Hindawi
Discrete Dynamics in Nature and Society
Volume 2022, Article ID 4544499, 9 pages
https://doi.org/10.1155/2022/4544499

Research Article

Obstacle Avoidance Path Planning for UAV Based on Improved RRT Algorithm

Fan Yang,[1] Xi Fang,[1] Fei Gao,[1] Xianjin Zhou,[1] Hao Li,[2] Hongbin Jin,[2] and Yu Song[1]

[1]*School of Science, Wuhan University of Technology, Wuhan 430070, China*
[2]*People's Liberation Army Air Force Early Warning Academy, Wuhan 430070, China*

Correspondence should be addressed to Xi Fang; fangxi@whut.edu.cn

Received 22 October 2021; Revised 8 December 2021; Accepted 22 December 2021; Published 19 January 2022

Academic Editor: Shi Cheng

Obstacle avoidance path planning capability, as one of the key capabilities of UAV (Unmanned Aerial Vehicle) to achieve safe autonomous flight, has always been a hot research topic in UAV research filed. As a commonly used obstacle avoidance path planning algorithm, RRT (Rapid-exploration Random Tree) algorithm can carry out obstacle avoidance path planning in real time and online. In addition, it can obtain the asymptotically optimal obstacle avoidance path on the premise of ensuring the completeness of probability. However, it has some problems, such as high randomness, slow convergence speed, long transit time, and curved flight trajectory, so that it cannot meet the flight conditions of the actual UAV. To solve these problems, the paper proposes an improved RRT algorithm. In the process of extending the random tree, ACO (ant colony optimization) is introduced to make the planning path asymptotically optimal. The optimized algorithm can set pheromones on the path obtained by RRT and select the next extension point according to the pheromone concentration. And then through a certain number of iterations, it converges to an ideal path scheme. In addition, this paper also uses MATLAB to verify the effectiveness and superiority of the algorithm: Although RRT is easy to fall into local optimization, since the optimization method in this paper can almost certainly converge to the optimal solution, when it is necessary to preplan the path before UAV takeoff, it can be used.

1. Introduction

Planning on autonomous cruise of UAV (Unmanned Aerial Vehicle) with a collision-free track, which meets the high requirements about continuity constraints and output stability, is the research hotspot of multiagents' intelligence. With the increase of the complexity of planning space, it is more difficult to accurately calculate the UAV track. Thus, it is necessary to make a comprehensive study on how to obtain a track planning method that can effectively deal with most complex environments. Barros et al. [1] focus on the uncertainty of UAV and believe that this is mainly due to the existence of most random sampling points in the planned path. Zhang et al. [2] propose a new method to solve the UAV track problem by using a specific reward matrix. Ollero et al. [3] show the path planner is essential in the procedure schedule which is feasible given the specific constraints of UAVs. Merino et al. [4] give the techniques on trajectories planning of cooperated UAVs and decreased the uncertainty in fire detection and increased the fire localization accuracy. Oh et al. [5] design a framework and use the concept of differential geometry to discuss the cooperative mission and path planning of multiple UAVs.

By all rights, the desired well-planned path should have such features: is smooth, avoids obstacles, and has a relatively short path length. Therefore, many scholars use ant colony algorithm (ACO) to solve related problems. Wang et al. [6] introduce many common algorithms for solving UAV problems including ACO. They show that ants can find the best path because they have special secretions called pheromones, and by releasing pheromones, ants can influence the path of other ants. At the same time, it can be noted that, in each iteration of ACO, pheromone accumulation is faster than volatilization on the better path, but opposite on the worse path. Marco and Christian [7] read a lot of literature about ACO and then summarized its relevant application, optimization, and changes. Ma and Zhang [8] applied ACO to the route planning of cruise missiles with

complex environment and strict constraints. Bo and Zhao [9] proposed that cloud computing technology and multi-objective ACO can be used for route planning for UAV. On the basis of analyzing the tactical characteristics and mission requirements of UAV, Qian et al. [10] designed an ACO that can search and iterate independently. Deng et al. [11] introduced the yaw angle into ACO, so that it can search the optimal route more quickly and effectively. It can be known that when looking for the path, ants will focus on the path with high information concentration. After enough iterations, ACO can finally converge to a global better solution.

