Research on Dynamic Path Planning Based on the Fusion Algorithm of Improved Ant Colony Optimization and Rolling Window Method

QIBING JIN, CHUNING TANG, AND WU CAI
College of Information Science and Technology, Beijing University of Chemical Technology, Beijing 100029, China
Corresponding author: Chuning Tang (13161667785@163.com)
This work was supported by the National Natural Science Foundation of China under Grant 61673004.

ABSTRACT This paper focuses on the problem that the current path planning algorithm is not mature enough to achieve the expected goal in a complex dynamic environment. In light of the ant colony optimization (ACO) with good robustness and strong search ability, and the rolling window method (RWM) with better planning effect in local path planning problems, we propose a fusion algorithm named RACO that can quickly and safely reach the designated target area in a complex dynamic environment. This paper first improves the ant colony optimization, which greatly improves the convergence performance of the algorithm and shortens the global path length. On this basis, we propose a second-level safety distance determination rule to deal with the special problem of the research object encountering obstacles with unknown motion rules, in order to perfect the obstacle avoidance function of the fusion algorithm in complex environments. Finally, we carry out simulation experiments through MATLAB, and at the same time conduct three-dimensional simulation of algorithm functions again on the GAZEBO platform. It is verified that the algorithm proposed in this paper has good performance advantages in path planning and dynamic obstacle avoidance.

INDEX TERMS Motion planning, path planning, ant colony optimization, rolling window approach method, mobile robot.

I. INTRODUCTION
With the development and progress of the discipline, the application areas of the path planning problem have gradually expanded, becoming a key part of technological innovation and upgrading. Among them, robot obstacle avoidance search, satellite real-time navigation, unmanned aerial vehicle air environment reconnaissance, factory workshop scheduling and other issues have higher and higher requirements for the accuracy of path planning results. The algorithm research of this type of problem has also attracted the attention of scholars. According to the degree of grasp of environmental information, path planning can be divided into global path planning based on a priori complete information and local path planning based on sensor information. Global path planning algorithms can be roughly divided into traditional algorithms, graphics derived algorithms, intelligent bionics algorithms, neural network algorithms and fusion algorithms. An improved C-Space graph derivation algorithm was proposed based on deformed configuration space by Chen (2010). The concept of configuration space was introduced into the algorithm to bring the problem into a high-dimensional space to solve. The fusion algorithm combining ant colony algorithm and neural network (Jin & Guo, 2010) improved the convergence speed and avoided the algorithm from falling into the local optimum. An improved BP neural network algorithm (Zhang et al., 2017) reduced the imbalance of the data set, and improved the classification accuracy of small data sets significantly. By combining the fuzzy logic system with the sliding film control algorithm, Ge et al. (2018) proposed an adaptive fuzzy sliding film trajectory tracking control algorithm, which reduced input chattering and enhanced the trajectory tracking control accuracy effectively. In order to solve the problem of convergence and local optimal solution of traditional planning methods in complex environment, Han et al. (2018) improved the planning ability of the algorithm by combining repulsive potential field with particle swarm optimization.

The associate editor coordinating the review of this manuscript and approving it for publication was Yingxiang Liu.
Local path planning algorithms mainly include artificial potential field method and rolling window method. The path of the traditional artificial potential field method was optimized by Qu et al. (2018) to avoid the oscillation problem. Xi et al. (2020) solved the problem of local sub-target selection in dynamic window method through heuristic function. In recent years, neural network and fuzzy control theory have been widely used in local path planning. With the continuous development of path planning technology, a single algorithm can no longer meet the needs of practical problems. Therefore, the fusion algorithm of path planning has a good prospect and is very necessary in solving dynamic and complex environmental problems.

The contributions of RACO proposed in the paper to the path planning problem are as follows. 1) During the algorithm iteration process, the pheromone concentration is dynamically adjusted to obtain the global optimal solution faster and avoid the algorithm from falling into the local optimal solution. 2) The improved algorithm performs secondary planning on the obtained path to minimize the path length. 3) In the face of various obstacles in complex environments, RACO can adopt corresponding obstacle avoidance strategies, and reduce energy consumption and enhance efficiency.

This paper is structured as follows. Section 2 describes the basic principles of ant colony optimization and rolling window method. Section 3 first explains the improvement of the ant colony optimization, then presents various obstacle avoidance strategies in complex environments, and finally combines the ant colony optimization and the rolling window method. In section 4 we conduct simulation experiments on the improved ant colony optimization and fusion algorithm while analyzing the experimental results. Section 5 gives the conclusion of this paper.

II. ALGORITHM INTRODUCTION

A. ACO

The ant colony optimization is an efficient heuristic algorithm. The operation process of the algorithm adopts a distributed computing method. Multiple individuals perform parallel computing at the same time, which greatly improves the computing power and operating efficiency of the algorithm [13].

The ant colony optimization is a random heuristic search algorithm that mimics the foraging behavior of ants, leaving behind chemicals called pheromones along their path. (Song, 2013) When choosing the next direction, the ants are judged by the concentration of pheromones in the path, and the higher the concentration, the more likely the path is to be chosen. With the continuous search of individual ants, the pheromone concentration on the superior path becomes higher and higher, and finally the ant colony gets an optimal path.

