Path planning for unmanned wheeled robot based on improved ant colony optimization

Lin Wang

Abstract
This research presents a simple and novel improved ant colony optimization for path planning of unmanned wheeled robot. Our main concern is to avoid the random deadlock situation and to reach at the destination using the shortest path, to decrease lost ants and improve the efficiency of solutions. The aforementioned reasons, we design an adaptive heuristic function by adopting the Euclidean distance between the ant and the target destination, in order to avoid the initial blindness and later singleness of ant path searching. The historical best path when appropriate to retain the previous effort would supersede the current worst path. Simulation results under random maps show that the improved ant colony optimization considerably increases the number of effective ants. During the searching process, the probability to find the optimal path increases, as well as the search speed. Moreover, we also compare the improved ant colony optimization performance with the simple ant colony optimization.

Keywords
Unmanned wheeled robot, ant colony optimization, path planning and optimal path planning

Date received: 28 December 2019; accepted: 28 January 2020

Introduction
Path planning relates to the mission execution efficiency and navigation safety of unmanned wheeled robot (UWR), and is one of the key technologies of UWR. The traditional path planning methods mainly include Dijkstra method, A* algorithm, and so on. Many scholars use intelligent bionic algorithms such as particle swarm optimization (PSO) and ant colony optimization (ACO) to solve path-planning problems. Among them, ACO is a positive feedback moreover, heuristic random search method to simulate biological population foraging. It has the advantages of good robustness, strong global search ability and convenient expression of environmental constraints.

In the literature, path planning uses the ACO, and the simulation verifies the effectiveness of the method. In order to further improve the performance of the ACO, the literature combines the improved fireworks algorithm with the ACO. It generates the initial pheromones on the path through the fireworks algorithm, while the literature uses PSO method to optimize the parameters of the ACO. These two methods can make the ACO have better initial pheromone distribution and parameter configuration. It also avoids the bad practice of relying on manual experience to select algorithm parameters. However, it also increases the complexity of the path planning process to some extent.

In the grid map, considering that there is no obvious difference in the distance between adjacent grids, using its reciprocal as a heuristic function has little effect on path search. Therefore, the literature uses the reciprocal of the Euclidean distance from the grid around the ant to the end grid as the heuristic function. That enables the ant to move to the end point with greater probability. This method avoids the blindness of the initial path search of the algorithm to some extent. Nevertheless, we ignore the cost of going from the current grid to the next grid. In classic ACO, ants can only transfer between adjacent grids each time. It limits the convergence speed and accuracy of the algorithm. To solve this problem, the literature proposes a multi-step ACO which is not limited to the surrounding grid.
The algorithm can search a shorter path with fewer transfer steps, but the path search process in large-scale maps is more complicated. In the literature, the pheromone of the ACO resides on the connection path between the grids, and when the map is large, it needs to take up a lot of memory space to store the pheromone distribution matrix. Therefore, according to the characteristics of path planning in grid map, we propose an ACO with pheromone stored in grid, which reduces the complexity of the algorithm.

When we apply ACO to path planning in grid map, there exists the problem of ant deadlock, which reduces the number of effective ants in the algorithm and affects the search efficiency of the algorithm. This paper analyzes the phenomenon of deadlock in the ACO, and put forward several solutions. Among them, when the ant has a deadlock, this paper first punishes the pheromone of the current grid, and then makes the ant take a step back. This method can reduce the probability of the ant falling into the deadlock in the same position.

The key objectives of this research are the following: (1) to implement the ACO in which pheromones resides in the grid to perform UWR path planning; (2) analyze the deadlock problem deeply, and propose a corresponding deadlock handling strategy; (3) design adaptive heuristic function to improve the processing of optimal path planning; (4) MATLAB used to simulate the classic and improved algorithms; and (5) the simulation results shows the efficiency and effectiveness of our proposed algorithm when compared with the classic algorithm.