At the same time, RRT (Rapid-exploration Random Tree) algorithm is also used to solve such problems. Fu et al. [12] present a new variant of particle swarm optimization, phase angle-encoded and quantum-behaved PSO, which can be designed to generate a safe and flyable path in the different threat environments for UAV. RRT algorithm can effectively solve some problems existing in traditional path planning. Dong et al. [13, 14] give the experimental test about unmanned ground vehicle to deliver goods using RRT (Rapid-exploration Random Tree) algorithm for path planning, and many factors, such as collision detection, nodes selection, tree expansion of the RRT, are taken into account. Kong et al. [15] improved the search process of RRT by using the principle of odor diffusion. In order to develop a safe and effective lifting path in a complex environment, Wang et al. [16] applied RRT to crane hoisting. Wu et al. [17] greatly shortened the planning path of RRT by traversing the high-quality nodes of the sampling pool and using these nodes to customize the spanning tree. Taheri et al. [18] proposed a path planning method that can effectively reduce the running time and computational complexity, which is called fuzzy greedy fast exploration random tree (FG-RRT). Wang et al. [19] modelled the tree selection process as a multiarm bandit problem and used reinforcement learning algorithm to learn action values. It can not only enhance the local space exploration ability of each tree, but also ensure the efficiency of global path planning. Salzman and Halperin [20] combined RRT with another algorithm, and then the system decides which one to choose according to the value of approximation factor. Wei and Ren [21] give a path optimization strategy based on maximum curvature constraint, which can greatly improve the sampling speed of RRT and generate smooth continuous executable paths for robot manipulators. Fernando et al. [22] solved a motion planning problem by using RRT and obtained a feasible, low-cost, asymptotically optimal and probability complete path. Pharpatara et al. [23] combined the artificial potential field with RRT, which can accelerate the convergence speed to the suboptimal solution by bias against the generation of random states. Lin [24] shows an adaptive RRT algorithm based on dynamic step size and smooths the planning path by Hermite difference polynomial, which reduces the planning time and improved the path quality. Yang et al. [25] solves the difficult problem of programming under differential constraints. Because of its high optimization performance, RRT does not need to geometrically divide the task area first and the computational complexity will not change significantly with the increase of obstacles or threats. The algorithm should be completed under the query condition of sampling list and can quickly find a reasonable planning space combined with the information in the environment. Compared with other algorithms, it has significant advantages, such as fast adaptation to task environment, strong short-term replanning ability, fast dynamic change of environment, and so on. However, when the environment is surrounded by obstacles and the exit channel is very narrow, it often cannot find the exit, resulting in the failure of finding the path. Therefore, it is hoped to eventually converge to a good path with expectations by continuously optimizing RRT results. It can be known that there are different nodes in the convergence process of RRT. By analyzing these nodes, a greater probability is given to the nodes on the better path to extend to random trees; meanwhile, probabilities of nodes on the worse path of being extended are smaller. The next RRT expansion node is determined by the previous evaluation result, and then continuous iterative optimization is carried out to obtain a better solution. Considering that ACO can always find the best path after enough iterations, it can be used to optimize the path obtained by RRT and finally get better results. Based on this idea, ACO is introduced and an RRT path optimization method for ACO is proposed in this paper.

2. RRT Algorithm

When optimizing RRT, Feng and Liu [26] proposed a new idea of expanding random tree. At the same time, Lin et al. [27] also believed that the improvement of tree nodes is an important direction of RRT. When planning the path with RRT, once the leaf nodes of the random tree are included in the points in the target area, the expansion of the random tree stops and a line connecting the two points appears. The extension of RRT is shown in Figure 1.

In Figure 1, T represents the currently existing expansion tree, and q_{rand} represents a random sampling pt in the state space. The selection principle of q_{rand} is as follows: the target point q_{goal} is determined by probability goal – pro, and a point is randomly selected in the planning area with probability 1-goalpro. Besides the random sampling point q_{rand}, there is a closest tree node q_{near}, and then on the connection of q_{rand} and q_{near} a new node q_{new} will be intercepted in units of extended steps. If no obstacles are encountered in the process of moving toward the new node q_{near} things, then add q_{near} to the expansion tree; otherwise you need to reselect it as q_{rand}. Continue iterative calculation until q_{new} reaches the target area and the algorithm ends; then a path through the beginning q_{ini} and the end q_{goal} appears in the expanding tree T.

3. Comprehensive Improved RRT Algorithm

3.1. Add Optimization Process in RRT Algorithm. According to the above, it can be known that RRT can quickly obtain feasible flight path of UAV. However, it usually cannot obtain the shortest flight path. Aiming at this shortcoming, an improved RRT algorithm is proposed in this paper. The algorithm will add a new node to the tree and search whether there is a new parent node in the tree node in the ring neighborhood far away from it. If a lower cost path can be obtained by connecting new node to its parent node,

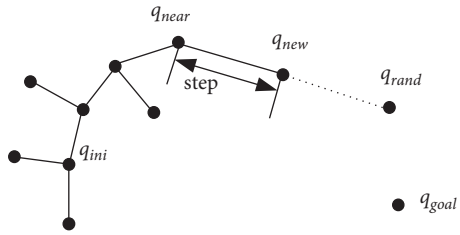


Figure 1: Basic RRT algorithm expansion process.

