Path planning of lunar robot based on dynamic adaptive ant colony algorithm and obstacle avoidance

Shinan Zhu¹, Weiyi Zhu¹, Xueqin Zhang¹ and Tao Cao²

Abstract
Path planning of lunar robots is the guarantee that lunar robots can complete tasks safely and accurately. Aiming at the shortest path and the least energy consumption, an adaptive potential field ant colony algorithm suitable for path planning of lunar robot is proposed to solve the problems of slow convergence speed and easy to fall into local optimum of ant colony algorithm. This algorithm combines the artificial potential field method with ant colony algorithm, introduces the inducement heuristic factor, and adjusts the state transition rule of the ant colony algorithm dynamically, so that the algorithm has higher global search ability and faster convergence speed. After getting the planned path, a dynamic obstacle avoidance strategy is designed according to the predictable and unpredictable obstacles. Especially a geometric method based on moving route is used to detect the unpredictable obstacles and realize the avoidance of dynamic obstacles. The experimental results show that the improved adaptive potential field ant colony algorithm has higher global search ability and faster convergence speed. The designed obstacle avoidance strategy can effectively judge whether there will be collision and take obstacle avoidance measures.

Keywords
Lunar robot, ant colony algorithm, artificial potential field, dynamic obstacle avoidance

Introduction
There are abundant unexploited resources on the moon. Many countries are promoting the lunar exploration project actively. The lunar robot is an indispensable part of the lunar exploration project, and the path planning is the guarantee for the lunar robot to complete its tasks safely and accurately in the lunar environment. The goal of path planning is to enable the robot to find an optimal path from the start point to the destination. Limited by the energy carried by lunar robot, seeking the shortest path is the optimal goal of the path planning algorithm.

In various traffic planning problems, route planning has been widely studied and discussed. These algorithms mainly include A* algorithm, simulated annealing algorithm, genetic algorithm, differential evolution algorithm, particle swarm optimization algorithm, ant colony algorithm, and so on. Those algorithms have been widely used in engineering
practice. A* algorithm\textsuperscript{1} determines the next path grid step-by-step by comparing eight adjacent heuristic function values of the current path grid. In the simulated annealing algorithm,\textsuperscript{2} by controlling the temperature parameter $T$ and using the Metropolis criterion, the attenuation function is used to change the temperature $t$ to minimize the objective function and determine the next target point. Genetic algorithm\textsuperscript{3} uses crossover and mutation to determine the next target point and to achieve the optimal path selection. The differential evolution algorithm based on the evolution idea of genetic algorithm\textsuperscript{4} solves the shortest path problem through hybridization, mutation, and selection. Particle swarm optimization algorithm\textsuperscript{5} is to know the next iteration position of particles using the current position, global extremum, and single extremum. Ant colony algorithm\textsuperscript{6} is a kind of bionic algorithm that simulates the way ants search for food in nature. The basic idea is to find the shortest path according to the concentration of pheromone left by ants. In these methods, with the increasing complexity of obstacles, the convergence speed of genetic algorithm, simulated annealing algorithm,\textsuperscript{7} A* algorithm,\textsuperscript{8} and differential evolution algorithm\textsuperscript{9} will slow down. The robustness of particle swarm optimization algorithm is poor slightly, and the optimal solution depends on initialization. Ant colony algorithm has advantages in positive feedback, robustness, and so on. It is easy to combine with various heuristic algorithms to speed up convergence speed and reduce resource consumption when the algorithm is running, so it has been widely used in robot path planning.

Considering the energy consumption and dynamic obstacle avoidance, the grid method is used to process the simulated map data. Then the adaptive potential field ant colony algorithm is used to do the path planning. This method adjusts the heuristic function dynamically by introducing the inducing heuristic factor, so as to improve the global optimization ability and accelerate the convergence speed. The proposed adaptive potential field ant colony algorithm is called A-APFACO. In addition, predictable and unpredictable obstacle avoidance strategies are also considered. Therefore, we propose a geometric method for detecting dynamic obstacles.

The organizational structure of the article is as follows. The second section describes the related work. The third section mainly describes the environment modeling based on grid map method and an A-APFACO. The fourth section describes the dynamic obstacle avoidance strategy. The fifth section describes the experiments and results. The sixth section summarizes the whole paper.

Related work

Ant colony algorithm has been proved as good performance in path planning. However, it still has unavoidable shortcomings, such as long search time, slow convergence speed, and falling into local optimum easily. To improve the performance of ant colony algorithm, many researchers proposed improvement schemes. Stutzle and Hoos\textsuperscript{10} proposed the maximum and minimum ant system algorithm, which introduced the definite maximum and minimum trajectories into the path. By constraining the maximum and minimum trajectories, the problem of stagnation was solved. Liu et al.\textsuperscript{11} proposed an algorithm for path planning of mobile robots. This algorithm uses grid method to simulate the environment, and in the process of global optimization, pheromone diffusion and geometric local optimization are combined to accelerate the convergence speed of ant colony algorithm. Cheng et al.\textsuperscript{12} combined simulated annealing algorithm with ant colony algorithm to speed-up pheromone updating. Long et al.\textsuperscript{13} combined the evaluation function of A* algorithm with ant colony algorithm to improve the speed of convergence and the smooth of planned path. As can be seen, the methods mentioned in the literature\textsuperscript{10–13} have improved the state transition rule and accelerated the convergence speed of ant colony algorithm, but they cannot adjust the state transition rule adaptively, which made the algorithm easily fall into local optimum. Akka and Khaber\textsuperscript{14} used new pheromone updating rule to adjust the evaporation rate dynamically, accelerated convergence speed, and expanded the search space of ant colony algorithm. Yang et al.\textsuperscript{15} proposed a double-deck ant colony optimization algorithm, which consisted of a parallel elite ant colony optimization algorithm and smoothness optimization algorithm. It was mainly used for autonomous navigation of robots. This method can generate better collision-free paths effectively and consistently. However, the method mentioned in the literature\textsuperscript{14,15} can only be applied to fixed environment, which means the distribution of obstacles is fixed. Once the environment changes, the parameters of algorithm need to be readjusted.