The overall content of this article is as follows: section “Problem description” describes the problem and establishes the simulation environment and the optimal path model we use which is followed by ACO with pheromone stored in grid and the improved ant colony optimization (IACO) discussions in sections “ACO with pheromone stored in grid” and “Improved ant colony optimization” Section “Implementation of IACO” implements the IACO. Section “Simulation research and discussion,” present the simulation results and discussions which shows the effectiveness and efficiency of the proposed algorithm. Finally, section “Conclusion” presents the conclusion.

Problem description

Establishing the environment

To help the ACO search the optimal path, we use the grid method to set up a two-dimensional (2D) random space which breaks into a grid randomly, and we number the grid from lower to upper, left to right. The grid with obstacles is an undesired grid, and the grid without obstacle is a free grid as shown in Figure 1.

Equation (1) establishes the relation between coordinates and the grid

\[
\begin{align*}
  x &= r \times (\text{mod}(n, R) - 0.5) \\
  y &= r \times (R + 0.5 - \text{ceil}(\frac{r}{R}))
\end{align*}
\]  

(1)

In above equation, grid’s scale is \( r \), the number of grid is \( n \), \( R \) is the total rows, \( \text{mod} \) returns the remainder of two numbers after division, and \( \text{ceil} \) returns the smallest integer value that is bigger than or equal to a number.

Optimal path model

In the grid map, assume that UWR is in the center of the grid, and we only select the adjacent free grid as the target for each transfer. When we transfer UWR in the up, down, left and right directions, the grid before and after the transfer should meet the requirements

\[
\begin{align*}
  &\text{Map}(g) = 0 \\
  &d_{gc} = \sqrt{(x_g - x_c)^2 + (y_g - y_c)^2} = 1
\end{align*}
\]  

(2)

In the above equation, \( c \) is the grid where UWR currently is, \( g \) is the grid where UWR moves, and \( d_{gc} \) is the distance between grid \( g \) and \( c \). Because the UWR is not a particle, when it moves diagonally to the upper left, lower left, upper right, and lower right grids, we also want the grids on both sides of the diagonals to be free. When the UWR moves to the upper left or lower left, the grid should satisfy

\[
\begin{align*}
  &\text{Map}(g) = 0 \\
  &d_{gc} = \sqrt{(x_g - x_c)^2 + (y_g - y_c)^2} = \sqrt{2} \\
  &|g - c + M| \mod M = 1 \\
  &\text{Map}(g - 1) = 0 \\
  &\text{Map}(c + 1) = 0
\end{align*}
\]  

(3)

When UWR shifts to the upper right or lower right, the grids before and after the transfer should meet

\[
\begin{align*}
  &\text{Map}(g) = 0 \\
  &d_{gc} = \sqrt{(x_g - x_c)^2 + (y_g - y_c)^2} = \sqrt{2} \\
  &|g - c + M| \mod M = M - 1 \\
  &\text{Map}(g + 1) = 0 \\
  &\text{Map}(c - 1) = 0
\end{align*}
\]  

(4)

Equations (2)–(4) are the constraint conditions of UWR transfer, which is the basis to judge whether its adjacent grid is feasible or not. Assuming that the
UWR is from the starting grid $S$ to the destination grid $E$ after $N$ transfers under the transfer constraint, and then the path length is written as

$$L = \sum_{i=1}^{N} d_{o_{i}o_{i+1}} + 1$$  \hspace{1cm} (5)$$

where $\{ o_1, o_2, \ldots, o_{n+1} \}$ is the set of grids that make up the path, and $o_1 = S$, $o_{n+1} = E$; $L$ is the path length. UWR has several feasible paths from the initial point to the end point, and the path planning works out an optimal or suboptimal feasible path. This paper mainly studies the shortest path, and its mathematical model is written as

$$L_{\text{opt}} = \min(L)$$  \hspace{1cm} (6)$$

where $\min(\ )$ represents the operation of taking the minimum value.

**ACO with pheromone stored in grid**

The principle of using ACO for path planning in grid map is similar to that of solving traveling salesman problem (TSP) problem. However, the constraint conditions of the TSP are different from the termination conditions of ant path search in the algorithm,\textsuperscript{12,13} which are as follows:

1. Ants do not have to traverse all the grids in the map from the start to the end;
2. Ants may appear deadlocked during the transfer process.