The ant chooses the direction of movement mainly based on the pheromone concentration on the path. In the $t$ iteration, the probability of the ant $K$ moving from node $i$ to node $j$ is shown in formula (1).

$$p^k_{ij}(t) = \begin{cases} \left[\tau_{ij}(t)^{\alpha} \prod_{s \in \text{allowed}} \eta_{is}(t)^{\beta}\right]^{1/\sum_{s \in \text{allowed}} \tau_{is}(t)^{\alpha} \prod_{s \in \text{allowed}} \eta_{is}(t)^{\beta}} & j \in \text{allowed} \\ 0 & \text{otherwise} \end{cases}$$

where $\alpha$ is the pheromone heuristic factor, the larger the value, the smaller the randomness of the algorithm in the search process. $\beta$ is the expected heuristic factor. If the value of this parameter is too large, the algorithm will fall into the local optimum easily. $\tau_{ij}$ is the pheromone concentration of nodes $i$ to $j$; $\text{allowed}$ represents the set of all advancing nodes found; $\eta_{ij}$ is heuristic information; $d_{ij}$ is the Euclidean distance from node $i$ to $j$. The relationship between $\eta_{ij}$ and $d_{ij}$ can be defined as formula (2).

$$\eta_{ij} = 1 / d_{ij}$$

After all the ants complete one iteration, the pheromone on the path is volatilized, and the pheromone concentration on the path $i$ to $j$ at time $t+1$ is configured as formula (3).

$$\tau_{ij}(t+1) = (1 - \rho) \tau_{ij}(t) + \Delta \tau_{ij}(t, t+1)$$

In the above formula, the increase in pheromone concentration from $t$ to $t+1$ is shown in (4).

$$\Delta \tau_{ij}(t, t+1) = \begin{cases} \frac{Q}{L} & \text{The path that Ant K passes through this iteration} \\ 0 & \text{Other} \end{cases}$$

where $\rho$ is the volatilization coefficient of the pheromone, and the value range is between 0 and 1. $\Delta \tau$ represents the pheromone increment produced by the ant through the process; $Q$ is the initial pheromone intensity; $L$ denotes the ant through the current iteration the total path length.

B. RWM

The core idea of the rolling window method draws on the relevant principles of predictive control, and is mostly used for local path planning of mobile robots. (Liang & Song, 2010) It can calculate the maximum collision-free speed required for the robot to reach the target. The green line in Figure 1 represents the sampling trajectory of the search space.

The rolling window method can be implemented in three steps: (1) Scene prediction: The mobile robot predicts the motion trajectory of dynamic obstacle by obtaining information within the sensing range, and determines whether a collision will occur based on its own motion law. (2) Rolling window optimization: According to the prediction results and environmental information, the corresponding strategy is adopted to find the local sub-target points and the mobile robot move accordingly. The detection range is continuously updated with the movement of the robot, entering a new detection cycle. (3) Feedback initialization: Each rolling plan
provides the latest data for the next local path selection, and initializes the obstacles in the sensing range in view of the known information [22].

In the constantly refreshing window, the local sub-target point can be regarded as the mapping of the end point of the planned path in the window. At time \( t \), the current window is denoted as \( W(t) \), the boundary is \( \partial W(t) \), and the local sub-target point \( P_{Gi} \) generation rule can be defined as:

\[
P_{Gi} = \begin{cases} 
G & G \in W(t) \\
\min f(P_1) \text{ s.t. } P_1 \in \partial W(t) & G \notin W(t)
\end{cases}
\] (5)

If the global end point \( G \) is in the current window, the local sub-target point is \( G \). Otherwise, the window boundary point \( P_1 \) that minimizes \( f(P_1) \) is selected as the sub-target point. The heuristic function \( f(P_1) \) is described as formula (6).

\[
f(P_1) = g(P_1) + h(P_1)
\] (6)

In the formula, \( g(P_1) \) is the cost of the robot traveling from the current position \( P \) to \( P_1 \), and \( h(P_1) \) is the cost of traveling from \( P_1 \) to the end point \( G \). The calculation of \( f(P_1) \) draws on the heuristic function calculation method of \( A^* \) algorithm. When there is no obstacle in the detection window, \( P_{Gi} \) is the intersection of the line connecting the current position of the mobile robot and the global end point \( G \) at the window boundary as shown in Figure 2. At this time, the local sub-goal point meets the requirement of advancing toward the global end point on the shortest path without obstacles.

When an obstacle is detected in the detection window, the selection of \( P_{Gi} \) is shown in Figure 3, and \( P_1 \) is a local sub-target point. We can see that the local sub-target point selected at this time can meet the requirements of moving toward the global end point while the mobile robot safely avoids obstacle.

FIGURE 1. Sample trajectory of search space.

FIGURE 2. When there is no obstacle in the window, the local sub-target point is determined by calculation to be the intersection of the mobile robot and the global end point \( G \) at the boundary of the window.

FIGURE 3. When there is an obstacle in the window, the local sub-target point position can be obtained by calculation, which ensures that the mobile robot can safely avoid the obstacle and move toward the global end point \( G \).