Dynamic obstacle avoidance problem is also a key point for robot. Many scholars have proposed solutions. Aiming at distributed robots, Javier et al.\textsuperscript{16} proposed a method of combining constrained nonlinear optimization with consistency to avoid static and dynamic obstacles. However, this method can only be applied to predictable dynamic obstacle, it is invalidation for unpredictable obstacle. Hu et al.\textsuperscript{17} used global routes to construct centerlines, generated a set of candidate paths in the $s-q$ coordinate system, and converted them into Cartesian coordinates. The optimal path was selected by considering the total energy consumption. Wang et al.\textsuperscript{18} proposed a probabilistic model for three kinds of robot dynamic obstacle avoidance based on Markov decision algorithm in uncertain dynamic environment. With PRISM software, the minimum time and energy consumption of each obstacle avoidance strategy were obtained, and the optimal obstacle avoidance strategy under different conditions was determined. However, only a single form of dynamic obstacle avoidance is considered in the studies by Hu et al.\textsuperscript{17} and Wang et al.,\textsuperscript{18} just codirectional motion is considered in the study by Hu et al.,\textsuperscript{17} while intersecting motion is considered in the study by Wang et al.\textsuperscript{18} Sun et al.\textsuperscript{19} proposed a new artificial potential
field algorithm danger index artificial potential field (DIAPF) based on hazard index, which considered the relative distance as well as the relative velocity between the robot and the moving obstacle. However, when encountering dynamic obstacle, this method needs to replan the path according to the speed of dynamic obstacle. Wang proposed an autonomous obstacle avoidance optimization algorithm. The problem of autonomous obstacle avoidance of unmanned aerial vehicle is transformed into the problem of solving quadratic equation of one variable, and the expected speed and angle are calculated to achieve obstacle avoidance. However, the premise of this method is that the robot and obstacles need to maintain a fixed speed. If the speed changes, it will lead to the failure of obstacle avoidance.

An adaptive potential field ant colony algorithm

Aiming at the problems of falling into local optimum, slow convergence, and dynamic obstacle avoidance in ant colony algorithm, firstly, we propose an improved potential field ant colony algorithm based on dynamic obstacle avoidance. Then, we improve the heuristic function in the transfer probability based on the ant colony algorithm. Finally, a dynamic transfer strategy, which enables robots to avoid obstacle in dynamic environment, is proposed. The main flowchart is as shown in Figure 1.

Environment modeling based on grid method

Considering that, the lunar environment is relatively simple and full of ring mountains and plains. The environment model is simplified to a two-dimensional (2D) grid map, and the moving path of robot is planned in map as shown in Figure 2.

Figure 1. Flowchart of A-APFACO based on dynamic obstacle avoidance strategy.

Figure 2. Grid map.

Figure 2 shows a 2D grid map with 20 rows and 20 columns, where black grid represents obstacle and white grid represents accessible space. In the 2D plane, the coordinates of each grid are calculated in the orthogonal reference coordinate XOY. Assuming that the entire grid is marked with integer and their length are equal to the unit length of coordinate, the following coordinate can be obtained as

\[
\begin{align*}
    x_i &= \text{mod}(i, N_x) \\
    y_j &= \text{mod}(j, N_y)
\end{align*}
\]

Among them, \(N_x\) denotes the number of grids of each row, \(N_y\) denotes the number of grids of each column, and mod is the remainder operation. Here, \(N_x = 20, N_y = 20\), the coordinate of point A is \(x_i = 13, y_j = 19\).

Adaptive potential field ant colony algorithm

Ant colony algorithm. Dorigo et al. proposed an ant colony algorithm (ACO) in 1996, which is a heuristic algorithm for finding the optimal solution. It simulates the foraging behavior of ants. Assume at time \(t\), the ant \(k\) chooses the next node according to formula (2)

\[
P^k_\gamma(t) = \left\{ \begin{array}{ll}
    (\tau_\gamma(t))^{\alpha} (\eta_\gamma(t))^{\beta} & \text{if } j \in \text{allow}_k \\
    0 & \text{otherwise}
\end{array} \right.
\]

Here, \(P^k_\gamma(t)\) denotes the probability of ants moving from point \(i\) to point \(j\), \(\tau_\gamma(t)\) represents the remaining pheromone from point \(i\) to the next node, \(j \in \text{allow}_k\) denotes the point that has been allowed by ant \(k\), \(\eta_\gamma(t)\) is a heuristic function, and \(\alpha\) and \(\beta\) denote the importance of pheromone and heuristic function in the state transition rule, respectively. Heuristic function is defined as

\[
\eta_\gamma(t) = \frac{1}{d_j}
\]
where \( d_{ij} \) denotes the distance from point \( i \) to point \( j \), \( d_{ij} \) is defined as
\[
d_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}
\] (4)

The updated pheromone is determined by the volatile pheromone and the pheromone left by ants. The updated formula of pheromone is as follows
\[
\tau_{ij}(t + 1) = (1 - \rho)\tau_{ij}(t) + \sum_{k=1}^{m} \Delta \tau_{ij}^k
\] (5)

Here, \( \rho \) represents the evaporation rate of pheromone, \( m \) is the number of ants, and \( \Delta \tau_{ij}(t) \) represents the pheromone left by ants under the current iteration round, which can be expressed as
\[
\Delta \tau_{ij}^k = \frac{Q}{L_k} \text{ if } L_k \text{ is the shortest path}
\] (6)

Improvement of heuristic function.

