Path Planning for Mobile Robot in 3D Environment Based on Ant Colony Algorithm

Fan Wenrui*
Aerospace Engineering, Beijing Institute of Technology, Beijing, China, 100089
*Corresponding author: aky1_liujian@163.com

Abstract. This paper introduces the ant colony algorithm to the path planning of mobile robots, and explores the optimal path solution. The definition of path planning was studied firstly. Then modelling method of path planning was discussed. Also, current research status of three-dimensional path planning at home and abroad he existing problems of 3D path planning was analyzed. These basic theoretical knowledge have laid the foundation for the research work of this paper, and we then modelled the three-dimensional terrain environment, and used MATLAB software for simulation experiments to implement the ant colony algorithm to plan the robot's three-dimensional terrain path, and finally performed fuzzy reinforcement learning method based on ant colony algorithm to find the optimal path for the mobile robot.

Keywords: ant colony algorithm, path planning, reinforcement learning

1. Principle and implementation of ant colony algorithm

The algorithm flow of the three-dimensional path search algorithm based on ant colony algorithm is shown in Figure 1.

The ant colony algorithm uses pheromones to attract ants to search for food. Position setting and updating methods of the pheromone play a very important role for utilizing the ant colony algorithm to successfully search for food. The local pheromone update is synchronized with the search, and the pheromone update formula is:

$$\tau_{ijk} = (1 - \zeta)\tau_{ijk}$$

(1-1)

Where $\tau_{ijk}$ is the pheromone value carried on the node (i, j, k) in the three-dimensional space, and $\zeta$ is the pheromone attenuation coefficient.

Global update means that when the ants complete a path search, the path is used as a comparison object, and other paths are selected for comparison, and finally the shortest path is selected for foraging. Then the pheromone of the optimal path node will be updated. The pheromone update formula is as follows

$$\tau_{ijk} = (1 - \rho)\tau_{ijk} + \rho \Delta \tau_{ijk}$$

(1-2)

$$\tau_{ijk} = \frac{K}{\min\{length(m)\}}$$

(1-3)
Where length(m) is the path length passed by the \( m \)th ant, \( \rho \) is the pheromone update coefficient, and \( K \) is the coefficient.

\[
H(i, j, k) = (1 - \omega_1 D(i, j, k)^{\omega_2} S(i, j, k)^{\omega_3} Q(i, j, k)^{\omega_4})^{\omega_5}
\]

In the above formula, \( D(i, j, k) \) is the distance between two points, it will make the ants pick the one with the shortest distance between the two points, and \( S(i, j, k) \) is the value that can be reached in the algorithm. When the selected node is a node that is not reachable by the ants, the value of this point is 0, \( S \) will prompt the ants to select the next reachable node. \( Q(i, j, k) \) is the path length between the next node to the target point which will prompt the ants to choose a point closer to the target. \( \omega_1, \omega_2, \omega_3 \) are coefficients, representing the importance of the three factors \( D, S, \) and \( Q \), respectively. The calculation formula of \( D(i, j, k) \) is as follows:

\[
D(i, j, k) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}
\]

Where \( a \) is the current node and \( b \) is the next node.

The calculation formula of \( S(i, j, k) \) is as follows:

\[
S(i, j, k) = \frac{\text{Num} - \text{UNum}}{\text{Num}}
\]

where \( \text{Num} \) is the number of nodes that can be seen when the ants are at the node \((i, j, k)\), and \( \text{UNum} \) is the number of nodes that the ants cannot reach.

The calculation formula of \( Q(i, j, k) \) is as follows:

\[
Q(i, j, k) = \sqrt{(x_i - x_d)^2 + (y_i - y_d)^2 + (z_i - z_d)^2}
\]

**Figure 1** Three-dimensional path planning algorithm

When the ant moves from the current node to the next node, the selection probability of each node in the visible area can be calculated according to the heuristic function. The heuristic function is calculated as

\[
H(i, j, k) = D(i, j, k)^{\omega_2} S(i, j, k)^{\omega_3} Q(i, j, k)^{\omega_4}
\]

In the above formula, \( D(i, j, k) \) is the distance between two points, it will make the ants pick the one with the shortest distance between the two points, and \( S(i, j, k) \) is the value that can be reached in the algorithm. When the selected node is a node that is not reachable by the ants, the value of this point is 0, \( S \) will prompt the ants to select the next reachable node. \( Q(i, j, k) \) is the path length between the next node to the target point which will prompt the ants to choose a point closer to the target. \( \omega_1, \omega_2, \omega_3 \) are coefficients, representing the importance of the three factors \( D, S, \) and \( Q \), respectively. The calculation formula of \( D(i, j, k) \) is as follows:

\[
D(i, j, k) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}
\]

Where \( a \) is the current node and \( b \) is the next node.