III. ALGORITHM IMPROVEMENT
A. IMPROVED ANT COLONY OPTIMIZATION

Although ant colony optimization has many advantages in solving path planning problems, it also has some problems such as easy to fall into deadlock and local optimal solutions. To solve these problems and improve the convergence speed of the algorithm as well as optimize the path length, in this section, the initializing part of the ant colony optimization, the state transition probability formula and the pheromone updating mechanism are optimized. Meanwhile, quadratic programming is carried out on the excellent path obtained by the improved algorithm to finally obtain the optimal path solution.

The initialization process of ant colony optimization is improved in two aspects. First of all, compared to the uniform distribution of pheromone concentrations on the map in the traditional ant colony optimization, we adopt the strategy of increasing the initial pheromone concentration by selecting the global favorable area for the starting point and the target point to improve the convergence speed of the algorithm. On the other hand, for ants in the process of searching for paths, if they encounter a disconnected path and no path is feasible, the algorithm is prone to stagnation. This paper introduces a fallback strategy. When the ant falls into the trap and cannot move forward, it returns to the previous node. If it has not escaped the “trap”, it continues to return to the previous node until it exits. Then according to the state transition probability formula combined with the roulette method, the next node is reselected to avoid falling into the local optimum and reduce the number of ant deaths.

The state transition probability formula determines the path selection during the operation of the algorithm. The values of the pheromone heuristic factor \( \alpha \) and the expected heuristic factor \( \beta \) both have an important impact on the randomness of the path search. If the value of \( \alpha \) is too large or too small, the algorithm may fall into the local optimum. The larger the value of \( \beta \) is, the ants tend to choose the local shortest path closer to the target point at a certain point, which is not conducive to searching for the global optimal solution. In summary, we make the following improvements. In the initial stage of the iteration, \( \alpha \) and \( \beta \) are set to the minimum, thereby increasing the global search ability in the initial stage of the algorithm and raising the efficiency of finding the
When we adjust the volatilization coefficient dynamically, the same, the parameter $\rho$ the path length resulting from three consecutive iterations is

\[ \alpha = \begin{cases} \alpha_{\text{max}} - (\alpha_{\text{max}} - \alpha_{\text{min}}) & n \geq N \\ \alpha_{\text{min}} & \end{cases} \quad (7) \]

\[ \beta = \begin{cases} \beta_{\text{max}} - (\beta_{\text{max}} - \beta_{\text{min}}) & N \geq n \\ \beta_{\text{min}} & \end{cases} \quad (8) \]

Among them, $\alpha_{\text{min}}$ and $\beta_{\text{min}}$ are the initial values of $\alpha$ and $\beta$, respectively. $\alpha_{\text{max}}$ and $\beta_{\text{max}}$ are the maximum values, and $N$ is the current iteration number.

The traditional ant colony optimization uniformly updates the pheromone concentration in the environment after each iteration, and cannot make personalized adjustments to the high-quality path. At the same time, the algorithm has the risk of falling into a local optimum. For the purpose of avoiding the above problems, in this section, the pheromone update mechanism is improved, and the volatilization coefficient is dynamically adjusted to avoid falling into the local optimum, which further promotes the efficiency of the algorithm in finding the optimal solution.

The improved pheromone update mechanism is as follows. After each iteration, only the pheromone of the path that the ant has reached the target point is updated, and the path that has not reached the target point is discarded, thereby avoiding blind update of the global pheromone. The pheromone concentration update is shown in formula (9). After selecting a high-quality path from many paths, the pheromone concentration on the path is updated in a guiding way to increase the pheromone concentration on the optimal path, which is represented as (10).

\[ \tau_{ij}(t + 1) = (1 - \rho) \tau_{ij}(t) + \rho \Delta \tau_{ij}(t, t + 1) \quad (9) \]

\[ \Delta \tau_{ij}(t, t + 1) = \begin{cases} \frac{Q}{L} & \text{The path that Ant K passes through this iteration} \\ 0 & \text{Other} \end{cases} \quad (11) \]

where $\rho$ is the pheromone volatilization coefficient; $\Delta \tau$ represents the pheromone increment generated by the ant in this iteration through the path; $\lambda$ is the pheromone amplification coefficient, and $Q$ is the pheromone intensity. For the purpose of avoiding the algorithm from falling into the local area during operation optimally, this paper makes adaptive adjustments to the pheromone volatilization coefficient $\rho$. When the path length resulting from three consecutive iterations is the same, the parameter $\rho$ is dynamically adjusted, as shown in formula (12).

\[ \rho(t + 1) = \begin{cases} \theta \cdot \rho(t) & \theta \cdot \rho(t) \geq \rho_{\text{min}} \\ \rho_{\text{min}} & \text{Other} \end{cases} \quad (12) \]

In the formula, $\theta$ is the pheromone attenuation coefficient. When we adjust the volatilization coefficient dynamically, the algorithm continues to iterate. If the path length obtained is less than the historical optimal solution, it means that the algorithm has fallen into a local optimal. In order to continue searching for the global optimal path, it is necessary to update the pheromone concentration on the map again according to formula (9) (10), and repeat the search process until the optimal path requirement is met or the maximum number of iterations is reached.