(1) Improvement of heuristic function based on artificial potential field method.

Artificial potential field method is an effective method to solve the problems of slow convergence caused by blind search in the early stage of the ant colony algorithm. Artificial potential field is a virtual force field exerting on the environment, usually consisting of gravity and repulsion. Let the gravitational function be \( G(t) \), which is defined as the relative distance between the robot and the target. The calculation formula is as follows
\[
G(t) = \frac{1}{1 + \sqrt{(x_s - x_t)^2 + (y_s - y_t)^2}}
\] (7)

Among them, \((x_t, y_t)\) is the coordinate of the current node, \((x_s, y_s)\) is the coordinate of the target point. If the current point is close to the target point, the denominator will be zero. The denominator introduces a constant of 1 to ensure computability. Here, the gravitational function is equivalent to the heuristic function \( \eta_{ij} \) in the original ant colony algorithm. Compared with the formula (3), the gravitational function as the heuristic function only needs to consider the distance between the current node and the end, instead of the location of the next node. That is, the selection of the next node can be directly obtained by the gravitational function, which can speed up the convergence rate.

The repulsion function in the artificial potential field is \( R(t) \). The repulsion function takes advantage of the position information of the obstacle. It is defined as a function between the robot and the obstacle. The calculation formula is as follows
\[
R(t) = \begin{cases} 
1 & (x_t, y_t) \text{ is not in the obstacles} \\
0 & \text{otherwise}
\end{cases} 
\] (8)

(2) Improvement of heuristic function based on adaptive inducement heuristic factor.

The heuristic function solves the problem of convergence speed for the original heuristic function in ant colony algorithm, but it is still unable to solve the problem of weak global search ability in the later stage of the algorithm. In this article, iteration is considerable as the negative feedback coefficient to make the pheromone change adaptively, which ensures the consistency between the optimization ability and the convergence speed. We define \( \lambda \) as an adaptive inducing heuristic factor
\[
\lambda = 0.7 + \log \left(1 + \frac{N_{\text{max}} - N}{N_{\text{max}}}\right)
\] (10)

Here, \( N_{\text{max}} \) and \( N \) are the maximum and current number of iterations, respectively. In the prophase, the heuristic factor is more than 1, the heuristic function is dominant, and the heuristic function is strengthened, which leads ants to move in a directional manner. In the anaphase, \( N \) becomes larger and the heuristic factor is less than 1. At this time, the heuristic function will be weak and the pheromone will play a major role in the stage, which helps ants to find a new way based on pheromone updating.

Finally, the heuristic function after introducing adaptive inducing heuristic factor is as follows
\[
\eta_{ij} = \lambda G(t)R(t)
\] (11)

A-APFACO compared with the maximum and minimum ant colony algorithm proposed in the study by Stutzle and Hoos\(^{10}\), although the maximum and minimum ant colony algorithm improves the local optimization ability to a certain extent, this article combines the artificial potential field with the ant colony algorithm, which has better convergence and global optimization characteristics. Compared with the ant colony optimization with pheromone diffusion model and geometrical optimization path method (ACO-PDG) algorithm proposed in the study by Liu et al.\(^{11}\), the convergence speed of the algorithm in Liu et al.’s work will be slower in the later stage, and the adaptive heuristic function designed in this article can further accelerate the convergence speed of the algorithm in the later stage.
Adaptive artificial potential field ant colony algorithm. The flowchart of the A-APFACO is as shown in Figure 3. The pseudocode of the adaptive artificial potential field ant colony algorithm is presented in Table 1.

**Robot dynamic obstacle avoidance strategy**

**Obstacle avoidance strategy**

Ant colony algorithm can plan to avoid known static obstacles. However, the dynamic obstacle avoidance between lunar robots should also be considered when a lunar robot cluster appears. Since the scene is set on the surface of the moon, the number of lunar robots is limited at present. Assuming that the number of robots is $n$, $1 \leq n \leq 3$, all of these robots obey the control from the same organization.

In the case of dynamic obstacle avoidance, moving obstacle can be divided into predictable obstacle and unpredictable obstacle. Since all lunar robots have path planning, and their moving speed is fixed or known, the collision point is predictable. We define this kind of obstacle avoidance problem as predictable obstacle avoidance. However, if something goes wrong while robot is moving, the obstacle avoidance problem will be unpredictable.

Defining robot $R$ has a set of traveling attributes \{direction, action, velocity, priority, interaction\}, in which direction is the moving direction of robot, direction = \{forward, backward, left, right, left up, left down, right up, right down\}. The action is the motion state of the robot, action = \{move, stop\}. The velocity is the speed of the robot, usually it is a known quantity. The priority is the priority of the robot. According to the number of robots, priority = \{1, 2, 3\}. The collision state of robots can be defined as interaction = \{opposite motion, codirectional motion, intersectional motion\}.

**Predictable dynamic obstacle avoidance strategy.**

1. Minimum safe distance.

For multi-robot working environment, collision avoidance rule should be given to enable robots to work cooperatively and improve work efficiency. To avoid collision, the distance between two lunar robots should be greater than the minimum safety distance.