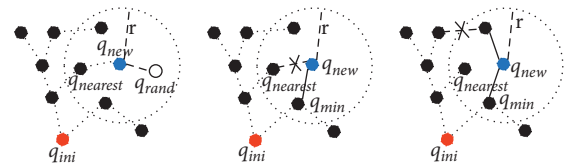


Figure 2: The path optimization process. (a) Node expansion. (b) Parent node reconnection. (c) Random tree reconnection.

the algorithm will just do it. In addition, it will search for other nodes around. If a shorter path can appear by connecting the tree node and the new node, the algorithm will do it. In Figure 2, the improved path optimization process described above will be shown.

After adding the optimization process, the algorithm can have asymptotic optimality while maintaining its own completeness. This solves the problem that the RRT algorithm cannot guarantee that the resulting path is an optimized path. The following table will describe the specific steps of this improved algorithm in detail:

Step 1. Firstly, the tree should be initialized, and starting point will become root node of this tree. In the space domain, sampling points are obtained by random sampling.

Step 2. Search the closest tree node in the neighborhood of sampling point to get the nearest neighbor.

Step 3. Starting from the position of the nearest neighbor, extending a step distance to the position of the sampling point, a new node is obtained, and the nearest neighbor is used as the minimum point near the new node.

Step 4. Check whether the path between the nearest neighbor and the new node is threatened by obstacles. If the path segment is not threatened by obstacles, it indicates that the new node is feasible, and it can be added to the random tree. Otherwise, the algorithm will give up on it and skip back to Step 1 to continue.

Step 5. Get a circle with the new node as the center and its step size as the radius to create a new neighborhood of the new node. Connect the tree nodes contained in the neighborhood to it, respectively, then check whether the newly connected path segments will be collided, and discard obstacles and regional point.

Step 6. Iteratively search whether there are tree nodes that meet the conditions in the cyclic neighborhood. The total cost of the new node and its parent node is greater than the nearest neighbor point obtained in Step 2. The path cost obtained after the new node is connected is smaller. If it exists, use this tree node to replace the nearest neighbor as the parent node of the new node and the minimum point in the neighborhood of the new node; otherwise continue to perform subsequent steps.

Step 7. Ignore the minimum point in the circular neighborhood obtained in Step 6, establish the connection in the neighbor between the new tree node and other tree nodes, and then check whether it will encounter obstacles to discard the points located in the obstacle area.

Step 8. Iteratively find whether there are tree nodes that meet the collision detection condition in the cyclic neighborhood. The total cost of the new node and its parent nodes should be greater than its current cost. If such a lower path exists, make the new node become the parent node and establish the contact path of the tree node and its child nodes; otherwise continue the subsequent steps.

Step 9. Repeat these steps unless the target point has been already in the tree.

The path is asymptotically optimal, but its process of searching for the optimal path increases the computation. The algorithm converges slowly and consumes more time, so its real-time performance is not as good as RRT algorithm. When we plan the path, we hope that the planning time is short and the planning path is short. It is most satisfactory to be able to plan the optimal or suboptimal path under the condition of ensuring real-time performance. Therefore, we continue to improve the algorithm.

3.2. Ant Colony Algorithm Is Added into RRT Algorithm. The intelligent optimization algorithm proposed by bionics research can get the ideal solution after enough iterations. In the optimization of ACO, pheromone accumulation speed on the better path is indeed faster than that on the volatile one. However, pheromone accumulation speed on the worse path is not slow. This phenomenon affects whether ants can choose a better path or not. Thus, the speed of ACO convergence is quite slow. At the same time, it is found that the speed of RRT to get the final path is very fast, but it is much different from the best solution. Therefore, pheromones are set according to the path obtained by RRT. Then the next expansion point is randomly obtained according to the number of pheromones through roulette selection method. Finally, it can come to an ideal path.

3.2.1. Pheromone Calculation. When the first-generation ants complete the path search, the task of subsequent ants can be regarded as the optimization problem of this path. This paper takes the traveling salesman problem as an example to illustrate the ant colony system model. Make the

following assumptions about the behavior of ants: the number of ants is m; the count of cities is n. Then the probability of ants in generation k from city i to city j at time t is

$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}(t)\eta_{ij}^{\beta}}{\sum_{u\varepsilon \text{allowed}_k}\tau_{iu}(t)\eta_{iu}^{\beta}}, & j\varepsilon \text{allowed}_k, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Among them, $\tau_{ij}(t)$ represents, at time t, the number of residual pheromones from city i to city j. At the beginning, the number of pheromones on each path is equal $\tau_{ij}(0) = C$; C is a constant; d_{ij} represents the distance from city i to city j, $\eta_{ij} = 1/d_{ij}$ means the heuristic factor between these two cities, and β corresponds to its importance; allowed_k indicates that ant k is the set of next allowed to reach the city.