At present, the path obtained by the algorithm is the result of the ideal map model established by the grid method. Energy loss and time cost are not taken into account. Therefore, the paper proposes a secondary planning strategy for the path. After the algorithm completes the path planning, we will find all the node information in the path, and make every three consecutive nodes form a closed triangle. Then, we judge whether the end point of the adjacent infeasible area is within the triangle. If the end point is not in the triangle, the intermediate node can be deleted to optimize the path again, and complete one more iteration according to the above rules, we will obtain the shortest safe path, as shown in Figure 4.

The flow chart of the improved ant colony optimization is shown in Figure 5.

**B. LOCAL PATHING PLANNING**

This paper chooses the rolling window method to solve the local path planning problem, and expands the rolling window perception range into a circular area. In the simulation environment, a particle is selected to simulate the mobile robot to execute the obstacle avoidance strategy. Combining the problem characteristics of dynamic path planning, we fully analyze the situation of particles and dynamic obstacles encountered in the environment. At the same time, for the problem of dynamic obstacle avoidance with unknown motion rules, The paper proposes a second-level safety distance determination rule, which improves the algorithm’s obstacle avoidance function in complex environments.

For dynamic obstacles with known motion rules, the following four coping strategies are proposed according to the trajectories of the particle and the dynamic obstacle in a rolling window period. (1) If the particle and the dynamic obstacle do not intersect in the motion trajectory within a cycle perceived by the dynamic window, no collision will occur, and obstacle avoidance strategy is not required. The particle can move toward the end according to the known
FIGURE 5. After the algorithm is initialized, we get the sub-optimal solution of the path through preliminary improvement of the algorithm and the sub-optimal solution is secondarily planned to output the optimal path.

(2) If the motion trajectories predicted by the particle and the dynamic obstacle in one cycle have an intersection point at the same time and they are moving sideways. The particle and the dynamic obstacle will collide sideways, and the obstacle avoidance strategy is executed at this time. The particle decelerates and brakes, and restarts after the dynamic obstacle passes the virtual collision point and meets the safety distance. The obstacle avoidance effect is shown in Figure 6.

(3) If the motion trajectories predicted by the particle and the dynamic obstacle in one cycle have an intersection point at the same time and they are moving in opposite directions. The particle and the dynamic obstacle will collide frontally, and the obstacle avoidance strategy is executed at this time. The predicted virtual collision point is regarded as a static obstacle to avoid, and then according to the improved global path planning algorithm, the local path of the particle from the current position to the next sub-target point is planned. The obstacle avoidance effect is shown in Figure 7.

(4) In one cycle, if the particle predicts that there is a dynamic obstacle moving in the same direction with a higher speed behind it and will collide with the particle at some point in the future, it is necessary to implement an obstacle avoidance strategy. The strategy is the same as that in case (3), and the obstacle avoidance effect is shown in Figure 8.

When a dynamic obstacle is detected to change in speed direction or size during two consecutive environmental refreshing processes, it is regarded as a dynamic obstacle with unknown motion law. For the purpose of avoiding collision with such obstacles, this paper proposes a two-level safety distance determination rule, and adopts obstacle avoidance strategies according to this rule. The rule divides the distance between the moving particle and the dynamic obstacle into two categories: controllable safety distance and emergency safety distance. Reasonable values for the two types
of distances can shorten the global path length to the greatest extent and reduce energy consumption. The regional division is shown in Figure 9.

When the moving particle is determined to encounter a dynamic obstacle with unknown motion law, the velocity decreases to $1/2$ of the initial velocity.

$$v(t+1) = \frac{v}{2}$$ (13)

Determine whether the distance between the moving particle and the dynamic obstacle after the next environmental information refresh time $t$ meets the requirement of controllable safety distance or is outside the sensing range. If the requirements are met, the particle point continues to move for $t$ time along the planned path. If moving obstacle will enter the emergency safety distance, the circular region with a dynamic obstacle as the center and an emergency safety distance length as the diameter is assumed to be a static obstacle, and the local path of the particle from the current position to the next sub-target point is planned and moved along the new local path. At this time, the velocity of the moving particle is restored to the initial velocity.

In the process of particle moving along the new planned path, it is judged whether the distance between them meets the emergency safety distance requirement after a certain refresh time. If the requirement is met, the particle continues to move along the new path. Otherwise, stop advancing immediately and return along the passed path until the distance between the two is greater than the emergency safety distance again. After the obstacle gradually moves away, the particle continues to move towards the target point. During the movement, according to the above strategy, always ensure that the distance between the two is greater than the emergency safety distance. The flow diagram of the rolling window method is shown in Figure 10.

C. FUSION ALGORITHM

In section A, the global optimal path is obtained by improving the traditional ant colony optimization, but this algorithm cannot effectively avoid dynamic obstacles in complex environments. Section B describes that the rolling window method realizes local path planning by making corresponding dynamic obstacle avoidance strategies for real-time acquisition of environmental information, but the algorithm lacks global prior information, and the planned path often cannot meet the optimal standard. Therefore, this paper combines the two algorithms and proposes a fusion algorithm based on the improved ant colony algorithm-rolling window method (RACO), which enables mobile robots to achieve autonomous obstacle avoidance in a dynamic environment in view of obtaining global information. The specific steps can be divided into the following three parts.