Based on the first difference, we can consider storing pheromones in each grid and stipulating that ants are more likely to transfer to adjacent grids with high pheromone concentration. This pheromone storage method can reduce the memory space requirements of the algorithm, reduce the amount of computation in the pheromone update process, and improve the execution efficiency of the algorithm.\textsuperscript{14,15} At this point, the state transition probability is written as

$$P_{ij}^{k}(t) = \begin{cases} \frac{\tau_{ij}^{k}(t) \eta_{ij}^{k}(t)}{\sum_{r \in \text{allowd}_{ik}} \tau_{ij}^{k}(t) \eta_{ij}^{k}(t), j \in \text{allowd}_{ik}} & \text{if } j \in \text{allowd}_{ik} \\ 0, & \text{otherwise} \end{cases}$$  \hspace{1cm} (7)$$

where $P_{ij}^{k}(t)$ is the probability that ant $k$ moves from grid $i$ to grid $j$ at time $t$; $\tau_{ij}^{k}(t)$ represents the pheromone concentration of grid $j$; $\alpha$ is the information heuristic factor, which represents the relative importance of the trajectory; $\eta_{ij}^{k}(t)$ represents the heuristic function, which is $1/d_{ij}$; $\beta$ is the expected heuristic factor; and $\text{allowd}_{ik}$ represents that when the $k$ ant is on grid $i$, it can move to the next grid set, which is written as

$$\text{allowd}_{ik} = \text{allowd} - \text{Path}$$  \hspace{1cm} (8)$$

In the above equation, $\text{allowd}$ is the set of feasible grids among eight grids around grid $i$; $\text{Path}$ is the set of $k$ ants passing through the grid. In each iteration, all ants complete a path search, and the search stops when each ant reaches the end point or when there is a deadlock. At the end of each iteration, according to the path searched by each ant, the pheromone concentration of each grid is as follows

$$\tau_{j}^{k}(t+1) = (1-\rho)\tau_{j}^{k}(t) + \sum_{k=1}^{m} \Delta\tau_{j}^{k}$$  \hspace{1cm} (9)$$

where $m$ represents the total number of ants; $\Delta\tau_{j}^{k}$ is the pheromone concentration increase brought to grid $j$ by the $k$th ant in this iteration. Only the ants who successfully searched for the end point contributed to the population, and only their pheromones can be used for reference. Therefore

$$\Delta\tau_{j}^{k} = \begin{cases} \frac{Q}{L_{k}}, & \text{ant } k \text{ reaches the end and passes grid } j \\ 0, & \text{other} \end{cases}$$  \hspace{1cm} (10)$$

In the equation, $Q$ is constant and $L_{k}$ is the length of the path searched by ants. From equations (9) and (10), we can conclude that the ACO with pheromone stored in grid is not suitable for solving TSP problem; otherwise, the pheromone increment of all nodes in the solution process would always be the same.\textsuperscript{10–15}

**Improved ant colony optimization**

**Deadlock handling strategy**

Deadlock means that in the process of path search, after the ant moves to a non-destination grid, there is no next grid that meets the transfer conditions, resulting in the forced termination of the path search. The ants with deadlock do not reach the end point, and the search path is invalid, which is equivalent to reducing the number of effective ants in the algorithm, which is not good for the convergence speed and accuracy of the algorithm.

In Figure 2, any ant moving to a concave obstacle area is bound to have an obstacle deadlock. At present,
there are many literatures on the deadlock situation. The proposed solution is to remove pheromones or add an obstacle list to the grid where the ants are, to prevent other ants from entering the grid again. Different paths have different effects on the same grid, and its analysis and processing is much more complicated. The probability of deadlock in some maps is high, which may have a serious impact on the performance of the algorithm.

In this paper, we establish global obstacle list and local obstacle list to store different types of grids. The global obstacle list $G_{Obs}$ is empty during the initialization of the algorithm. During the execution of the algorithm, the list keeps track of the grids where the ants face deadlock. The obstacle list has a constraint effect on all ants in each iteration of the algorithm. The local obstacle list $L_{Obs}$ empties when each ant starts the path search and only stores the grid that the ant passes through and does not belong to the global obstacle list. $L_{Obs}$ only constrains the current ants in this iteration.