---

**Figure 3.** Flowchart of ant colony algorithm.

**Table 1.** Algorithm of A-APFACO.

| Algorithm 1. A-APFACO |
|-----------------------|
| **Input:** map information: $(X_{\text{start}}, Y_{\text{start}}), (X_{\text{end}}, Y_{\text{end}})$, can_go_list, taboo_list |
| **Initialize:** Ant number $m$, Maximum number of iterations $N_{\text{max}}$, Pheromone matrix $\eta$ |
| **Output:** $l_{\text{shortest}}$ |
| for iterations < $m$ do |
| for each ant $k$ do |
| if grid $i \in allow_k$: |
| if grid $i \in \text{can\_go\_list}$: |
| Choose the next grid $j$ according to formulas 1, 7, 8, 10 and 11 |
| Remove $j$ in can_go_list |
| Put $j$ in taboo_list |
| Update pheromone matrix according to formula 4 and 5 |
| end |
| end |
| end |
| end |
Assuming that the two robots are \( R_1 \) and \( R_2 \), and their current coordinates on the 2D grid map are \((x_1, y_1)\) and \((x_2, y_2)\). The collision avoidance strategy will be initiated when the distance \( d \) between the two robots is less than the minimum safe distance \( h \).\(^{21}\)

\[
d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} < h \quad (12)
\]

2. Obstacle avoidance strategy.

To avoid obstacles, priority-based alternative traffic strategy is adopted in this article.

Let two robots be \( R_1 \) and \( R_2 \) respectively, because obstacles are predictable and their trajectories are known. The obstacle avoidance strategies are as follows:

i. Strategy 1:

If interaction = opposite motion, \( R_1 \) move up and down and \( R_2 \) move up and down on the 2D grid map, \( R_1 \).action = move, \( R_2 \).action = move.

At time \( t \), the distance \( d \) between \( R_1 \) and \( R_2 \) is less than \( h \). If \( R_1 \).priority > \( R_2 \).priority, \( R_1 \) judges whether there are obstacles on its left or right side based on the map information.

If bilateral or unilateral obstacle-free, at moment \( t + 1 \), \( R_2 \) conducts lateral avoidance. \( R_2 \).direction = left or right. Then \( R_2 \).action = stop until \( R_1 \) passes through. Then \( R_2 \).action = move, \( R_2 \) returns to the original route and continues to drive.

If there are obstacles on both sides, at moment \( t + 1 \), \( R_2 \).action = move, \( R_2 \).direction = backward, until \( R_2 \) retreating to the barrier-free area, \( R_2 \).direction = left or right, \( R_2 \) executes lateral avoidance, \( R_2 \).action = stop, until \( R_1 \) passes through, \( R_2 \).action = move, \( R_2 \) returns to the original route and continues to drive.

ii. Strategy 2:

If interaction = codirection, \( R_1 \) and \( R_2 \) move up and down from the 2D grid map, \( R_1 \).action = move, \( R_2 \).action = move.

If \( R_1 \) is in front of \( R_2 \), \( R_1 \).velocity > \( R_2 \).velocity, no evasion is required. \( R_1 \) moves ahead of \( R_2 \).

If \( R_1 \) is in front of \( R_2 \), \( R_1 \).velocity < \( R_2 \).velocity, \( R_1 \) executes lateral avoidance.

Firstly, \( R_2 \).action = stop. Secondly, \( R_1 \) judges whether there are obstacles around it. If there is no obstacle, \( R_1 \).direction = left or right, and then \( R_1 \).action = stop, \( R_1 \) executes lateral avoidance. \( R_2 \) restarts to move, \( R_1 \).action = move. Finally, \( R_1 \) restart to move until \( R_2 \) pass through. If there are obstacles, \( R_1 \).direction = forward, \( R_1 \) drives to obstacle-free point, then executes lateral avoidance until \( R_2 \) passes through, \( R_1 \).action = motion, \( R_1 \) returns to the original route and continue to move.

iii. Strategy 3:

Intersectional motion can be regarded as an extension of the opposite motion. By setting priority, obstacle avoidance strategy can be applied to multi-robots.

\[
\text{Figure 4. Collision detection model. (a) is based on \( R_1 \) and (b) is based on the ground.}
\]

If interaction = intersectional motion, \( R_1 \).velocity > \( R_2 \).velocity, according to strategy 1, let \( R_2 \) stop moving at collision point until \( R_1 \) passes through.

Unpredictable obstacle avoidance strategy. When robots moving on the moon, the most complex situation is obstacles are unpredictable. This article presented a geometric method based on moving route of robot to detect the location of the unpredictable obstacle. The method mainly calculates the distance between two objects. If the distance between two robots is less than the minimum safe distance, it is considered that there will be a collision. The geometric model is shown in Figure 4. Figure 4(a) is based on \( R_1 \) and Figure 4(b) is based on the ground.

In Figure 4(a), let robot \( R_1 \) be a reference. \( R_2 \) moves relative to \( R_1 \), and the arrow represents the direction of the robot’s movement. Then there exists a minimum distance \( L_{\text{min}} \) between two robots along the motion route at some moment.\(^{22}\)

According to the geometric relationship in Figure 4(a), it can be deduced that the minimum distance between two robots is

\[
L_{\text{min}} = \frac{L_1L_2\sin(\theta_1 - \theta_2)}{\sqrt{L_1^2 + L_2^2 - 2L_1L_2\cos(\theta_1 - \theta_2)}} \quad (13)
\]

In Figure 4(b), the lunar surface is taken as the reference. Suppose that the robot \( R_1 \) locates at point \((x_1, y_1)\) at \( t - 1 \) moment, point \( B \) \((x_2, y_2)\) at \( t \) moment, robot \( R_2 \) locates at point \( C \) \((x_3, y_3)\) at \( t - 1 \) moment, and point \( D \) \((x_4, y_4)\) at \( t \) moment. The \( L_1, L_2, L_3, L_4, L_5, \) and \( L_6 \) are distances of \( A-C, B-D, B-C, A-B, A-D, \) and \( C-D \), respectively. Among them

\[
L_1 = \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2} \quad (14)
\]

The calculation formulas of \( L_2, L_3, L_4, L_5, \) and \( L_6 \) are similar with \( L_1 \) and will not be described again.

