose start points are $(0,180), (20,120), (20,60)$ and end points are $(280,20), (300,80), (280,130)$ respectively. The number of particles in the particle swarm $n = 100$, and $time_{max} = 500$, ST=5. First of all, plan the paths for all robots separately using PSO. The optimal m for 3 robots are 8, 3, 4 respectively. Their path planning results are illustrated in Fig. 6, and the coordinates of all points are given in TABLE I. Robot 1's path is $S_1 - B - C - G_1$, and robot 2's path is $S_2 - A - C - G_2$, and robot 3's path is $S_3 - A - B - G_3$. The sum is 905.98, adding all the paths' length.

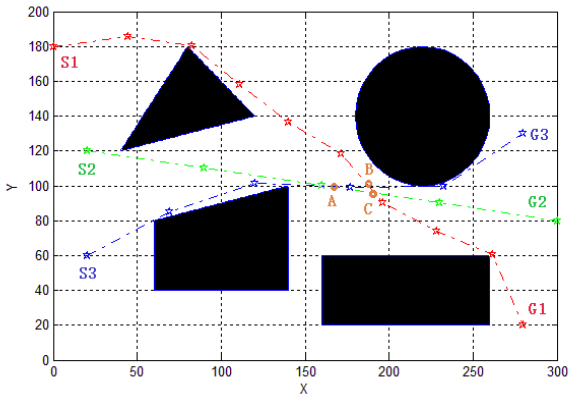


Fig. 6 The simulation results

TABLE I
THE SIMULATION PATH

Point Number	Robot 1	Robot 2	Robot 3
1	0,180	20,120	20,60
2	44.53,185.7	90.02,110.19	69.04,84.97
3	82.68,180.26	160.04,100.32	120.3,101.71
4	111.23,158	230.03,90.23	176.72,99.31
5	140.29,136.63	300,80	232.3,100
6	171.19,118.48		280,103
7	196.27,90.14		
8	228.46,74.26		
9	261.86,60.48		
10	280,20		

Assume that the maximum speed of all robots is 1 for convenience. According to the paths of all robots, we can draw a project diagram shown in Fig. 7. And the time parameters of nodes and arcs are listed in TABLE II and TABLE III respectively.

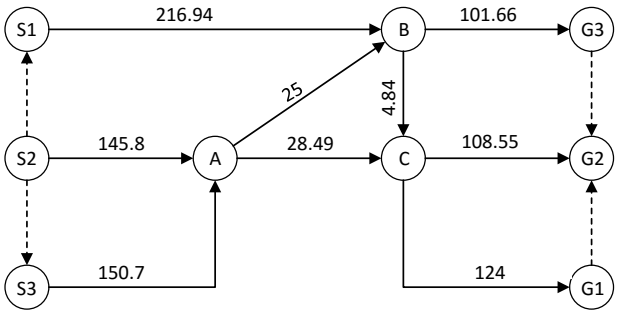


Fig. 7 The project diagram

TABLE II
THE TIME PARAMETERS OF NODES

Node(i)	ET(i)	LT(i)
S1	0	0
B	216.94	216.94
C	221.78	221.78
G1	345.78	345.78
S2	0	0
A	150.7	191.94
G2	345.78	345.78
S3	0	41.24
G3	318.6	345.78

The critical path is $S_1 - B - C - G_1$. So robot 1 must move at its maximum speed and the time when robot 1 reach its target is the task completion time. For node A, $TF(S_2, A) > TF(S_3, A)$, so robot 2 should pass node A before $150.7 - ST = 145.7$ and robot 3 should pass node A before $L(A) - ST = 186.94$. There won't be any collisions at node A in this way.

For node B, robot 1should pass node B at 216.94 since robot 1 is on the critical path. $TF(A,B) > TF(S_1,B)$, so robot 3 should pass node B first. Robot 3 should pass node B before $216.94 - ST = 211.94$ to avoid collisions. The detail schedule shows in TABLE IV.

TABLE III
THE TIME PARAMETERS OF ARCS

Activity(i,j)	Duration	TF(i,j)
S1B	216.94	0
BC	4.84	0
CG1	124	0
S2A	145.8	46.14
AC	28.49	42.59
CG2	108.55	15.45
S3A	150.7	41.24
AB	25	41.24
BG3	101.66	27.18

TABLE IV
THE SCHEDULE OF 3 ROBOTS

Robot Number	Node 1	Pass Time	Node 2	Pass Time	Node 3	Pass Time
1	*B*	216.94	*C*	221.78	*G1*	345.78
2	*A*	before 145.7	*C*	before 216.78	*G2*	before 345.78
3	*A*	before 186.94	*B*	before 211.94	*G3*	before 345.79

This is just a simple example. And we can regard the part (shown in Fig. 8) from D to A as a straight line, which robot 1 and robot 2 need to pass, to avoid the collisions when moving if necessary. According to the sizes of robots and obstacles, we can also consider the circle E including point A, B, C as a point if necessary.