On the path, the count of pheromones will accumulate with the release of ants and dissipate with the passage of time. There are three pheromone update methods in Ant System: ant volume model, ant dense model, and ant perimeter model. The first two models update the pheromone after each ant moves, while the last one updates the pheromone only after all ants arrive at the destination. Compared with the other two, ant perimeter model pays more attention to the change of global pheromone and is used more in the research process. Therefore, this model is selected in this paper. The calculation method of updating pheromone in ant perimeter model is as shown below:

$$\begin{aligned} \tau_{ij}(t+1) &= (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t), \\ \Delta\tau_{ij}(t) &= \sum \Delta\tau_{ij}^{k}(t), \\ \Delta\tau_{ij}^{k}(t) &= \begin{cases} \dfrac{Q}{L_k}, & \text{tour } (i,j) \text{ is done}, \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{2}$$

Among them, Q represents the preset total amount of pheromones, L represents the length of the path passed by ant k in the process of moving, and pheromones are distributed on this path evenly. p is the volatility coefficient of pheromone. It means that if p becomes larger, the speed of pheromone volatilization will become faster and the number of pheromones remaining on the path will become less. The smaller the influence of previous ant routing results, the stronger the randomness and the slower the convergence speed. It can accelerate the accumulation of pheromones and the convergence speed of algorithm, but it will also make the algorithm sink into the local optimal solution and result in the stagnation of the algorithm. When the algorithm gets a good enough solution or after a preset number of iterations, ants can get a path. This is the optimal solution after optimization.

The improved algorithm still uses grid method to describe the location of obstacles and various points in the state space. In subsequent optimization process, based on the route obtained by first-generation ants through RRT, we change the weight calculation method in the neighborhood from being affected by gravity field and repulsion field to being affected by pheromone.

3.2.2. Pheromone Update. If there are m ants moving in the state space during each iteration, the first-generation ants can find m paths through RRT. The m paths are given different pheromone concentrations according to a certain gradient: if the path passes through fewer grids, the path length will be shorter and more pheromones will be gathered, and conversely, if the path passes through more grids, the path length will be longer and fewer pheromones will be gathered. Besides, the pheromone concentration of all grid points covered by the same path is the same. Pheromones should be focused on pheromone points such as grid (i, j) on the grid point covered by the k-th optimal path being $\Delta\tau_{ij}^{k}$. When some grids are accessed by more than one path, pheromones on such grid point grid (i, j) are superimposed by pheromones on every path traversing the grid point:

$$\Delta\tau_{ij} = \sum \Delta\tau_{ij}^{k}, \quad k \in [1, m]. \tag{3}$$

When the first-generation m ants find m paths according to RRT, the pheromone concentration of all grid points on the map can be updated according to the above method. If some grid points are not passed by ants, their pheromones are still 0. Each generation of ants can leave certain pheromones for the reference of the next generation. For each grid point in the map, set a random initial weight random_weight_0 within a certain range; then the weight in each grid affected by the pheromone is

$$\text{weight}_0 = \left(1 + \Delta\tau_{ij}\right)\text{random\_weight}_0. \tag{4}$$

From the third generation, after each path search, while retaining all pheromones which belong to the previous generation, the pheromones belonging to other generations will volatilize to a certain extent. Similar to pheromone updating method in ant colony algorithm:

$$\tau_{ij}(t) = (1-\rho)\tau_{ij}(t-1) + \Delta\tau_{ij}(t-1). \tag{5}$$

Then the weight in each raster is

$$\text{weight}_0 = \left(1 + \tau_{ij}\right)\text{random\_weight}_0. \tag{6}$$

Once a generation of ants completes their task, pheromones and corresponding weight in each grid point on the map will be updated. According to the weights of the adjacent grids of the current grid, the ant selects the next grid point through the roulette wheel selection method.

4. Simulation

4.1. Theory of Track Planning. Obstacle avoidance planning of UAV means that, in the environment of unknown obstacles, UAV can independently analyze the environmental information and plan a collision-free path from the initial state to the target state under some constraints (such as the shortest time, the shortest distance, and the lowest energy consumption). It can be explained mathematically.