\( \theta_1, \theta_2 \) are the azimuth angles of \( R_1 \) at \( t \) and \( t - 1 \) moment, respectively. The calculation formula is as follows

\[
\theta_1 = \arccos\left(\frac{L_2^2 - L_3^2 - L_1^2}{2L_4L_1}\right) \quad (15)
\]
\[ \theta_2 = \arccos \left( -\frac{L_x^2 - L_y^2 - L_z^2}{2L_xL_y} \right) \]  

If it is predicted that \( R_1 \) will collide with \( R_2 \) at \( t \) moment, \( R_1 \) will be stopped. The minimum distance is changed to the distance \( L'_{\text{min}} \) between B and E at this time. The definition of \( L'_{\text{min}} \) is as follows:

\[ L'_{\text{min}} = L_2 \sin \left( \arccos \left( -\frac{L_x^2 - L_y^2 - L_z^2}{2L_xL_y} \right) \right) \]

Assuming that the lunar robot always travels along a straight line (non-curve), the moving direction of \( R_1 \) is \( \text{direction}_{R_1} = (x_2 - x_1, y_2 - y_1) \), that of \( R_2 \) is \( \text{direction}_{R_2} = (x_4 - x_3, y_4 - y_3) \). If \( \text{direction}_{R_1} = K \text{ direction}_{R_2} \), \( K \) is a constant, which means the relative movement of robots is opposite or codirectional. If \( K > 0 \), it is codirectional movement, on the contrary, it is opposite movement. If \( \text{direction}_{R_1} \neq K \text{ direction}_{R_2} \), then the relative movement is intersection movement. When codirectional or cross-directional motion occurs, the robots execute strategy 1 and 2 according to predictable obstacle avoidance method, and when cross-directional motion occurs, the robots execute strategy 3 according to predictable obstacle avoidance method.

In this method, the distance and azimuth angle are measured by binocular camera, radar, or other equipment on the lunar robot and can also be estimated by the position of the robot in the grid map.

### Experimental results and analysis

#### Experimental environment and evaluation indicators

In the following experiments, the ant colony algorithm is implemented in a 20 \( \times \) 20 2D grid map. To verify the effectiveness of the proposed algorithm, the proposed algorithm is compared with the initial ant colony algorithm, the improved algorithm in the study by Liu et al., the genetic algorithm, and the A* algorithm under the same environment.

The main evaluation index of the experiment is the parameter evaluation degree. It is a comprehensive evaluation criterion under which the algorithm can obtain both a higher success rate of the shortest path and fewer average iterations.

\[ \text{Evaluate} = \frac{S(x)}{N} \]

Among them, \( N \) represents the average number of iterations and \( S(x) \) is the shortest path success rate when the parameter value is \( x \).

In this article, experiment platform is Python 3.7. The computer operating system is Windows 10, the CPU is Core i7-8750H, and the memory is 8.00 GB.

#### Ant colony algorithm experiment

**Experiment 1: Parameter choice.** This experiment is used to seek the appropriate pheromone parameter \( \alpha \), heuristic information parameter \( \beta \), pheromone evaporation parameter \( \rho \) to make the algorithm obtain high success rate, and fewer number of iterations. The obstacle environment is shown in Figure 5, the number of ants is 20, the number of iterations is 100, the start point is (5, 18), and the end point is (19, 0). The initial parameters \( \alpha = 1.0, \beta = 18, \) and \( \rho = 0.5 \) are specified. The experimental results are presented in Tables 2 to 4. When the algorithm runs 50 times, the proportion of the shortest path is planned.

In the table, the number of iterations represents the average number of iteration. It can be seen that the maximum parameter evaluation is achieved when the parameters are \( \alpha = 1.0, \beta = 27, \) and \( \rho = 0.2 \), respectively. With these parameters, the convergence curve of the algorithm is shown in Figure 6. The convergence curve is stable, and the algorithm converges at the 12th iteration. In addition, the success rate of the shortest path is 96%.

**Experiment 2: Scene simulation.** To verify the effectiveness of the proposed algorithm, this experiment compares other path planning algorithms in four different typical obstacle scenes. These scenarios are shown in Figure 7, four kinds of obstacle environments.

They separately express the location of obstacles locate near the starting point, near the terminating point, in the middle of path with simple distribution, and in the middle of path with complex distribution. Assuming that the start
Adaptive potential field ant colony algorithm, inertia weight is 0.9, and learning factor probability is 0.8, mutation probability is 0.2 in GA algorithm.

1.1, runs 100 times. The planned shortest paths are shown with point is (0,0) and the end point is (19,19), the experiment gets all the shortest path in the four scenes. As a result, compared to other algorithms, our A-APFACO algorithm has better performance of path planning in the four obstacle scenes.

The convergence curves of traditional ACO, ACO-PDG, A-APFACO algorithms in complex scenes are shown in Figure 8. In scene (c), the convergence iterations of the three algorithms are 7, 11 and 8 respectively with similar performance. In the scene (d), only A-APFACO can plan a better path, and the algorithm converges only after four iterations. Experimental results show that the proposed method has better robustness for complex scene.

Dynamic obstacle avoidance experiment

This experiment is used to verify the effectiveness of the strategy for predictable dynamic obstacle avoidance. In the experiment, a 20×20 grid map was used with a moving speed of one step per second.

Experiment 3: Predictable obstacle avoidance of two targets.

1. Opposite motion.

This experiment verifies the strategy of obstacle avoidance when two targets collide in opposite motion. In Figure 9, the red “*” represents R1, and the blue “•” is R2. △ represents the collision point.

The starting point of robot R1 is (19, 10) and the ending point is (1, 10). The starting point of robot R2 is (1, 10) and the ending point is (19, 10). The priority of R1 is higher than that of R2. The motion path planned by ant colony algorithm of R1 is 7→6→5→3→2, and that of R2 is 1→3→5→6→7→3. It can be seen in Figure 9, there will be a collision in opposite directions at the position of node 5, and there are obstacles beside the collision point.