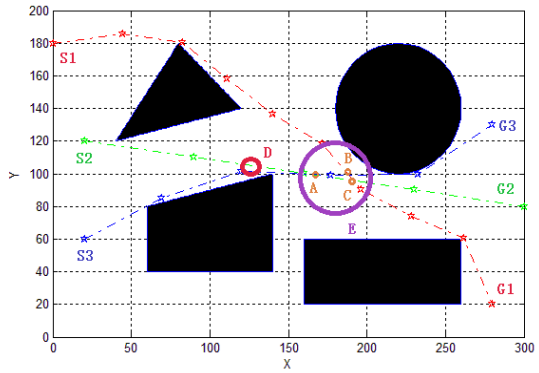


Fig. 8 The modified simulation results

V. CONCLUSIONS

Based on the CPSM, an algorithm is proposed to solve path planning for multi-robot systems, which just takes two steps and can get a satisfactory result. Different from traditional path planning for multi-robot systems, this method can't give a path without collisions or obstacle avoidance strategy but a schedule for all robots. The proposed method has a good performance in completeness, complexity and optimality. It needs more advanced communication and control technique to support the implement. The first step is a method to solve the path planning for individual robot. It's simple and fast without the dependence on the shape of obstacles in a free space. The second step is based on the CPSM which can be used to manage multiple projects using some public resources. This approach provides the possibility of transforming multi-robot problem into an individual robot problem. And it's easy to find the critical path, which influences the completion time of the task, and make the corresponding adjustment. There are also some shortages, such as it's difficult to estimate the influence on time when robots making a turn.

REFERENCES

[1] E. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," IEEE Trans. SSC, vol. 4, no. 2,pp. 100–107, Jul. 1968.

[2] J. E. Hopcroft, J. T. Schwartz, and M. Sharir, "On the complexity of motion planning for multiple independent objects: PSPACE-hardness of the 'warehouseman's problem'," Int. J. Robot. Res., vol. 3, no. 4, pp. 76–88, 1984.

[3] M. Peasgood, C. M. Clark, and J. McPhee, "A Complete and Scalable Strategy for Coordinating Multiple Robots Within Roadmaps," IEEE Trans. On Robot., vol. 24, no. 2, pp.283-292, 2008

[4] M. R. K. Ryan, "Graph decomposition for efficient multi-robot path planning," Proc. IJCAI, pp. 2003–2008., 2007

[5] K. Kant and S. W. Zucker, "Toward efficient trajectory planning: The path-velocity decomposition," Int. J. Robot. Res., vol. 5, no. 3, pp.72–89, 1986.

[6] M. Erdmann and T. Lozano-Pérez, "On multiple moving objects," Proc. IEEE Int. Conf. on Robotics and Automation, San-Francisco, CA, pp. 1419–1424., 1986

[7] C.W. Warren, "Multiple robot path coordination using artificial potential fields," IEEE Int. Conf. on Robotics and Automation, Cincinnati, OH, pp. 500–505., 1990

[8] V. J. Lumelsky and K. R. Harinarayan, "Decentralized motion planning for multiple mobile robots: The cocktail party model," Auton. Robots, vol. 4, no. 1, pp. 121–135, 1997.

[9] M. Erdmann and T. Lozano-Perez, "On multiple moving objects," Algorithmica, vol. 2, pp. 477–521, 1987.

[10] K. Azarm and G.Schmidt, "Conflict-free motion of multiple mobile robots based on decentralized motion planning and negotiation," Proc. IEEE Int. Conf. Robot. Autom., pp. 3526–3533., 1997

[11] M. Bennewitz, W. Burgard, and S. Thrun, "Optimizing schedules for prioritized path planning of multi-robot systems," Proc. IEEE Int. Conf. Robot. Autom., pp. 271–276., 2001

[12] R. Alami, F. Robert, F. Ingrand, and S. Suzuki, "Multi-robot cooperation through incremental plan-merging," Proc. ICRA, pp. 2573–2579., 1995

[13] Goyal, Jitin Kumar, and K. S. Nagla. "A new approach of path planning for mobile robots." Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on IEEE, 2014:863-867.

[14] Sun, Bo, Wei-dong Chen, and Yu-geng Xi. "Particle swarm optimization based global path planning for mobile robots." Control and Decision 20.9 (2005): 1052.

[15] Winston, W. "Operations Research: Applications and Algorithms, 4th edition." Wako Department of Economics Gakushuin University Tokyo 171-8588, Thomas Quint Department of Mathematics University of Nevada at Reno (1994).

[16] Hillier, F S, and G. J. Lieberman. "Introduction to operations research, 9th ed." McGraw-Hill international editions 464(2001):47.