When the ant $k$ on the grid $i$, transfers, it first determines whether a deadlock currently occurs. If equation (11) is satisfied, it means that no deadlock has occurred:

$$ (allow_d_i \cap (L_{Obs} \cup G_{Obs})) \neq allow_d_i $$  \hspace{1cm} (11)

When a deadlock does not occur, add grid $i$ to $L_{Obs}$ and make the ants continue to transfer to the next grid according to equation (7). At this time, equation (8) should be

$$ allow_d_i = allow_d_i - (L_{Obs} \cup G_{Obs}) $$  \hspace{1cm} (12)

If the deadlock occurs when the ant is on grid $i$, the type of deadlock needs to be further judged. If it is an obstacle deadlock, it should meet

$$ \text{Card}(allow_d_i \cap L_{Obs}) = 1 $$  \hspace{1cm} (13)

Among them, the $\text{Card}()$ function represents the number of elements in the collection. At this time, add grid $i$ to $G_{Obs}$, and then return the ant to the previous grid. If grid $i$ is already in $L_{Obs}$, remove it from $L_{Obs}$.

If a deadlock occurs and does not satisfy the equation (13), it indicates that the deadlock is more complex. In most cases, a step back is of limited help to jump out of the deadlock, while returning directly to the starting point is equivalent to increasing the total number of ants in the algorithm, which essentially does not improve the performance of the algorithm. Therefore, this paper adopts a compromised transfer strategy, that is, when the ant deadlock occurs, it moves to the eight surrounding grids. The grid that passes first and the grid that forms the ring part in the path is removed from $L_{Obs}$. In implementation, assume that $L_{Obs}$ contains $m$ elements $\{o_1, o_2, \ldots, o_m\}$, first take the set $A$ as

$$ A = allow_d_i \cap L_{Obs} $$  \hspace{1cm} (14)

Adaptive heuristic function design

The path search in the TSP problem needs to traverse all nodes, so it is impossible to estimate the shortest distance from the next node to the end point, and its heuristic function only uses the reciprocal of the distance between adjacent nodes. However, in the grid map, the distance $d_j$ of the adjacent grid is 1 or 1.414. When the pheromone concentration evenly distributed at the initial stage of the implementation of the ACO, the heuristic function in equation (7) cannot work. In the process of transfer, we generally want the ant to move to the end point with the shortest straight line distance, so the Euclidean distance between the transferred grid $j$ and the destination grid $E$ can also be used as a reference factor in the transfer process, and the heuristic function is written as

$$ \eta_j(t) = \frac{1}{d_{ij} + d_{IE}} $$  \hspace{1cm} (15)

Based on equation (15), when the scale of the map is large and the ants are far from the end point, the difference of the heuristic function values corresponding to different grids is very small, and the path selection still has a high randomness. When the ants approach the end point, the heuristic function values of different grids are too different, which will make the path choice of ants tend to be single in the area close to the destination. Therefore, based on equation (15), we introduce the distance between the current grid and the end grid, and the adaptive heuristic function becomes

$$ \eta_j(t) = \frac{1}{d_{ij} + d_{IE} - d_{IE} + C} $$  \hspace{1cm} (16)

In the equation, $C$ is a constant. The equation (16) can make the ratio between the heuristic function values of the ants choose different grids, which do not change greatly with the location, and overcome the defect of the heuristic function in the equation (15). Through $C$, you can set the range of the ratio between the values of the heuristic function.

Implementation of IACO

Step 1. Initialize the algorithm parameters, determine the maximum number of iterations $NC_{max}$, the maximum number of ants $M_{max}$, let the variable $k = 1$, $nc = 1$. Figure 3 define the overall working of IACO.
Step 2. Let the deadlock counter $\text{dead}_\text{num} = 0$, and perform steps 2.1–2.3 to implement the path search of ant $k$.