Implement obstacle avoidance strategy 1, the motion path of R1 is 7→6→5→3→2, and that of R2 is 1→3→5→4→4→5→6→7, as shown in Figure 9(b). Here the second “4” in the path of R2 represents that R2 stopped a step at node 4. In the later experiments, if there are many identical nodes in the traveling path of robot, it means the same thing, and no further explanation is given.

2. Codirectional motion.

This experiment verifies the strategy of obstacle avoidance when two targets collide in codirectional motion. In Figures 4 to 9, the red “*” is R1, and the blue “•” is R2, △ represents the collision point.

The starting point of robot R1 is (19, 10) and the ending point is (0, 10). The starting point of robot R2 is (13, 10) and the ending point is (0, 10). The priority of R1 is higher than that of R2. The speed of R1 is two steps per second and R2 is one step per second. The motion path planned by ant colony algorithm of R1 is 1→3→5→7→9, and that of R2 is 3→4→5→6→7→8→9. It can be seen from Figure 10 that there will be a collision in codirection at the position of node 5.

Table 3. Control parameters α and β, the success rate in shortest path and average iteration number evaluation with change of parameter β.

| Parameter β | Success rate (%) | Iteration | Evaluate (β) |
|-------------|------------------|-----------|--------------|
| 24          | 90               | 34        | 2.687        |
| 25          | 82               | 21        | 3.868        |
| 26          | 80               | 51        | 1.575        |
| 27          | 94               | 24        | 4.017        |
| 28          | 86               | 30        | 2.848        |

Table 4. Control parameters α and β, the success rate in shortest path and average iteration number evaluation with change of parameter ρ.

| Parameter ρ | Success rate (%) | Iterations | Evaluate (ρ) |
|-------------|------------------|------------|--------------|
| 0.1         | 88               | 74         | 1.193        |
| 0.2         | 90               | 16         | 5.745        |
| 0.3         | 84               | 41         | 2.049        |
| 0.4         | 86               | 59         | 1.470        |
| 0.5         | 76               | 37         | 2.043        |

Figure 6. Curve of convergence with selected parameter.
Implement obstacle avoidance strategy 1, the motion path of \( R_1 \) is \( 7 \rightarrow 6 \rightarrow 5 \rightarrow 3 \rightarrow 2 \), and that of \( R_2 \) is \( 1 \rightarrow 3 \rightarrow 5 \rightarrow 4 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \), as shown in Figure 10(b).

3. Intersectional motion.

This experiment verifies the strategy of obstacle avoidance when two targets collide in intersection motion. In Figure 11, the red “*” represents \( R_1 \), and the blue “o” is \( R_2 \). A represents the collision point.

The starting point of robot \( R_1 \) is \((0, 14)\) and the ending point is \((9, 7)\). The starting point of robot \( R_2 \) is \((0, 6)\) and the ending point is \((9, 15)\). The priority of \( R_1 \) is higher than that of \( R_2 \). It can be seen in Figure 11, according to the route planned by ant colony algorithm, there will be a collision at the position of node 10.

Implement obstacle avoidance strategy 3, \( R_2 \) with low priority stops two steps at node 4 until \( R_1 \) passes and \( R_2 \) continues to move forward. The motion path of \( R_1 \) is \( 6 \rightarrow 7 \rightarrow 8 \rightarrow 9 \rightarrow 10 \rightarrow 11 \rightarrow 12 \), and that of \( R_2 \) is \( 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 4 \rightarrow 10 \), as shown in Figure 11(b).

**Experiment 4: Comprehensive movement of three targets.** This experiment verifies the strategy of obstacle avoidance when three targets collide. In Figure 12, the green “+” represents \( R_1 \), the blue “o” is \( R_2 \) and the red “*” represents \( R_3 \). A represents the collision point.

The starting points of robot \( R_1 \), \( R_2 \), and \( R_3 \) are \((19, 0)\), \((19, 1)\), and \((19, 10)\), respectively, and the ending points of robot \( R_1 \), \( R_2 \), and \( R_3 \) are \((0, 19)\), \((6, 14)\), and \((0, 10)\).

The motion path planned by ant colony algorithm of \( R_1 \) is \( 9 \rightarrow 10 \rightarrow 11 \rightarrow 4 \rightarrow 12 \rightarrow 13 \rightarrow 14 \), and that of \( R_2 \) is \( 7 \rightarrow 6 \rightarrow 5 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 1 \), \( R_3 \) is \( 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \). It can be seen in Figure 12, there will be a collision at the position of node 4.

As can be seen from Figure 12, at node 4, \( R_1 \) will intersect with \( R_3 \) and \( R_1 \), while \( R_2 \) and \( R_1 \) are in opposite motion. If the priority is \( R_2 > R_3 > R_1 \), \( R_1 \) adopts the stop obstacle avoidance strategy in the safe range, while \( R_3 \) with the second priority avoids at node 8. After robot \( R_2 \), which has the highest priority, moves out of the safe range, \( R_3 \) and \( R_1 \) move in turn according to the original planning path. The process of obstacle avoidance is completed.

Implement obstacle avoidance strategy 1 and 3, the motion path of \( R_1 \) is \( 9 \rightarrow 10 \rightarrow 11 \rightarrow 4 \rightarrow 12 \rightarrow 13 \rightarrow 14 \), and that of \( R_2 \) is \( 7 \rightarrow 6 \rightarrow 5 \rightarrow 4 \rightarrow 8 \rightarrow 8 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 1 \), \( R_3 \) is \( 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \).