1. First, the improved ant colony optimization is used to obtain the global optimal path in the known map environment, and the coordinates of each turning point in the path are recorded. (2) Next, we take each coordinate point as the sub-target point in the local path planning, and use the rolling window method to make the moving particle reach these target points in sequence. If the particle encounters a dynamic obstacle during the movement, the obstacle avoidance strategy is activated. (3) The moving particle passes through each sub-target point in turn and then reaches the target area and stops. The fusion algorithm finally realizes the path planning function in the complex environment. The flow chart of fusion algorithm is shown in Figure 11.

IV. SIMULATION EXPERIMENT ANALYSIS

The simulation experiment is divided into three parts to verify the path planning function of the fusion algorithm proposed in this paper in complex environment and its application value in solving practical problems. First of all, the fusion algorithm is used in the MATLAB environment for global path planning experiments, and the experimental data of the improved ant colony optimization in this paper is compared with other similar algorithms. Then, dynamic obstacles with different motion laws are introduced into the global optimal path. According to the environmental changes in the map, the algorithm executes the corresponding dynamic obstacle avoidance strategies according to the types of obstacles until reaching the end point.
avoidance strategy. Finally, using the GAZEBO platform to create a three-dimensional space scene, we select the Turtlebot robot in the platform to perform a three-dimensional simulation of the fusion algorithm to simulate and solve the actual problems in production and transportation.

A. IMPROVED ANT COLONY OPTIMIZATION SIMULATION EXPERIMENT

In the MATLAB simulation experiment, we use the grid method to establish a two-dimensional static map model. The map is composed of 20*20 grids, each of which has a length and width of 1. The black parts represent static obstacles, the white parts represent feasible areas, and the particle can move arbitrarily within the white range. We set the starting point of the moving particle at the upper left corner of the grid map, the end point at the lower right corner, and the obstacle positions are randomly set. In this section, we compare the improved ant colony optimization proposed in this paper with the classic ant colony algorithm and the other three improved ant colony optimization by predecessors [20], [22], [23] in the same Simulation experiment in map model. Each group of experiments four times, we will compare the number of iterations required for different algorithms to obtain the global optimal path and the shortest path length, and analyze the experimental data.

In the following experiment, we ensure that the basic parameter settings of different algorithms are the same, the number of ants is 50, and the maximum number of iterations is 100. The values of special parameters in other algorithms follow the authors’ recommendations in the corresponding article. The parameter values of the improved algorithm in this paper are obtained after repeated experiments, as shown in Table 1.

Compare the results from the global path. The red solid line in every map model represents the global path finally obtained by the improved algorithm. We compare the four paths obtained by the improved algorithm with the paths obtained by other algorithms on the corresponding map. It can be seen that the path results given by different algorithms are relatively safe and effective. From the analysis of the data in Table 2, the global path obtained by the improved algorithm in this paper is better than the results of other algorithms in terms of route length, shortening by two units on average, and the path length obtained after many experiments is the same basically.

Compare the convergence performance of the algorithm. In the case of the same map model, the number of iterations required by the improved algorithm in this paper is significantly less than that of the previous three groups of comparison algorithms. In the first three groups of experiments, the algorithm in this paper only took 1/3 of the time of the control group to complete the path search task. In the fourth set of comparative experiments, the iteration times of the two algorithms are almost the same, and the time used is the same basically. In summary, we conclude that the improved ant colony optimization in this paper converges faster in global

---

**TABLE 1. The parameter values of the improved ant colony optimization proposed in this paper.**

| Parameter                  | Value |
|----------------------------|-------|
| Ant number                 | 50    |
| The maximum number of iterations | 100   |
| Initial pheromone concentration | 1   |
| $\alpha_{max} \beta_{min}$ | 1.4   |
| $\alpha_{min} \beta_{max}$ | 4.9   |
| Number of iterations $n$  | 5     |
| $\lambda$                 | 1.2   |
| $\rho$                    | 0.3   |
| $\theta$                  | 0.85  |
| $\rho_{max}$              | 0.2   |

**TABLE 2. Number of iterations/minimum path length in four sets of comparative experiments.**

| Test case | RACO ACO | RACO ACO1 | RACO ACO2 | RACO ACO3 |
|-----------|----------|-----------|-----------|-----------|
| Test 1    | 13/27.73 | 40/28.55  | 19/30.21  | 53/32.20  | 13/35.64  | 29/38.10  | 14/29.59  | 11/30.10  |
| Test 2    | 14/27.73 | 42/28.55  | 19/30.21  | 53/32.20  | 15/35.64  | 29/38.10  | 13/29.59  | 11/30.10  |
| Test 3    | 15/27.73 | 42/28.55  | 19/30.21  | 53/32.20  | 15/35.74  | 29/38.10  | 12/29.76  | 11/30.10  |
| Test 4    | 16/27.73 | 43/28.55  | 20/30.21  | 53/32.20  | 17/35.74  | 29/38.10  | 13/29.76  | 11/30.10  |
| Average   | 15/27.73 | 42/28.55  | 19/30.42  | 53/32.20  | 15/36.56  | 29/38.10  | 13/29.68  | 11/30.10  |
path planning, the output path results are high-quality and the distance is short, and the algorithm has good robustness.