Step 2.1. Select path using our proposed transfer strategy. Determine whether ant $k$ is deadlocked according to equation (11). If no deadlock occurs, follow equations (7), (12), and (16). Perform the transfer, then skip to step 2.3; otherwise, continue to step 2.2.

Step 2.2. Process according to the proposed deadlock handling strategy, while the deadlock counter $\text{dead}_\text{num} = \text{dead}_\text{num} + 1$, then proceed to step 2.4.

Step 2.3. If ant $k$ does not reach the end point and $\text{dead}_\text{num}$ is less than the set number of chances, go back to step 2.1; otherwise, terminate the ant’s path search and continue to step 3.

Step 3. Let $k = k + 1$, if $k \leq Mmax$, skip to step 2; otherwise, continue to step 4.

Step 4. If the length of the optimal paths in the $Mmax$ paths searched are longer than the optimal path searched in the previous round, then replace any invalid or longest path in the current round of iterations with the optimal path in the previous round.

Step 5. Update the pheromone according to equations (9) and (10).

Step 6. Let $\text{nc} = \text{nc} + 1$, if $\text{nc} \leq NC\text{max}$, then let $k = 1$, and go to step 2; otherwise, the algorithm iteration ends and the optimal path is output.

### Simulation research and discussion

In order to verify the effectiveness of the IACO, we implement this method to conduct multiple groups of path planning simulation studies. The simulation computer has the configuration of Intel i5 5th generation processor, 16 GB memory, 64-bit Win8 operating system and Matlab 2016 simulation software. The parameters of the algorithm are in Table 1.

### Case 1

In this case, we use our algorithm to perform multiple repeated path planning in a random map. We also plot the number of times we run the algorithm, the best path length, and total ants that we lost. The simulation results are in Figure 4(a)–(c).

The simulation results of Figure 4(a)–(c) show that the IACO can find a better path after some iterative calculation in random maps, and can successfully identify the grid that causes the deadlock in the path search process. It can also prevent other ants from having the same deadlock again. While the pheromone distribution is relatively uniform at the initial stage of the algorithm, the frequency of invalid ants is higher. At the same time, the larger the map size, the lower the

### Table 1. Improved ant colony optimization parameters.

| Parameter       | Value | Parameter | Value |
|-----------------|-------|-----------|-------|
| Number of iterations | 30    | $\beta$   | 7.8   |
| Number of ants   | 50    | $\rho$    | 0.325 |
| Chances          | 3     | $Q$       | 30    |
| $\alpha$         | 2.2   | $C$       | 10    |
Case 2

In order to prove the usefulness and the efficiency of our proposed algorithm, we compare the proposed algorithm and the classic algorithm, we perform repeated path planning multiple times in a random map, and the simulation results are in Figure 5(a)–(c).

The simulation results of the two algorithms show that the proposed algorithm can greatly decrease the ants lost in the map, and the average value of the optimal path length searched in each iteration is smaller and the convergence speed is faster. Combined with Figure 4(a) and (b), we can conclude that after 30 iterations, the probability of finding the feasible path of the classic algorithm is still lower than that of the IACO in the first iteration.

As shown in Table 2, our proposed algorithm is much better in getting the shortest possible path length and decreasing the number of total ants lost. The optimal path length of our proposed method is 75, and the average path length of the classic method turns out to be 88. These results demonstrate that our method is more efficient.

Conclusion

We presented an improved ant colony optimization (IACO) for the path planning of UWR. The IACO avoided the occurrence of deadlocks and reduced the probability of invalid ants caused by deadlock. At the same time, the adaptive heuristic function improved, and the optimal path retention strategy enhanced the algorithm’s convergence speed and accuracy. Simulation results under random maps show that the IACO considerably increased the number of effective ants. During the searching process, the probability to find the optimal path solution also increased, as well as the convergence speed. Moreover, we also compared the IACO performance with the simple ACO. The IACO has a simple implementation process, and the proposed deadlock handling strategy can be easily superimposed on ant colonies currently used for path
planning. In the algorithm, it does not conflict with other improvement measures of the algorithm and has high application value.

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.