**Experiment 5: Dynamic avoidance of unpredictable obstacles.** This experiment is designed to verify the strategy of dynamic avoidance of unpredictable obstacles. In Figures 4 to 9, the red “*” represents \( R_1 \), and the blue “o” is \( R_2 \). A represents the collision point. Suppose the robot \( R_1 \) cannot predict where it will collide with \( R_2 \). And the priority of \( R_2 \) is higher than that of \( R_1 \).

In Figure 13, the distance and angle are calculated based on the grid in the map. Assuming the detection radius \( r \) of the robot \( R_1 \) is 8, the safe distance \( h = 1.5 \). At \( t + 1 \) moment, when the robot \( R_1 \) reaches point \( A \), it detects that \( R_2 \) reaches point \( C \). At the next moment, \( R_1 \) moves to point \( B \) and \( R_2 \) moves to point \( D \). According to formula (13), \( L_{\text{min}} \approx 0 \), \( L_{\text{min}} \) is less than the safe distance, which means that the two robots will collide at some point. At this time, robot \( R_1 \) stops at point \( B \), \( R_2 \) continues to move to point \( E \). Here, according to formula (17), \( L'_{\text{min}} \) was calculated again, \( L'_{\text{min}} = 1.41 \times h \) so \( R_1 \) keeps stopping at point \( B \) until \( R_2 \) passes through. Then \( R_1 \) moves following the original route.

After completing the obstacle avoidance, the motion path of \( R_1 \) is \( A \rightarrow B \rightarrow B \rightarrow B \rightarrow E \).
Table 6 presents the shortest distance predicted by $R_2$. Because $R_1$ stops avoiding at point B, the azimuth angle $\theta_1$ between $R_2$ and $R_1$ is null.

**Comparative experiment of dynamic obstacle avoidance**

In this section, the effectiveness of the proposed method is further verified by comparing the unpredictable dynamic obstacle avoidance strategy with the dynamic obstacle avoidance strategy in the literature.18,20.

Figure 14 shows the 2D grid environment map used in the comparative experiment. The rectangular box in the figure is the area involved in the subsequent obstacle avoidance experiment.

**Experiment 6: Dynamic obstacle avoidance of robot with constant velocity.** As shown in Figure 14, the red "*" represents $R_1$, and the blue "●" is $R_2$, $\Delta$ represents the collision point. The speed
of $R_1$ is one step per second, $R_2$ is two steps per second, and the speed remains constant. The detection radius of $R_2$ is 8, and the priority of $R_1$ is higher than that of $R_2$.

Figure 15(a) shows the robot’s path map formed according to our obstacle avoidance strategy. Within the detection radius of the robot, the collision will be judged according to formula (13) for each position the machine travels. When $R_1$ moves to position 3 and $R_2$ moves to position 11, $R_2$ calculates $L_{\text{min}} \approx 0$. And the two robots will collide at a point in front. At this time, $R_1$ stops at position 11 and starts obstacle avoidance. Finally, the motion path of $R_1$ is 1 → 2 → 3 → 4 → 5 → 6 → 7 → 8, the motion path of $R_2$ is 9 → 10 → 11 → 11 → 12 → 5 → 13 → 14 → 15. $R_2$ avoid obstacle successfully.

Figure 15(b) is a path map of the robot formed by adopting the obstacle avoidance strategy in the study by Wang et al.\textsuperscript{18} Because the hypothesis of the strategy is that the dynamic collision point is known, the algorithm cannot calculate the collision point for the obstacles suddenly appearing in the path, and the obstacle avoidance fails. $R_1$ and $R_2$ collide at position 5. The motion path of $R_1$ is 1 → 2 → 3 → 4 → 5, and that of $R_2$ is 6 → 7 → 8 → 9 → 5.

Figure 15(c) shows the path map of the robot formed by the obstacle avoidance strategy in the study by Wang.\textsuperscript{20} Since the speed of the robot remains the same, according to the obstacle avoidance algorithm, we can calculate that collision will occur, start the obstacle avoidance strategy, $R_2$ accelerates the movement, and the speed will return to...
the original after the obstacle avoidance is successful. The motion path of \( R_1 \) is \( 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 8 \), and that of \( R_2 \) is \( 9 \rightarrow 10 \rightarrow 11 \rightarrow 5 \rightarrow 12 \rightarrow 13 \).

**Experiment 7: Dynamic obstacle avoidance in case of sudden change of robot speed.** As shown in Figure 16, the red “*” represents robot \( R_1 \), and the blue “•” represents robot \( R_2 \), "Δ" represents robot \( R_2 \)
is the collision point. The initial velocity of $R_1$ is two steps per second, $R_2$ is one step per second. The detection radius of $R_2$ is 8, and the priority of $R_1$ is higher than that of $R_2$. When $R_1$ is in position 3, the speed changes suddenly for some reason, and $R_1$ speed changes to one step per second.

Figure 16(a) is the path map of the robot formed according to our obstacle avoidance strategy. Within the detection radius of robot $R_2$, the collision will be judged according to formula (13) for each position the machine travels. When $R_2$ is in position 8, $R_1$ is in position 1 and enters $R_2$ detection range. Then, $R_2$ moves to position 9, $R_1$ moves to position 2, and the speed suddenly changes to one step per second. Then, when $R_1$ is at 4, $R_2$ is at 12 and $R_2$ is calculated to be $L_{\text{min}} \approx 0$, the collision will occur at a later position. $R_2$ starts obstacle avoidance strategy 3, stops two steps at position 12 and continues to move. The movement path of $R_1$ was $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7$, and that of $R_2$ was $8 \rightarrow 9 \rightarrow 10 \rightarrow 10 \rightarrow 4 \rightarrow 11 \rightarrow 12 \rightarrow 13$. $R_2$ avoided obstacles successfully.

