Research on optimization of complex path of inspection robot

Dongyang Li \(^1\)\(^a\), Zhifeng Zhong \(^2\)\(^b\), Tianxiong Chen \(^3\)\(^c\), Qiangqiang Yan \(^4\)\(^d\)

\(^1\)Department of Computer and Information Engineering University of Hubei Wuhan, Hubei Province, China
\(^2\)Department of Computer and Information Engineering University of Hubei Wuhan, Hubei Province, China
\(^3\)Department of Maritime Academy University of Hainan Vocational Science and Technology Haikou, Hainan Province, China
\(^4\)Department of Computer and Information Engineering University of Hubei Wuhan, Hubei Province, China
\(^a\)1010914972@qq.com
\(^c\)*1187856708@qq.com
\(^b\)zfzhong@hubu.edu.cn
\(^d\)471932021@qq.com

Abstract—It is a common method to use intelligent robots for automatic inspection in highly automated ships. Path planning and optimization research are an important part of robotics. Intelligent tracking is the key to autonomous movement of mobile robots. In this paper, a grid map is used to establish a simulation model of the robot tracing environment. In a grid map, the more complex the map environment, the slower the algorithm will converge. The -fill-merge method processes the concave obstacles in the environment into regular shapes, reduces the number of unnecessary calculations, thereby greatly optimizing the tracing environment and speeding up the convergence speed. Finally, the MATLAB simulation experiment shows that the method based on environment optimization can effectively accelerate the convergence speed, avoid the local optimum, and quickly find the global optimum when dealing with the problem of autonomous tracking of intelligent robots in a map environment with a large number of concave obstacles.

1. Introduction

Intelligent ships are an important direction for the future development of ships. With the increasing automation and complexity of the equipment in the ship’s engine room, and the increasing degree of unattended, the use of inspection robots to inspect the equipment on the ship is a common method. Inspection robots can effectively reduce labor costs, and can effectively locate where faults may occur in real time. In a complex environment, the robot inspection path is complex, resulting in low inspection efficiency, and it is difficult to find an optimal path for efficient inspection. Therefore, it is of great significance to study the path optimization of inspection robots in complex environments.

The research on autonomous movement of robots is a hot topic in the current scientific field\(^{[1-4]}\). Many scholars at home and abroad have applied various optimization algorithms to mobile robots and achieved good results: Literature \(^{[5]}\) adopts ant colony algorithm and passed Setting the unequal initial
pheromone concentration enables the ants to perform a focused search in the area where the optimal solution is most likely to appear, which improves the efficiency of the algorithm; literature [6] uses the method of artificial potential field to effectively avoid obstacles; [7] proposed a global path planning method based on the ray model, which can effectively reduce the search time. Although the above document has some optimization for robot path planning, but all but ignored the impact of environmental factors, in complex environmental conditions, there will still be slow convergence, it is difficult to find the optimal solution and so on. Literature [8] takes into account the impact of the environment and applies a negative feedback mechanism. While increasing the search range, it also increases the probability of falling into a dead end. Literature [9] uses a fallback strategy to reduce the probability of ants entering concave obstacles. However, this method is computationally intensive and cannot completely prevent entry into concave obstacles.

In order to better enable the robot to quickly and effectively find the optimal path in a complex environment, this article proposes a method to optimize the environment from the perspective of the environment, that is, it can fill in quickly and effectively without blocking the effective path. Concave obstacles are combined with other optimization algorithms (ant colony algorithm is used in this article) to realize the path planning of the robot.

2. Environment modeling and environment optimization

Commonly used environmental modeling methods include grid graph method [10], free space method [11], geometric information method [12] and so on. This paper chooses a relatively simple and intuitive grid map method to model the environment of the target two-dimensional motion space, and fills in the concave obstacles in the map environment, that is, environment optimization. The grid is composed of a matrix of 0 and 1. 0 represents a free grid, which means that there is no obstacle and can pass freely, and the color is white; 1 represents a taboo grid, which means that there is an obstacle that needs to be moved around, and the color is black. Set both the free grid and the taboo grid to be a square with a side length of 1, as shown in Figure 1.