The calculation formula of \( S(i, j, k) \) is as follows:

\[
S(i, j, k) = \frac{\text{Num} - \text{UNum}}{\text{Num}}
\]

where \( \text{Num} \) is the number of nodes that can be seen when the ants are at the node \((i, j, k)\), and \( \text{UNum} \) is the number of nodes that the ants cannot reach.

The calculation formula of \( Q(i, j, k) \) is as follows:

\[
Q(i, j, k) = \sqrt{(x_i - x_d)^2 + (y_i - y_d)^2 + (z_i - z_d)^2}
\]
Where b represents the next node and d represents the target node. The steps for the ants at current node \( \rho_i \) on the plane \( \pi_i \) to select the next point \( \rho_{i+1} \) on the plane \( \pi_{i+1} \) are as follows:

1) Determine the set of feasible points in the plane \( \pi_{i+1} \) according to the abstract environment.

2) According to the heuristic function, the heuristic information value of feasible points in plane \( \pi_{i+1} \) from point \( \rho_i \) is calculated in sequence.

3) Randomly select a node \( (i+1, u, v) \) in the plane \( \pi_{i+1} \) and calculate the selection probability \( p(i+1, u, v) \) of this node:

\[
P(i+1, u, v) = \frac{\tau_{\rho_{i+1, u, v}} H_{\rho_{i+1, u, v}}}{\sum \tau_{\rho_{i+1, u, v}} H_{\rho_{i+1, u, v}}} \quad \text{(1-8)}
\]

where, \( \tau_{\rho_{i+1, u, v}} \) represents the magnitude of the pheromone value at this point on the plane \( \pi_{i+1} \).

4) Finally, the roulette method is used to select the nodes in the plane \( \pi_{i+1} \), which have different probability of selecting the area that each node can reach in the visible space.

![Flow chart of ant colony algorithm](image)

**Figure 2** Flow chart of ant colony algorithm

2. 3D Path Planning on Rough Terrain for Mobile Robots

2.1 3D Modeling of Terrains

First of all, a mountain topographic map must be constructed. As shown in Figure 3, a relatively complex mountain topographic structure map (21kmx21kmx2km) is directly constructed using MATLAB.
Then construct a cubic region, and put the three-dimensional terrain space inside the cubic region completely. The cubic region is then called the planning space, as shown in Figure 4.

The grid method is used to model the planning space. First, the cubic region mentioned above is modelled in a three-dimensional coordinate system and represented by letters ABCD-EFGH. Here AB is in the OX axis direction, AE is in the OY axis direction, and AD is in the OZ axis direction. Then divide the space according to the grid division method to get an abstract environment model that the robot can understand and recognize. Assuming that the lengths of the AB and AD sides are 1, and the length of the AE side is $h$, first divide $n+1$ equal parts of the three-dimensional planning space ABCD-EFGH along the side OS, and then construct planes are parallel to the ABCD plane passing each equal point. Hence, $n$ planes $\pi_i$ ($i=0, 1, 2, \ldots, n$) are obtained. Finally, for each plane, equal divisions into $m$ pieces are performed along the X-axis direction of the side, and equal divisions into $m$ pieces are also performed along the Z-axis direction of the side to discretize the planes into $m \times m$ grids. The result of using the grid method is to discretize the planning space ABCD-EFGH into many points in the space. Assuming that the set of points is $S$, then there exists the following mathematical correspondences between any point $P(i,j,k)$ in the planning space and point $P(x,y,z)$ in the three-dimensional coordinate system O-XYZ (where $i=0, 1, 2, \ldots, m$ and $j=0, 1, 2, \ldots, n$, $k=0, 1, 2, \ldots, m$):

$$x = -l + \frac{2l*i}{m}, \quad y = \frac{j \cdot h}{n}, \quad z = -l + \frac{2l*k}{m} \quad (2-1)$$

The divided planning space for mobile robot is shown in Figure 5.
Finally, 3D terrain modelling using grid method is shown in Figure 6.