Figure 16(b) is a path map of the robot formed by adopting the obstacle avoidance strategy in the study by Wang et al. The assumption of the strategy is that the collision point is known, the algorithm cannot calculate the collision point for the obstacles that suddenly appear in the path and the speed changes. The obstacle avoidance fails, $R_1$ and $R_2$ collide at position 4. The motion path of $R_1$ is $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$, and that of $R_2$ is $5 \rightarrow 6 \rightarrow 7 \rightarrow 4$.

Figure 16(c) is the path map of the robot formed according to the obstacle avoidance strategy in the study by Wang. Because of the sudden change of $R_1$’s speed in position 3, and the change of moving obstacle’s speed is not considered in the obstacle avoidance strategy. It is impossible to use acceleration or deceleration strategy to avoid obstacles. The obstacle avoidance fails, and $R_1$ and $R_2$ collide in position 4.

**Conclusion**

In this article, for the issue of lunar robot path planning, a heuristic function of ant colony algorithm is improved by adjusting the heuristic function dynamically. By adding adaptive parameters, the success rate of shortest path planned by ant colony algorithm is improved, while the convergence speed is reduced in complex environments. For the predictable and unpredictable dynamic collision problems in the moving route of lunar robot, the
corresponding obstacle avoidance strategies are proposed. Especially for detecting unpredictable dynamic obstacles, a collision prediction model based on geometric estimation is proposed, which can avoid collision between robots. The simulation results show the effectiveness of the proposed method. The proposed method enables the lunar robot to travel in the energy-saving path, that is, the shortest path, and effectively avoids collisions with each other.

Declaration of conflicting interests
The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding
The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by Shanghai Aerospace Science and Technology Innovation Fund (SAST2018-086).

ORCID iD
Shinan Zhu  https://orcid.org/0000-0003-2822-1772

References
1. Ammar A, Bennaceur H, Chaâri I, et al. Relaxed Dijkstra and A* with linear complexity for robot path planning problems in large-scale grid environments. Soft Comput 2016; 20(10): 4149–4171.
2. Hayat S and Kausar Z. Mobile robot path planning for circular shaped obstacles using simulated annealing. In: 2015 international conference on control, automation and robotics, Singapore, 20–22 May 2015, pp. 69–73. IEEE.
3. Patle BK, Parhi DRK, Jagadeesh A, et al. Matrix-binary codes based genetic algorithm for path planning of mobile robot. Comput Electr Eng 2018; 67: 708–728.
4. Storn R and Price K. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 1997; 11(4): 341–359.
5. Shao S, Peng Y, He C, et al. Efficient path planning for UAV formation via comprehensively improved particle swarm optimization. ISA Trans 2020; 97: 415–430.
6. Dorigo M, Maniezzo V, and Colomi A. Ant system: optimization by a colony of cooperating agents. IEEE Trans Syst Man Cybern B Cybern 1996; 26(1): 29.
7. Wang X and Shi P. Comparative analysis of ant colony algorithm with simulated annealing and genetic algorithm. Wirel Interconnect Technol 2015; (13): 126–127.
8. Garip ZB, Karayel D, Ozkan SS, et al. Path planning for multiple mobile robots using A* algorithm. Acta Physica Polonica 2017; 132(3): 685–688.
9. Wang X, Xu L, Yang F, et al. Optimization of dynamic shared bike scheduling based on ant colony. J Nanchang Inst Eng 2019; 38(3): 71–76.
10. Stutzle T and Hoos H. The MAX-MIN ant system and local search for the traveling salesman problem. In: Proceedings of IEEE international conference on evolutionary computation, Indianapolis, IN, USA, 13–16 April 1997, pp. 309–314.
11. Liu J, Yang J, Liu H, et al. An improved ant colony algorithm for robot path planning. Soft Comput 2017; 21(19): 5829–5839.
12. Cheng J, Miao Z, Li B, et al. An improved ACO algorithm for mobile robot path planning. In: 2016 IEEE international conference on information and automation (ICIA), Ningbo, China, 1–3 August 2016, pp. 963–968. IEEE.
13. Long S, Gong D, Dai X, et al. Mobile robot path planning based on ant colony algorithm with A* heuristic method. Front Neurorobotics 2019; 13: 15.
14. Akka K and Khaber F. Mobile robot path planning using an improved ant colony optimization. Int J Adv Robot Syst 2018; 15(3): 1–7.
15. Yang H, Qi J, Miao Y, et al. A new robot navigation algorithm based on a double-layer ant algorithm and trajectory optimization. IEEE Trans Ind Electron 2018; 66: 8557–8566.
16. Alonso-Mora J, Montijano E, Nägeli T, et al. Distributed multi-robot formation control in dynamic environments. Auton Robot 2018; 43: 1079–1100.
17. Hu X, Chen L, Tang B, et al. Dynamic path planning for autonomous driving on various roads with avoidance of static and moving obstacles. Mech Syst Signal Process 2018; 100: 482–500.
18. Wang R, Wang M, Guan Y, et al. Modeling and analysis of the obstacle-avoidance strategies for a mobile robot in a dynamic environment. Math Probl Eng 2015; 2015: Article ID 837259.
19. Sun J, Liu G, Tian G, et al. Smart obstacle avoidance using a danger index for a dynamic environment. Appl Sci 2019; 9(8): 1589.
20. Wang M. The optimization study on avoidance algorithm for UAV autonomous obstacle avoidance flight. Network New Media Technol 2019; 8(04): 36–39 + 51.
21. Wang X, Yang L, Zhang Y, et al. Robot path planning based on improved ant colony algorithm with potential field heuristic. Control Decision Making 2018; 33(10): 50–56.
22. Jing-Gang XU, Hong-Lun W, Chang L, et al. Modeling of dynamic collision avoidance zone for UAVS. Electron Opt Control 2014; 21(12): 30–35.