### B. DYNAMIC OBSTACLE AVOIDANCE SIMULATION EXPERIMENT

In this section, we use the map model of the first group of comparison experiments in 4.1, and we conduct simulation experiments on various obstacle avoidance functions of the rolling window method based on the known global optimal path. We set three kinds of dynamic obstacles in different areas of the map, which may cause side collisions, frontal collisions and random collisions with the moving particle. In the first two experiments, the obstacle movement law is known. In the third experiment, we set the movement route for the obstacle and simulate the random movement obstacle. Meanwhile, the influence of obstacle size is ignored in the experiment. The parameters of the moving particle and dynamic obstacle are shown in Table 3, and the path planning results are shown in Figure 13.

In the simulation experiment one, the dynamic obstacle moves in a straight line from left to right at a constant speed, and then stays still after a period of movement. When the dynamic obstacle enters the sensing range of the moving particle, the algorithm determines that the two will collide sideways at a certain position on the map. At this time, the moving particle decelerates and stops. After the dynamic obstacle passes the virtual collision point, the particle starts moving again and follows the known global path to the end of the map.

In the simulation experiment two, the dynamic obstacle moves in a straight line at a uniform speed in the direction shown by the arrow in the figure, and it will collide with the particle head-on during the movement. When the moving particle senses that the speed of the dynamic obstacle is opposite to itself and gradually approaches, the particle decelerates and stops. At the same time, the algorithm will take the virtual collision point as the center of the circle, and the controllable safety distance length is the diameter of the imaginary static obstacle. Then the algorithm plans the local path of the particle from the current position to the next sub-target point. The particle continues to move along the new local path to the global end point.

In the third simulation experiment, in order to verify the obstacle avoidance function of the algorithm when encountering obstacles with unknown motion rules and achieve the expected results of the experiment. We set up the motion trajectory for the obstacle and specify the speed range. The speed of the obstacle can be adjusted arbitrarily in this range. After repeated experiments, the following renderings are selected for analysis. When the moving particle finds the dynamic obstacle, it moves at a slower speed along the original path until the distance between the two is about to be less than the emergency safety distance. At this time, the algorithm regards the current position of the dynamic obstacle as the center of the circle and the emergency safety distance length as the diameter as the static obstacle. The algorithm plans the local path of the particle from the current position to the next sub-target point and restores the initial velocity. When the particle moves along the local path, the dynamic obstacle returns and approaches again. The particle retreats along the planned path according to the algorithm function until the distance between the two is greater than the controllable safety distance. After the obstacle is far away, the particle moves along the local path toward the end point and finally arrives.

Based on the above experimental analysis, we conclude that under the condition that the global optimal path is known, the rolling window method combined with the obstacle avoidance strategy proposed in this paper can realize the path planning function in the complex environment, which verifies the effectiveness of the algorithm in this paper.

### C. GAZEBO PLATFORM SIMULATION EXPERIMENT

In the process of factory production of products, assembly line is a basic production mode. However, how to efficiently and accurately distribute raw materials to the stations on the
assembly line is an important issue. In this section, we use the GAZEBO platform to create three-dimensional scenes for simulation experiments, combined with the fusion algorithm proposed in this paper, to solve the transportation scheduling problem.

GAZEBO is a powerful open source physical simulation environment. It can simulate the three-dimensional physical world, with functions such as changing the quality and color of objects, providing light sources, and generating sensor feedback. In the experiment of this section, we used the module that comes with the software to create a simple factory workshop production environment on the GAZEBO platform, as shown in the figure below. We allow the robot to smoothly reach a specified location in the environment according to the function of the fusion algorithm, thereby simulating the information transmission and logistics transportation functions within the factory floor.

We select Turtlebot robot as the research object and program it. Then, configure the robot through Move_base to solve the problems of action planning, kinematics analysis, collision detection and perception in the environment. During this experiment, we will no longer use the particle as the research object, so we must consider the size of the robot in the virtual environment. The two safety distance lengths and perception ranges stipulated by the dynamic window law in the preceding content should also be redefined according to actual conditions.

The realization of the algorithm function is mainly divided into the following parts: The virtual sensor configured by the Turtlebot robot collects environmental information and publishes the sensor information to the ROS node communication network through the GAZEBO interface. The robot system subscribes to the sensor data in the network and plans the local motion trajectory. The control node outputs the robot speed control command according to the trajectory. The speed control command is transmitted to the simulation platform through the GAZEBO interface, and finally the robot realizes the movement and obstacle avoidance in the simulation environment. In the experimental link of the GAZEBO platform, we always set the velocity of the Turtlebot robot to 1 and the velocity of dynamic obstacle to 0.5, which is the same as in the MATLAB simulation experiment.