![Fig.1 Raster map](image)

In order to facilitate the later algorithm to calculate the distance between each grid and the moving sequence of ants between grids, each grid in the \( n \times n \) grid matrix needs to be numbered. The number \( i \) and the corresponding relationship with its coordinate \((x, y)\) are shown in formula (1) Shown:

\[
\begin{align*}
x &= \begin{cases} 
\text{mod}(i / n) - 0.5 & \text{mod}(i, n) \neq 0 \\
\text{mod}(i, n) \neq 0 \\
n + \text{mod}(i / n) - 0.5 & \text{otherwise}
\end{cases} \\
y &= n - \text{ceil}(i / n) + 0.5
\end{align*}
\]

When there are a large number of concave obstacles in the environment, the ants are easy to fall into the concave trap, resulting in low search efficiency and possibly even stagnation. In order to solve this kind of problem, the environment optimization method adopted in this paper deals with the concave obstacles in the map environment through decomposition-fill-merge method.
Decomposition process: Take the connected points in the map as a whole. The expression is as follows:

\[ S(i, j) = \text{unique}(f(i_1, j_1) \cup f(i_2, j_2)) \quad f(i_1, j_1) \cap f(i_2, j_2) \neq \emptyset \]  

(2)

The meaning of function \( f(x, y) \) is that when there is an obstacle at point \((x, y)\), the coordinates of the point adjacent to point \((x, y)\) and the obstacle is placed in a set \( S(x, y) \); \( f(i_1, j_1) \cup f(i_2, j_2) \) means that if there are overlapping coordinate points between set \( f(i_1, j_1) \) and set \( f(i_2, j_2) \), then put the elements in the set \( f(i_2, j_2) \) into the set \( f(i_1, j_1) \). The unique function is to remove duplicate elements in the collection.

Through the above formula, multiple sets can be finally obtained. Each set stores the position coordinate information of an independent obstacle (that is, each set is a sub-map containing only one obstacle), and then each sub-map is filled to deal with. The formula is as follows:

\[
\begin{align*}
F(X_{\text{min}} \rightarrow X_{\text{max}}, \text{index1}) & \quad Y_{\text{min}} \leq \text{index1} \leq Y_{\text{max}} \\
F(\text{index2}, Y_{\text{min}} \rightarrow Y_{\text{max}}) & \quad X_{\text{min}} \leq \text{index2} \leq X_{\text{max}}
\end{align*}
\]  

(3)

Among them, \( X_{\text{min}} \) and \( X_{\text{max}} \) respectively represent the maximum and minimum values of \( x \) in the set \( S \), and \( Y_{\text{min}} \) and \( Y_{\text{max}} \) represent the maximum and minimum values of \( y \) in the set \( S \), respectively. The \( F(x, y) \) function assigns a value to the point \((x, y)\), 1 means that there is an obstacle, 0 means that there is no obstacle, and the meaning of the \( F \) function is that when \( y = \text{index} \) (or \( x = \text{index} \)), the corresponding filling is carried out in the \( x \) direction (or \( y \) direction).

After filling each sub-map, merge the sub-maps to obtain the final processed map. The formula is shown in formula (4), and the whole process is shown in Figure 2.

\[ S = S(i_1, j_1) + S(i_2, j_2) + \cdots + S(i_n, j_n) \]  

(4)

Fig. 2 Environment optimization process

3. Introduction to Ant Colony Algorithm

Ant colony algorithm is a population-based simulation evolution algorithm proposed by simulating the foraging behavior of ant colonies. Each ant positively feeds back to the following ants through the pheromone released on the moving path, thereby speeding up the convergence speed and finding the optimal path in a shorter time \([13]\).

When the ant moves, the direction of movement is determined according to the pheromone left in the front. The greater the pheromone concentration, the greater the probability of being selected. The probability \( p_{ij}^k(t) \) of ant \( k \) moving from position \( i \) to position \( j \) at time \( t \) is shown in equation (5) \([14]\).
\[ p_{ij}^k(t) = \begin{cases} \frac{\tau_{ij}^\alpha(t)\eta_{ij}^\beta(t)}{\sum_{\text{allowed}_k} \tau_{ij}^\alpha(t)\eta_{ij}^\beta(t)} & j \in \text{allowed}_k, \\ 0 & \text{otherwise} \end{cases} \tag{5} \]

In the formula, \( \tau_{ij}^\alpha(t) \) represents the residual pheromone concentration on path \( ij \) at time \( t \); \( \alpha \) represents the pheromone heuristic factor; \( \eta_{ij}^\beta(t) \) represents the expected heuristic function between time \( t \) and \( ij \); \( \beta \) is the expected heuristic factor, and \( \text{allowed}_k = \{ \text{Tabu}_k \} \) is the node that the ant is allowed to select in the next step. \( \text{Tabu}_k \) is a taboo table, which records the nodes that ant \( k \) has walked. The desired heuristic function is defined as the reciprocal of the distance \( d_{is} \) between the node \( i \) and the target point \( s \), which is \( \eta_{ij}(t) = 1/d_{is} \).