![3D terrain modelling using grid method](image)

**Figure 6** 3D terrain modelling using grid method

2.2 **Algorithm flow design**

The calculation process of the three-dimensional path planning algorithm in this study is as follows:

1. Construct an environment model, initialize each parameter, and input basic data. Set the optimal step counter \( N_3 = 0 \). Set \( N_r = 0 \) while \( N_r \) is the number of all paths found by the ant after each global pheromone update. Set the allowed list allowed \( a = 0 \); set the direction flag direct \( a = 0 \); determine the positions of the starting point and the target point in the environment model; put all ants at the starting positions.

2. For each ant \( k \) with the current node \( P(i^a, j^a, k^a) \) as the center, select and go to the next node \( P_{a+1}(i^{a+1}, j^{a+1}, k^{a+1}) \).

3. Perform pheromone update locally.
(4) Determine whether all ants can find the foraging path for the first time, if not, go back to step (2). Start looping steps (2) and (3).

(5) Update the pheromone globally to determine whether the ant has found the optimal path. If it meets the requirement, print out the optimal result. If not, return to step (2), and repeat steps (2) and (3) again. Until all conditions are met, the algorithm ends. The flow chart is shown in Figure 7.

![Flow chart design](image)

**Figure 7** Flow chart design

2.3 Simulation experiment and analysis of the algorithm

At the end of the simulation experiment of path planning, MATLAB software was used to simulate the ant colony algorithm. The choice of MATLAB software as the simulation software is mainly because it is simple and easy to use. The results of the operation are illustrated by pictures, which are intuitive and easy to understand.

Taking the simulated three-dimensional topographic map as an example, the three-dimensional environment is set to 21km*21km*2km, and the ant colony algorithm designed above is used for path planning. Use MATLAB software simulation for simulation. The results of the simulation experiment are shown in Figure 8.
2.4 Performance analysis of improved ant colony algorithm

The number of ants, m, is a very critical parameter to improve the overall performance of the ant colony algorithm. When the size of the ant group is large, the stability and global search ability of the ant colony algorithm will be improved, but when the number of ant colony groups is too large, it will lead to a more balanced change in the amount of pheromone before finding the optimal path, weakening the positive feedback of the ant colony algorithm and also slows down the running speed of the algorithm. However, if the number of ants is set up to be too small in pursuit of the convergence speed, the randomness of the global search of the algorithm will be weakened and the stability will be deteriorated. Considering the combining effect of the three-dimensional planning space for the robot and the division degree of the grid, the algorithm selects the number of ants $m' = 20$ (in order to avoid the symbol conflict with the grid division behind, $m'$ is used to denote the number of ants).

The functions of information heuristic factor $\alpha$, expected heuristic factor $\beta$ and pheromone residual factor $\rho$ are tightly coupled. During the path planning of mobile robots, if the combination of $\alpha$, $\beta$ and $\rho$ is improperly set, it will cause the algorithm to solve too slowly and the quality of the obtained solution will be poor. The improved ant colony algorithm in this chapter has considered the selection of parameters at the beginning of the design. Based on the combination of theory and experiment, the improved ant colony algorithm sets the information heuristic factor $\alpha$ and the expected heuristic factor $\beta$ to be 1 by default, and settings of other relevant parameters after referring to the results of multiple experiments are shown in Table 1.

| m   | n   | $\lambda$ | $\mu$ |
|-----|-----|-----------|-------|
| 21  | 21  | 0.2       | 0.2   |