In order to further verify the obstacle avoidance function of the algorithm, we introduced the Kuka robot as a dynamic obstacle in the environment. Figure 16 shows that the Turtlebot robot encounters the Kuka robot moving from the side while passing through the intersection. The sensor senses that a moving obstacle is approaching and judges that a side collision is about to occur. At this time, the algorithmic obstacle avoidance function is activated. Through the information
interaction between the sensor and ROS, the speed control command is transmitted to the simulation platform and finally transmitted to the Turtlebot, and the robot is controlled to decelerate and stop in place immediately. After the Kuka robot passes the intersection and the distance from the Turtlebot meets the requirements, the speed control command again controls the Turtlebot to start, and the robot continues to move toward the target point.

For the purpose of verifying the effect of the fusion algorithm in dealing with the frontal collision problem in the simulation platform, we adjusted the Kuka robot to the vicinity of the intersection again to move it towards the upcoming Turtlebot robot. Figure 17 shows the obstacle avoidance process of Turtlebot avoiding the Kuka robot according to the speed control command. When the sensor senses a frontal collision with an obstacle, the robot system immediately takes obstacle avoidance measures based on the fusion algorithm, assumes the virtual collision point in the environment as a static obstacle, and re-plans the global path from the current position to the target area. Then, Turtlebot changes the speed direction according to the latest speed control instructions and moves along the new global path, thus avoiding collisions with obstacles.

The above experimental results show that the fusion algorithm proposed in this paper can quickly obtain the global optimal path in a three-dimensional map environment and guide the robot to safely reach the target area. While the algorithm realizes the established functions, it also provides valuable experience for solving logistics and transportation problems.

V. CONCLUSION

This paper proposes a fusion algorithm (RACO) that combines improved ant colony optimization and rolling window method to solve path planning problems in complex environments. By optimizing the parameters, improving the algorithm structure, and fully considering various problems that may arise, we have improved the universality and application value of the algorithm. The improved ant colony optimization in this paper is compared with the classic ant colony optimization and three other improved algorithms in simulation experiments. Experimental results show that our improved algorithm has obvious advantages in terms of convergence speed and path planning results. In the dynamic simulation experiment, the fusion algorithm can guide the research object to avoid all kinds of dynamic obstacles to reach the target point, which verifies the obstacle avoidance performance of the algorithm in the complex environment.

In this paper, a simple factory workshop environment is created using the GAZEBO platform. The Turtlebot robot is used as the research object and combined with the fusion algorithm to solve the logistics transportation problem in this environment. In the simulation experiment, Turtlebot realized the obstacle avoidance function in the three-dimensional map environment, and finally reached the designated target area. This further verifies the effectiveness of the fusion algorithm in the three-dimensional map environment and the ability to solve practical problems.

In future studies, we will further optimize the improved ant colony optimization to ensure that the results of each run are consistent and enhance the convergence performance of the algorithm. In terms of obstacle avoidance, we will consider more unexpected situations that may occur in the environment and broaden the scope of application of the fusion algorithm.

REFERENCES

[1] M. Ceriotti and M. Vasile, “MGA trajectory planning with an ACO-inspired algorithm,” Acta Astronautica, vol. 67, nos. 9–10, pp. 1202–1217, Nov. 2010.
[2] A. H. Karami and M. Hasanzadeh, “An adaptive genetic algorithm for robot motion planning in 2D complex environments,” Comput. Electr. Eng., vol. 43, pp. 317–329, Apr. 2015.
[3] Y. Li, K. Mei, and Y. Liu, “Improving the area efficiency of ACO-based routing by directional pheromone in large-scale NoCs,” Microprocessors Microsystems, vol. 45, pp. 81–94, Aug. 2016.
[4] C. Henkel, A. Bubeck, and W. Xu, “Energy efficient dynamic window approach for local path planning in mobile service robotics,” IFAC-PapersOnLine, vol. 49, no. 15, pp. 32–37, 2016.
[5] A. M. Zanchettin and P. Rocco, “Motion planning for robotic manipulators using robust constrained control,” Control Eng. Pract., vol. 59, pp. 127–136, Feb. 2017.
[6] D. Baeza, C. F. Ihle, and J. M. Ortiz, “A comparison between ACO and dijkstra algorithms for optimal ore concentrate pipeline routing,” J. Cleaner Prod., vol. 144, pp. 149–160, Feb. 2017.

[7] X. Pu, J. Li, H. Wu, and Y. Zhang, “Mobile robot multi-goal path planning using improved particle swarm optimization,” CAAI Trans. Intell. Syst., vol. 12, pp. 301–309, Jun. 2017.

[8] C. Ordonez, N. Gupta, B. Reese, N. Seegmiller, A. Kelly, and E. G. Collins, “Learning of skid-steered kinematic and dynamic models for motion planning,” Robot. Auto. Syst., vol. 95, pp. 207–221, Sep. 2017.

[9] J. Calle, J. Rivero, D. Cuadra, and P. Isasi, “Extending ACO for fast path search in huge graphs and social networks,” Expert Syst. Appl., vol. 86, pp. 292–306, Nov. 2017.

[10] S. Pellegrinelli, A. Orlandini, N. Pedrocchi, A. Umbrico, and T. Tolioli, “Motion planning and scheduling for human and industrial-robot collaboration,” CIRP Ann., vol. 66, no. 1, pp. 1–4, 2017.