Table 2. Comparison of the two algorithms in path planning.

| Algorithm                     | Optimal path length | Average number of lost ants | Average runtime (s) |
|-------------------------------|---------------------|-----------------------------|--------------------|
| Simple ant colony optimization| 88                  | 22                          | 1.277              |
| Our proposed algorithm        | 75                  | 15                          | 0.857              |

Figure 5. Simulation comparison of two algorithms: (a) proposed algorithm versus classical algorithm for optimal path planning, (b) average optimal path length, and (c) average number of lost ants.

Table 2. Comparison of the two algorithms in path planning.

| Algorithm                     | Optimal path length | Average number of lost ants | Average runtime (s) |
|-------------------------------|---------------------|-----------------------------|--------------------|
| Simple ant colony optimization| 88                  | 22                          | 1.277              |
| Our proposed algorithm        | 75                  | 15                          | 0.857              |

Funding
The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This research project funded by (1) project of young talents trust program of Shaanxi Association for Science and Technology (20190114), and (2) special scientific research plan project of Shaanxi Provincial Department of Education (19JK0432).

ORCID iD
Lin Wang https://orcid.org/0000-0001-9765-9415

References
1. Zeng Z, Sammut K, Lammas A, et al. Efficient path replanning for AUVs operating in spatiotemporal currents. J Intell Robot Syst 2015; 79(1): 135–153.
2. Yin C, Xiao Z, Cao X, et al. Offline and online search: UAV multiobjective path planning under dynamic urban environment. IEEE Internet Things 2017; 5(2): 546–558.
3. Huang H, Zhuo T. Multi-model cooperative task assignment and path planning of multiple UCAV formation. *Multimed Tools Appl* 2019; 78(1): 415–436.

4. Wang H, Peng L, Zhang Z, et al. Optimization method of power communication network based on improved particle swarm optimization. *IOP C Ser Earth Env* 2018; 170(4): 042014.

5. Yen CT and Cheng MF. A study of fuzzy control with ant colony algorithm used in mobile robot for shortest path planning and obstacle avoidance. *Microsyst Technol* 2018; 24(1): 125–135.

6. Gao Y, Wang J, Wu W, et al. A hybrid method for mobile agent moving trajectory scheduling using ACO and PSO in WSNs. *Sensors* 2019; 19(3): 575.

7. Zhao Y, Zheng Z and Liu Y. Survey on computational-intelligence-based UAV path planning. *Knowl-Based Syst* 2018; 158: 54–64.

8. Che G, Liu L and Yu Z. An improved ant colony optimization algorithm based on particle swarm optimization algorithm for path planning of autonomous underwater vehicle. *J Amb Intel Hum Comp*. Epub ahead of print 23 October 2019. DOI: 10.1007/s12652-019-01186-5.

9. Jurado S and Nebot Mugica AF. K nearest neighbour optimal selection in fuzzy inductive reasoning for smart grid applications. In: 2019 IEEE international conference on fuzzy systems (FUZZ-IEEE), 23 June 2019, pp. 1–6. New York: IEEE.

10. Aliper AM, Putin E, Zavoronkovs A, et al. Deep transcriptomic markers of human biological aging and methods of determining a biological aging clock. Patent application 16/104391, USA, 2019.

11. Wang B and Mueller K. The subspace voyager: exploring high-dimensional data along a continuum of salient 3D subspaces. *IEEE T Vis Comput Gr* 2017; 24(2): 1204–1222.

12. Mirjalili S. Ant colony optimisation. In: Mirjalili S (ed.) *Evolutionary algorithms and neural networks*. Cham: Springer, 2019, pp. 33–42.

13. Dorigo M and Stützle T. Ant colony optimization: overview and recent advances. In: Gendreau M and Potvin JY (eds) *Handbook of metaheuristics*. Cham: Springer, 2019, pp. 311–351.

14. Dorigo M and Birattari M. *Ant colony optimization*. New York: Springer, 2010.

15. Dorigo M and Blum C. Ant colony optimization theory: a survey. *Theor Comp Sci* 2005; 344(2–3): 243–278.