Taking the simulated three-dimensional topographic map as an example, the three-dimensional environment is set up to be 21km*21km*2km, and the designed ant colony algorithm mentioned above is used for path planning. The results obtained by MATLAB software simulation are shown in Figure 9. Figure 9 (a) and (b) are the simulation effect diagram of the improved ant colony algorithm and the simulation effect diagram of the basic ant colony algorithm respectively. From the effect diagrams, the path planned by the improved ant colony algorithm is significantly improved compared with the unimproved path.
The superiority of an algorithm is mainly reflected in the value of the optimal results obtained by the algorithm, the operational efficiency of the algorithm and the practicability of the algorithm. Taking simulation experiment as an example here, it can be seen intuitively from Table 2 that the improved ant colony algorithm is superior to the basic ant colony algorithm in terms of path planning effect and planning efficiency.
Table 2 Performance comparison of different algorithms

| Algorithm                      | Comparison Quantity | 1    | 1    | 3    | Average Value |
|--------------------------------|--------------------|------|------|------|--------------|
| Improved Ant Colony Algorithm | Optimal Path Value | 43.188 | 39.860 | 37.081 | 10.043       |
|                                | Planing Time       | 1.801 | 1.692 | 1.838 | 1.777        |
| Basic Ant Colony Algorithm     | Optimal Path Value | 51.018 | 42.761 | 47.214 | 46.988       |
|                                | Planing Time       | 9.506 | 4.048 | 6.184 | 6.579        |

3. Research on fuzzy Sarsa learning based on ant colony algorithm
This chapter uses the good random search ability of ant colony optimization. Based on the fourth chapter, we further proposes fuzzy Sarsa learning (ACO-FSL) algorithm with a variable learning rate based on ant colony optimization. The main idea is to apply the ant colony optimization (ACO) in fuzzy Sarsa learning (FSL). The design method and flow of the algorithm are given. The simulation experiment of the car climbing problem shows that the proposed ACO-FSL algorithm has better learning performance.

3.1 ACO-FSL system structure
For complex learning tasks in continuous space, FSL cannot automatically adjust the learning factors online. Therefore, based on the structure of Figure 10, we propose a variable learning rate fuzzy Sarsa learning (ACO-FSL) algorithm based on ant colony optimization. The idea is to apply ant colony optimization (ACO) to FSL. The structure of its system is shown in Figure 10.

3.2 Ant Colony Optimization (ACO)
Ant Colony Optimization (ACO), as a meta-heuristic search strategy, has demonstrated excellent performance and efficiency in solving various combinatorial optimization problems. ACO can be described by a graph. Let \( \tau_{ij}(k) \) be the pheromone level of the edge \((i, j)\) at the kth iteration, and \( n_{ij} \) be the communication heuristic value, then The probability of an ant starting from the vertex \( i \) selecting vertex \( j \) at the kth iteration at time \( t \) is:
Among them, $J(i)$ is the set of all other possible vertices that the ant can reach from the vertex $i$, and $\beta$ is a parameter related to pheromone. The update rule of pheromone level is as follows:

$$
\tau_{ij}(K + 1) = \rho \tau_{ij}(K) + \Delta \tau_{ij}(K)
$$

(3-2)

Here, $\rho \in (0, 1)$, $1- \rho$ is the evaporation coefficient. If $L_{op}$ is the current path of the ant, $C$ is a constant, and $F(.)$ is a pheromone quality function, then the pheromone update rule is defined as:

$$
\Delta \tau_{ij}(K) = \sum_{i=1}^{R} \mu_{s_i}(s_t) \omega_{ij}^t
$$

(3-3)

According to the concept of ant colony optimization, we design the update rule for the change of learning rate as:

$$
a_t = \frac{(\omega_t^{ij} - \omega_t^{ij}) p_t^{ij}}{\Delta \omega_t^{ij}}
$$

(3-6)

where:

$$
\Delta \omega_t^{ij} = \frac{Q(s_t, a_t) \times \mu_{s_i}(s_t)}{\sum_{i=1}^{R} \mu_{s_i}(s_t)} \cdot e_{ij}(t)
$$

(3-7)

Here, $e_{ij}(t)$ is the qualification track and its definition is as follows:

$$
e_{ij}(t) = \begin{cases} 
\gamma e_{ij}(t-1) + 1, & \text{if } j = i \\
\gamma e_{ij}(t-1), & \text{otherwise}
\end{cases}
$$

(3-8)

Where $0 \leq \gamma \leq 1$ is the delay parameter.