"Ants" will release pheromone (including positive and negative pheromone) on the path it walks, to provide direction information for the "ants" behind. In order to avoid the excessive concentration of pheromone, the heuristic information will be submerged and invalidated, historical pheromone will volatilize with the passage of time. Assuming that the pheromone volatilization coefficient is \( \rho \) (\( 0 < \rho < 1 \)), the update rule of the pheromone is shown in equation (6).

\[ \tau_{ij}(t+\Delta t) = (1-\rho)\tau_{ij}(t) + \sum \tau_{ij}^k(t) \tag{6} \]

In formula (6), \( \tau_{ij}^k(t) \) represents the pheromone left by the \( k \)th ant between nodes \( i \) and \( j \) in this iteration. Here, the Ant-Cycle model proposed by Dorigo \cite{14} is used, as in formula (7) Shown.

\[ \Delta \tau_{ij}^k(t) = \begin{cases} \frac{Q}{L_k} & (i,j) \in p_k(\text{begin,end}) \\ 0 & \text{otherwise} \end{cases} \tag{7} \]

In the above formula, \( Q \) is the pheromone intensity; \( L_k \) is the total length of the path traversed by the ant \( k \) in this cycle; \( p_k(\text{begin,end}) \) is the path traversed by the ant \( k \) from the starting point to the end point in this cycle.

4. Experimental simulation

The environment of this experiment is: win10 system, CPU is E5-2697, 128G running memory.

In this paper, three sets of comparative experiments are carried out using the single-variable control method, and the environment optimization method based on environment filling and the ant colony algorithm are used to verify the speed and effectiveness of the robot's autonomous tracking. The scale of the map in Figure 3 is \( 20 \times 20 \), with fewer obstacles, and the environment is relatively simple; the scale of the map in Figure 4 remains unchanged, but the number of obstacles is increased, and the environment is more complicated; the scale of the map in Figure 5 is expanded to \( 40 \times 40 \), and further increase the number of obstacles, and make the distance between the obstacles more compact, the distribution is more disorderly, and the environment is more complicated.
Fig. 3 Comparison in a simple environment

(a) ACO

(b) ACO+ filling

(c) Convergence curve

The figure shows the robot trajectory for different algorithms in a simple environment. The convergence curve compares the minimum path length for ACO and ACO+ filling algorithms with respect to the number of iterations.
Fig. 4 Comparison in a more complex environment
Comparing (a) and (b) in Figures 3, 4, and 5, we can see that in an unoptimized map environment, the optimal path found by the algorithm will fall into a trap, and there will be local convergence at the end. This situation not only increases the path length, but also increases the amount of calculation and the possibility of algorithm stagnation. The essence of optimization is to optimize the map environment. Under the premise of not blocking the effective path, the dead ends in the map are filled, reducing the detours and dead ends in the path. In the three filled environments, the optimal path found...
by the ant colony algorithm is relatively short, which completely eliminates the situation of falling into a concave obstacle.

From the three convergence curves, it can be seen that the convergence speed after filling is faster, the convergence path is shorter, and the previous convergence curve fluctuates more smoothly. Under the same number of iterations, the path obtained is shorter. The filling and ant colony algorithm can achieve better results in the more complex map environment. The specific performance indicators are shown in Table 1, Table 2, and Table 3:

Table 1 Performance indicators in a simple environment

| Path length | Number of iterations | ACO running time | Fill time |
|-------------|----------------------|------------------|-----------|
| ACO         | 35.8                 | 14               | 8.538     | -         |
| ACO+ filling| 32.1                 | 6                | 7.621     | 0576      |

Table 2 Performance indicators in a more complex environment

| Path length | Number of iterations | ACO running time | Fill time |
|-------------|----------------------|------------------|-----------|
| ACO         | 35.8                 | 22               | 7.961     | -         |
| ACO+ filling| 32.9                 | 18               | 7.073     | 0973      |