Suppose $U = R^{n_u}$ is the control input set of the system, $X = R^{n_x}$ is the state space of the system, $X_{\text{obs}} \subseteq X$ represents the threat area, $X_{\text{free}} \subseteq X$ represents the safety area, $X_{\text{obs}} = X_{\text{free}} = X$, and $X_{\text{obs}} \cap X_{\text{free}} = \varphi$. The following formula is used to describe the UAV dynamic equation in the track planning problem:

$$\begin{aligned} x(t) &= f(x(t), u(t)), \\ x(0) &= x_0, \end{aligned} \tag{7}$$

where $x(t) \in X$ is the system state of UAV; f is the UAV system model function; $u(t) \in U$ represents the system control input, and $x_0 \in x_{\text{free}}$ is the initial state of the system at time $t = 0$.

In order to avoid obstacles better, the UAV shall meet the following constraints when planning the road:

$$\begin{aligned} &x_t \in x_{\text{free}}, \\ &u(t) \in U, \quad \text{for } \forall t \in \left[0, t_f\right], \end{aligned} \tag{8}$$

where t_f represents the time taken for the UAV to fly from the starting point to the target point.

If $x_{\text{start}} \in x_{\text{free}}$ is the initial point and $x_{\text{goal}} \in x_{\text{free}}$ is the target point, the obstacle avoidance planning problem can be described as seeking the path from x_{start} to x_{goal} under the conditions in expressions (7) and (8), which can minimize the following objective function:

$$J\left(0, t_f, u\right) = \int_0^{t_f} g(x(t), u(t))\mathrm{d}t. \tag{9}$$

4.2. Problem Description. According to the above calculation method of map pheromone update, this algorithm is implemented by using MATLAB. The 500 ∗ 500 grid map is selected for the simulation experiment in this paper. During every iteration, it will let 5 ants search the path. For these 5 routes obtained by the search, all nodes on the path are extracted and the pheromone concentration is calculated. Different pheromone concentrations are obtained according to the number of times the node is recorded. Bring the pheromone concentration and the previously recorded RRT path nodes into the improved algorithm for iterative calculation; set the bias probability $p = 0.5$. When the probability is greater than 0.5, the ant colony algorithm will be selected; otherwise RRT algorithm will be selected. The 5 paths are sorted according to their length, and the shortest path is the local optimal path. The pheromone on each grid point covered by this path is 1. As the length becomes longer, the pheromones on the grid points covered by other paths decrease by 0.1 in order. On the longest route pheromone is 0.6. For pheromone remaining in the previous iteration, the pheromone volatilization coefficient ρ is taken as 0.7. The maximum count of iterations is 10. Once the number of iterations reaches 10, current optimal route will be output as global optimal route. The specific process is shown in Figure 3.

Under this set of parameters, traditional RRT algorithm, ant colony algorithm, and the comprehensively improved

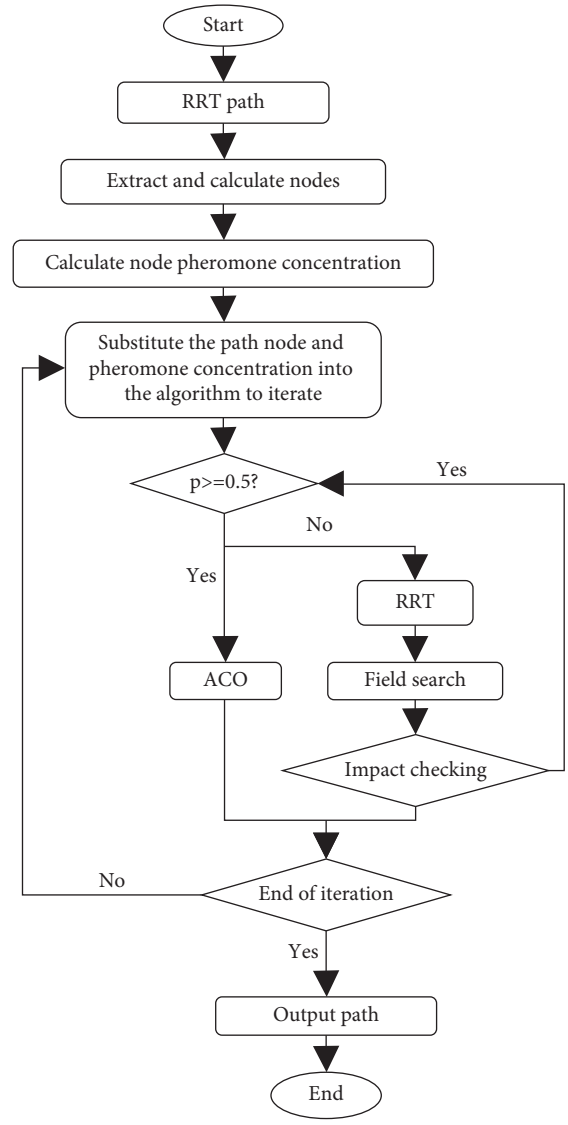
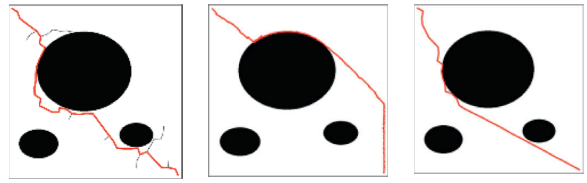


Figure 3: Flowchart of improved RRT algorithm.