### 3.3 Action selection

In the ACO-FSL algorithm, we first divide the fuzzy system according to the fuzzy rules, and then select the executable actions from the discrete set of actions $A=\{a_1, a_2, ..., a_n\}$. For each fuzzy rule, $N$ candidate actions can be selected. The first step here is to discretize the continuous space. The purpose of this is to narrow the search space and provide learning efficiency. For each fuzzy rule, we must choose the optimal action from the $N$ candidate actions according to their Q value. These rules are defined as follows:

$$
R_t: IFS_1 IS_{t1} AND, ... AND S_{t}\_Is_{t1} THEN a_{t1} \_with Q_t1 or ... ora_{tN} with Q_{tn}
$$

Here $S_1, S_2, ..., S_n, is$ the input state space, $L_{ij}$ is the fuzzy set, and $a$ is the selected action for the output. The goal of reinforcement learning is to find the best action for each fuzzy rule, so that the agent gets the maximum cumulative discount return value. In order to achieve this goal, we regard the entire fuzzy inference module as the process of ants foraging, and use the excellent random search performance of the ant colony algorithm to solve the combined optimization problem. For example, in Figure 11, there are 3 fuzzy rules $R_1, R_2, R_3$, defined in the fuzzy inference system, and each fuzzy rule has 4 candidate actions $a_1, a_2, a_3, a_4$. Starting from the initial position on the left, the ants move through the fuzzy rule $R_1 R_2$ and stop at the fuzzy rule $R_3$, so that the optimal path for the ant has a combination of $4^3=64$. The dotted line in the figure is one of the combinations, which is a kind of foraging path for ants. According to the pheromone level, a pheromone matrix $\tau_{ij}$ of size LxN can be obtained, where $i=1, 2, ..., L, j=1, 2, ..., N$. 

$$
p_t^{ij} = \begin{cases} 
t_{ij}(K)h_0^{ij}(k) \sum_{k=1}^{R} t_{ij}(K)h_0^{ij}(k) & \text{if } j \in (i) \\
0 & \text{otherwise}
\end{cases}
$$

(3-1)
3.4 ACO-FSL Algorithm Steps

The steps to get the ACO-FSL algorithm are as follows:

step1. Observe the status $s_t$ and obtain the return value $r_t$; and then calculate the vector $\mu(s_t)$.

step2. If the predetermined learning requirements are met, then stop learning.

step3. Use (3-4) and (3-5) to calculate $a_t(s_t)$ and $Q(s_t, a_t)$.

step4. Set the temperature parameter $T$.

step5. For each fuzzy rule, calculate the probability to select the appropriate action.

step6. Use formulas (3-7) and (3-8) to calculate the continuous actions $\Delta \omega_{ij}$ and $\epsilon_{ij}(t)$.

step7. Automatically adjust the learning rate $a_t$ according to (3-6).

step8. According to the updated $Q(s_t, a_t)$, calculate the error $\Delta Q(s_t, a_t)$.

step9. Use (3-5) to calculate the new $Q_{t+1}(s_{t+1}, a_{t+1})$.

step10. The selected action acts on the environment.

step11. t-t+1, return to step1.

As can be seen from the above, the main difference between the ACO-FSL algorithm and the IFSL algorithm in Section 3.3 is that the adjustable parameter learning rate $a$ and the temperature parameter $T$ are set differently. In the IFSL algorithm, the learning rate is according to the designed minimum function and $Q(s_t, a_t)$ is calculated using fuzzy reasoning. In ACO-FSL, the learning rate is calculated according to the function designed in (3-6), and the temperature is set according to the pheromone matrix.

3.5 Simulation experiment - simulation of car climbing problem

In order to compare the learning performance of FSL and ACO-FSL two algorithms, a simulation platform for car climbing problem is adopted. Trolley hill climbing learning control. In the literature on reinforcement learning, it is usually used as a typical continuous state space reinforcement learning.
problem to verify the learning efficiency and generalization performance of the algorithm. Using simulation software, Figure 12 shows the output curve of the learning controller under the ACO-FSL algorithm, where the horizontal axis is time steps and the vertical axis is the output of the discrete control variable \( u \). It reflects the change of the control quantity during the learning process of the car from the starting point to the target point. The following figure shows the change curve of the learning rate of the learning controller under the ACO-FSL algorithm. From the figure, it can be seen that the learning rate of the car during the mountain climbing process is not fixed. By dynamically adjusting the learning rate online, you can effectively improve the learning speed.