Table 3 Performance indicators in complex environments

| Path length | Number of iterations | ACO running time | Fill time |
|-------------|----------------------|------------------|-----------|
| ACO         | 73.7                 | 18               | 62.179    | -         |
| ACO+ filling| 61.5                 | 8                | 56.804    | 10.354    |

In summary, in an environment with a large number of concave obstacles, the ant colony algorithm is easy to fall into the local optimum, that is, into the concave obstacle; after the environment is optimized, the ant colony algorithm can easily find the optimal solution. And it will not fall into concave obstacles, while reducing the number of iterations required for convergence, effectively improving the efficiency and accuracy of algorithm operation. Although the total running time of the ant colony algorithm plus filling is relatively large, the filling method only needs to be run once when the map environment is unchanged. When the target point position changes, it only needs to search on the first optimized environment, which can reduce the time of the algorithm as a whole.

5. CONCLUSION

This paper uses the filling method combined with the ant colony algorithm to carry out multiple sets of comparative experiments, mainly through the number of iterations required for convergence (convergence speed) and the length of the convergence path to perform performance analysis, verifying the effectiveness of the filling method. The experimental results show that the filling method can effectively improve the calculation efficiency of the algorithm, reduce the execution time of the algorithm, and reduce the dependence of the algorithm on its own parameters. Environmental optimization is more effective in complex environments and can effectively solve path planning problems in many complex environments. In addition to being suitable for inspection robots, it can also be used for Mars rover to autonomously plan paths to target points in an unfamiliar Martian environment. In urban traffic, scenes such as real-time dynamic optimization of the environment and finding the optimal route based on road conditions.

References

[1] Mohd. Nayab Zafar and J. C. Mohanta and Anupam Keshari. GWO-Potential Field Method for Mobile Robot Path Planning and Navigation Control[J]. Arabian Journal for Science and Engineering, 2021, : 1-18.
[2] Qiang Huang. Research on Path Planning of Mobile Robot based on Particle Algorithm[J]. International Core Journal of Engineering, 2021, 7(6)
[3] Shao J. Robot Path Planning Method Based on Genetic Algorithm[J]. Journal of Physics: Conference Series, 2021, 1881(2)
[4] Chen Y Y and Zhou X M. Research and Implementation of Robot Path Planning Based on Computer Image Recognition Technology[J]. Journal of Physics: Conference Series, 2021, 1744(2).

[5] Chen Y J, GAO M T. Ant colony algorithm based on initial allocation and dynamic update of pheromone[J]. Computer Engineering and Applications, 2021: 4-6.

[6] Wang F, Li K P, Yang M X. An Ant Path Planning Algorithm Guided by Artificial Potential Field[J]. Computer Science, 2014: 47-48.

[7] Jiang L, Fang D J, Zhou H W, Huang H B. Improved global path planning algorithm based on ray model[J/OL]. Chinese Journal of Electronics: 1-10[2021-06-27].

[8] Bai J L. Research on Robot Path Planning Based on Ant Colony Algorithm of Negative Feedback Mechanism[D]. BeiJin: Wanfang Data Knowledge Service Platform, 2019: 22-28.

[9] Zhen W Q. Research on Robot Path Planning Based on Classification Search Ant Colony Algorithm[D]. BeiJin: Wanfang Data Knowledge Service Platform, 2018: 23-25.

[10] Kambhampati S, Davis L S. Multiresolution path planning for mobile robots[J]. IEEE J Robotics & Automation, 1986, 2(3): 135-145.

[11] Oommen B J, Iyengar S S, Rao N S V, et al. Robot navigation in unknown terrains using learned visibility graphs. Part I: The disjoint convex obstacle case[J]. IEEE Journal on Robotics and Automation, 1987, 3(6): 672-681.

[12] Zheng H, Peng C. Collaboration and fairness in opportunistic spectrum access[C]. IEEE International Conference on Communications, 2005, 5: 3132-3.

[13] Dorigo M, Gambardella L M. Ant system: optimization by a colony of cooperating agents[J]. IEEE Transactions on Evolutionary Computation, 1997, 1(1): 53-66. 136.

[14] Shi L X. Research on robot path planning based on improved ant colony algorithm[J]. Aeronautical Computing Technology, 2021, 51(02): 28-31.

[15] Qu Zhenggeng, Yang Chuan. Research on mobile robot global trajectory planning based on improved ant colony algorithm[J]. Journal of Nanjing Normal University (Natural Science Edition), 2015, 38(01): 81-85.