[11] M. G. Mohanan and A. Salgoankar, “A survey of robotic motion planning in dynamic environments,” Robot. Auto. Syst., vol. 100, pp. 171–185, Feb. 2018.

[12] Y. Zhao, X. Chai, F. Gao, and C. K. Qi, “Obstacle avoidance and motion planning scheme for a hexapod robot Octopus-III,” Robot. Auton. Syst., vol. 103, pp. 199–212, Mar. 2018.

[13] X. Zhang and J. Liu, “Effective motion planning strategy for space robot capturing targets under consideration of the berth position,” Acta Astronaut., vol. 148, pp. 403–416, Jul. 2018.

[14] R. Meyes, C. Scheiderer, and T. Meisen, “Continuous motion planning for industrial robots based on direct sensory input,” Procedia CIRP, vol. 72, pp. 291–296, Jan. 2018.

[15] H. Wang, F. Guo, H. Yao, S. He, and X. Xu, “Collision avoidance planning method of USV based on improved ant colony optimization algorithm,” IEEE Access, vol. 7, pp. 52964–52975, Mar. 2019.

[16] S. Mescher, “Oriented robot motion planning in Riemannian manifolds,” Topology Appl., vol. 258, pp. 1–20, May 2019.

[17] E. J. Molinos, Á. Llamazares, and M. Ocaña, “Dynamic window based approaches for avoiding obstacles in moving,” Robot. Auton. Syst., vol. 118, pp. 112–130, Aug. 2019.

[18] C. Xiong, D. Chen, D. Lu, Z. Zeng, and L. Lian, “Path planning of multiple autonomous marine vehicles for adaptive sampling using Voronoi-based ant colony optimization,” Robot. Auto. Syst., vol. 115, pp. 90–103, May 2019.

[19] S. Aggarwal and N. Kumar, “Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges,” Comput. Commun., vol. 149, pp. 270–299, Jan. 2020.

[20] C. Lei, H. Zhao, and N. Jiang, “Robot path planning based on particle swarm optimization and ant colony fusion algorithm,” J. Chongqing Univ. Nat. Sci., vol. 43, pp. 235–241, Jan. 2020.

[21] Y. Yang, J. Ni, Y. Chen, and Y. Chen, “Improved indoor robot path planning algorithm of RRT*,” Comput. Meas. Control., vol. 28, pp. 241–245, Jan. 2020.

[22] L. Zhang, L. He, D. Wu, and Z. Du, “Research on improved ant colony algorithm in path planning,” Manuf. Automat., vol. 42, pp. 55–59, Feb. 2020.

[23] C. Fang and P. Sun, “Path planning of robot based on improved ant colony algorithm,” Meas. Control Technol., vol. 37, pp. 28–31, Apr. 2020.

[24] K. Liang and J. Mao, “Dynamic path planning of robot based on improved ant colony algorithm,” Electron. Meas. Technol., vol. 43, pp. 56–60, Apr. 2020.

[25] Y. Zhang, J. Song, and Q. Zhang, “Local path planning of outdoor cleaning robot based on an improved DWA,” Robotics, vol. 42, pp. 617–625, Sep. 2020.

[26] Z. Wang, G. Li, and J. Ren, “Dynamic path planning for unmanned surface vehicle in complex offshore areas based on hybrid algorithm,” Comput. Commun., vol. 166, pp. 49–56, Jan. 2021.

[27] L. Zeng, H. Li, Y. Lin, and S. Zhang, “Study on simulation optimization of Gazebo based on asynchronous mechanism,” Comput. Sci., vol. 47, pp. 593–598, Nov. 2020.

[28] Engineering—Aerospace Engineering: New Findings from National University of Defence Science and Technology Describe Advances in Aerospace Engineering (Conflict Detection and Resolution for Civil Aviation: A Literature Survey), Defense & Aerospace Week, 2019, p. 103.

[29] J. Sun, J. Tang, and S. Lao, “Collision avoidance for cooperative UAVs with optimized artificial potential field algorithm,” IEEE Access, vol. 5, pp. 18382–18390, 2017.

QIBING JIN received the Ph.D. degree in control theory and engineering from Northeastern University, Shenyang, Liaoning, China, in 1999. He joined the Beijing University of Chemical Technology, Beijing, China, in 2002. He is currently a Full Professor with the College of Information Science and Technology, the Director of the Institute of Automation, Beijing University of Chemical Technology. He has rich experience in control engineering, and his many research results have been applied in petroleum and chemical industry. In recent years, he was awarded several Prizes for Science and Technology Progress. His main research interests include advanced control, intelligent instrument, system identification, and control theory.

CHUNING TANG received the B.S. degree from the Beijing University of Chemical Technology, Beijing, China, in 2018, where he is currently pursuing the M.A.Sc. degree in control science and engineering. His main research interests include path planning problem, including global path planning, local path planning, and dynamic path planning.

WU CAI received the B.S. degree from the Beijing University of Chemical Technology, Beijing, China, in 2014, where she is currently pursuing the Ph.D. degree with the Department of Information Science and Engineering. Her research interests include decoupling and disturbance rejection of multivariable systems.