![Figure 12 Control Output from the Learning Controller](image1)

**Figure 12** Control Output from the Learning Controller

![Figure 13 Learning rate curve of the Learning Controller](image2)

**Figure 13** Learning rate curve of the Learning Controller

In order to eliminate the influence of many random factors in one operation, the two algorithms are independently run 50 simulations. Figure 13 is the curve of the position change of the car from the starting point to the target point under the two algorithms in a certain operation. It can be seen that the car under the ACO-FSL algorithm can reach the target point within 31 time steps, while under the FSL algorithm it takes about 33 time steps to reach the target point.
Figure 14 Motion change curve of the dolly

Figure 14 only reflects the curve of the position change of the car from the starting point to the target point in a certain operation. The statistical performance of the two algorithms in 50 runs is shown in Table 3. It can be seen from the statistical results that ACO-FSL is superior to FSL in both learning speed and learning efficiency.

The main idea of ACO-FSL is to apply the ant colony optimization to the FSL, adjust the learning rate online through the ant colony optimization, achieve the purpose of controlling the learning speed, construct the pheromone matrix to optimize the learning performance index.

| Algorithm                                      | FSL | ACO-FSL |
|-----------------------------------------------|-----|---------|
| The average time step of the car to reach the target point | 37.2 | 32.8 |
| The minimum time step of the car to reach the target point | 35  | 26     |
| The maximum time step of the car to reach the target point | 42  | 36     |
| The ratio of the car successfully reaching the target point (%) | 96.8% | 98.5% |

4. Conclusion

Path planning and learning are hot topics in robotics. This paper mainly introduces the ant colony algorithm into the path and learning of mobile robots to explore the optimal path solution. The following work has been accomplished:

(1) The principle of robot path planning is briefly presented, and the three-dimensional environment is modeled. In environment modeling, the grid method most suitable for three-dimensional space is used for modeling, and finally the three-dimensional space is divided into nodes in space. Finding the optimal path is to select a minimum point set from the initial point to the target point among these nodes, and the nodes in the set constitute the optimal path in the three-dimensional space.
(2) Use MATLAB software for experimental simulation and run out the three-dimensional space graphics of path planning. Finally, by changing the parameters, some improvements were made to the ant colony algorithm to improve the efficiency of the robot path planning. And on the basis of Sarsa algorithm, choose the robot learning action, compare the learning efficiency to achieve the effectiveness and adaptability in parallel.

References
[1] Shi Feng, Wang Hui, Yu Lei and Hu Pei. MATLAB AI Algorithm—30 Case Analysis, Beijing University of Aeronautics and Astronautics Press, 2012.
[2] Deng Wen, Li Shi, Zheng Pan. Research on Path Planning Based on Ant Colony Algorithm, Logistics Technology, 2008, 27(10): 136-138.
[3] Liu Yanju, Yang Qingchuan, Gu Yinyin. Application Research of Ant Colony Algorithm in Robot Path Planning, Computer Science, 2008, 35(5): 263-265.
[4] Sun Shudong, Lin Mao, a multi-robot coordinated planning based on genetic algorithm. Journal of Automation, 2000, 26(5): 672-676.
[5] Zhou Ming, Sun Shudong, Peng Yanwu, Robot path planning based on genetic simulated annealing algorithm. Journal of Aeronautics, 1998, 19(1): 188-120.
[6] Yang Ming, Hong, He Kezhong, etc., Lidar-based Mobile Robot Environment Modeling and Barriers, Journal of Tsinghua University (Natural Science Edition), 2000, 40(70): 112-116.
[7] Yuan Zeng, Ren, Gao, A new method of mobile robot navigation and obstacle avoidance in a dynamic environment, Robot, 2000, 22(2): 81-88.
[8] Tan Bai, J. G, Gong Cheng, A New Landmark Framework for Mobile Robot LocMization and Asynchronous sensor Fusion. The proceedings of 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation, 2003.
[9] Gregory Dudek, M. J. Computational Principles of Mobile Robotics. Cambridge University Press, 2000.
[10] Ricardo Gutierrez-Osuna, J. A. J, Ren C Luo, Modeling of UI transonic Range Sensor for Localization of Autonomous Mobile Robot. IEEE Transactions on Industrial Electronic.