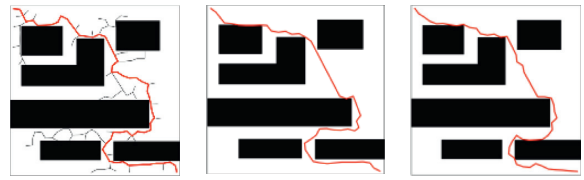
RRT algorithm of ant colony optimization were compared experimentally with MATLAB. After 10 iterations, the final results on four different maps are as follows.

4.3. Experimental Simulation. According to Figures 4–7 and Table 1, by comparing path lengths of RRT, ACO, and ant colony optimized RRT algorithm under four roadblocks, we can see that the paths of the last algorithm are obviously shorter than others. This verifies the effectiveness and superiority of our proposed algorithm. In map 1, the shortest path of ant colony optimized RRT is 15.5% shorter than RRT and 10.8% shorter than ACO. By comparing the other three maps, it can be seen that ant colony optimization RRT can usually be more than 15 percentage points less than RRT and more than 10 percentage points less than ACO. The advantage of ant colony optimization RRT is particularly obvious in map 2. Its shortest path is only 74% of RRT or 79% of ACO.



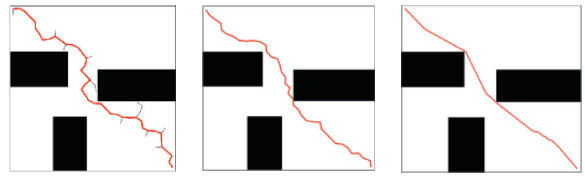
(a) (b) (c)

Figure 4: Map 1 running path diagram. (a) RRT. (b) ACO. (c) Ant colony optimization RRT.



(a) (b) (c)

Figure 5: Map 2 running path diagram. (a) RRT. (b) ACO. (c) Ant colony optimization RRT.



(a) (b) (c)

Figure 6: Map 3 running path diagram. (a) RRT. (b) ACO. (c) Ant colony optimization RRT.

(a) (b) (c)

Figure 7: Map 4 running path diagram. (a) RRT. (b) ACO. (c) Ant colony optimization RRT.

Table 1: The shortest paths of three algorithms under different maps

The shortest path	Map 1	Map 2	Map 3	Map 4
RRT	940	1259	806	1087
ACO	890	1178	789	968
Ant colony optimization RRT	**794**	**932**	**698**	**876**

After that, we compared the stability of the three algorithms under map 1, and the experimental results after 200 repetitions are listed in Table 2.

In Table 2, it is not difficult to see that the difference between the shortest path and the longest paths of ant colony optimized RRT is only 108, while the corresponding data of RRT and ACO reach 340 and 345, respectively. By calculating the distance variance of the results of the three algorithms, the advantage of ant colony optimization RRT in stability is more obvious. Its variance is only 16.1% of RRT and 14.4% of ACO. Therefore, it can be said that the stability of the improved algorithm has been greatly improved.

Due to the addition of domain optimization and ACO in the process of RRT algorithm, this new algorithm takes longer time for searching optimal route than RRT algorithm, but the results of the improved algorithm are obviously better than RRT and ant colony algorithm. The path is shorter and more practical. Here, three algorithms are used to find the path with a length of 900 to 930 in map 4, and the time required for the experiment is recorded.

In Figure 8 and Table 3, it can be seen that no matter how long the shortest path is set, the ant colony optimization RRT can always find the appropriate path, and the time is much lower than that of ACO. Although RRT can find the shortest path faster in some specific path lengths, it is difficult to find the path in most cases, such as when the shortest path is less than 920. To sum up, the algorithm is superior to RRT algorithm and ant colony algorithm in the time and reliability of finding feasible paths. This verifies its effectiveness and superiority.

Table 2: The stability of the three algorithms under map 1 (distance).

	Average distance	Shortest distance	Longest distance	Distance variance
RRT	905	803	1143	3267
ACO	1025	930	1275	3638
Ant colony optimization RRT	**787**	**740**	**848**	**525**

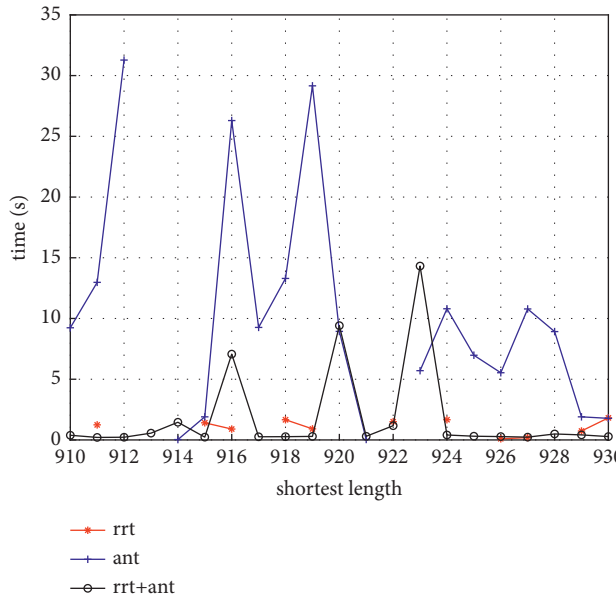


Figure 8: The time required for the three algorithms to find the path with a length of 900 to 930 in map 4.

Table 3: The time required for the three algorithms to find the path with a length of 900 to 930 in map 4.

	910	911	912	913	914	915	916	917	918	919
RRT	—	1.24	—	—	—	1.41	0.90	—	1.66	0.91
ACO	9.24	12.99	31.28	—	0.03	1.90	26.30	9.27	13.30	29.16
Ant colony optimization RRT	0.38	0.21	0.22	0.56	1.46	0.23	7.06	0.27	0.27	0.29
920	921	922	923	924	925	926	927	928	929	930
—	—	1.50	—	1.67	—	0.09	0.19	—	0.73	1.82
8.94	0.04	—	5.70	10.80	6.98	5.54	10.79	8.92	1.90	1.78
9.4	0.3	1.18	14.32	0.40	0.31	0.28	0.23	0.50	0.41	0.28

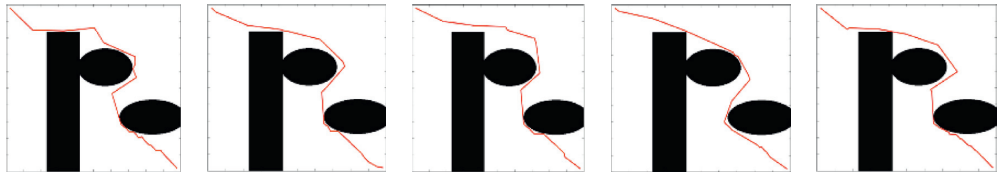
4.4. Discussion and Analysis of Parameters in Improved Algorithm. From the principle of ant colony algorithm searching for the shortest path, many of its parameters (such as convergence speed, path quality, and performance) are closely related to itself. The selection in this regard is usually determined according to experience and adjusted according to the experimental data. The general selection method and basis have not been formed.

In order to explore the impact of different parameters on the performance of the algorithm, this section takes map 4 as an example to conduct comparison experiments under different parameters.

4.4.1. Pheromone Volatilization Coefficient. Pheromones in ant algorithms are similar to human memory: memories of previous paths influence subsequent decisions and they will dissipate over time. ACO uses ρ to represent the volatility coefficient. The larger ρ, the faster the volatilization speed of pheromone, and the global search ability of the algorithm decreases. The smaller ρ, the slower the volatilization speed of pheromone. The algorithm has strong global search ability, but its convergence is poor.

Figure 9 and Table 4 are stating that, with the same other parameters, the value of ρ has a great impact on ACO's convergence speed. While ρ becomes smaller, positive feedback characteristic of the algorithm will be weakened and randomness will be enhanced. This makes it unable to converge well. With the increase of ρ, its positive feedback characteristics will be gradually enhanced and randomness will be weakened. However, it is always easy to fall into the local optimal state, so that it is difficult to obtain the global optimal solution. It is obvious from Table 4 that the best experimental result can be obtained when $\rho = 7$.

4.4.2. Population Size. ACO itself is essentially a random algorithm and seeks the optimal solution through the evolution of individuals in the population. Each individual in the population corresponds to a feasible solution, and m



(a) (b) (c) (d) (e)

Figure 9: Effect of different pheromone volatility coefficients on the path of ant colony optimization RRT algorithm (ρ is pheromone volatilization coefficient). (a) $\rho = 0.1$. (b) $\rho = 0.3$. (c) $\rho = 0.5$. (d) $\rho = 0.7$. (e) $\rho = 0.9$.

Table 4: Influence of different pheromone volatility coefficients on algorithm results.

ρ	0.9	0.7	0.5	0.3	0.1
Shortest distance	873.18	**831.61**	871.42	848.41	889.62

ants are a subset of the solution of the problem. With the increase of subsets and the count of individuals in population, the search ability of ACO will become stronger. However, once their number reaches a certain upper limit, the pheromone distribution will be relatively uniform, resulting in a reduction of the positive feedback of the algorithm. On the other hand, if the number of ant colonies is relatively small, its search ability will be reduced and it will be easy to stagnate. For different ant numbers, simulation results are shown below.

From Table 5, it can be seen that the number of ants has a great impact on the results. If the number of ants is close to the scale of the preset problem, the pheromone distribution will be relatively uniform so that it will hardly affect the next experimental process and results and will not get a better path. Studies show that, taking the TSP problem as an example, when the problem size is the number of cities, the suitable choice of the number of ants is between $\sqrt{n}$ and n/2. In this experiment, the effect is the best when $m = 70$.

4.4.3. Ant Maximum Feasible Distance. In Table 6, Ant_MaxDis is the longest distance that ants can walk. This means that when the ant algorithm is triggered, it will run to the longest distance to find the path. From the result data, it can be seen that ant_MaxDis is an important factor which affects the performance of the algorithm. When the other parameters are the same and ant_MaxDis is 150, the algorithm works better.

4.4.4. Ant Perception. In Table 7, ant_feelsize means the perception degree of ants. After using RRT algorithm, it starts to judge the use of ant colony algorithms and ensure that there are enough points on the path for ants. After testing, it can be seen from the results in the table that ant_feelsize has little influence on the results of algorithm. In other words, the RRT path is generated at the beginning of the algorithm, and it has little impact on the ant colony. Therefore, the randomness of RRT algorithm is solved, and this algorithm has good stability.

Table 5: Influence of different ant numbers on algorithm results.

m	10	30	50	70	90
Shortest distance	842.49	863.48	838.05	**826.82**	887.71

Table 6: Influence of different Ant_MaxDis on algorithm results.

Ant_MaxDis	50	100	150	200	250
Shortest distance	921.5314	887.5676	**852.1783**	857.5742	885.0413

Table 7: Influence of different ant_feelsize on algorithm results.

ant_feelsize	1	2	3
Shortest distance	882.7289	862.2385	866.8158
4	5	6	7	8
865.555	869.928	872.2428	867.1066	870.2232

5. Conclusion

The path planning problem of UAV obstacle avoidance is usually carried out in complex environment. Because the result of RRT algorithm is basically not affected by the complexity of the environment, this paper first thought of using it. However, the randomness of RRT algorithm is too strong to obtain a stable solution. At the same time, considering the strong convergence of ant colony algorithm, the optimal solution can always be found. Therefore, an RRT path optimization method based on ant colony algorithm is proposed. It obtains the global optimal solution through continuous iterative optimization of the path obtained by the basic RRT. In addition, this paper also carries out experimental simulation with MATLAB to verify the effectiveness, feasibility, and superiority of the integrated ant colony optimization RRT algorithm proposed in this paper. Therefore, when the path needs to be planned in advance before the UAV takes off, the RRT optimization method based on ant colony algorithm can be totally used.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by National Natural Science Foundation under Grant no. 61502522, Equipment Pre-research Fund under Grants no. JZX7Y20190253036101, Equipment Pre-Research Ministry of Education Joint Fund under Grants no. 6141A02033703, and Hubei Natural Science Foundation under Grant no. 2019CFC897.

References

[1] A. I. Barros, L. Evers, H. Monsuur, and A. P. M. Wagelmans, *UAV Mission Planning: From Robust to Agile*, Springer International Publishing, Berlin, Germany, 2015.

[2] B. Zhang, Z. Mao, W. Liu, and J. Liu, "Geometric reinforcement learning for path planning of UAVs," *Journal of Intelligent and Robotic Systems*, vol. 77, no. 2, pp. 391–409, 2015.

[3] A. Ollero, S. Lacroix, L. Merino et al., "Multiple eyes in the skies - architecture and perception issues in the comets unmanned air vehicles project," *IEEE Robotics and Automation Magazine*, vol. 12, no. 2, pp. 46–57, 2005.

[4] L. Merino, F. Caballero, J. Dios, and A. Ollero, "Cooperative fire detection using unmanned aerial vehicles," in *Proceedings of the IEEE International Conference on Robotics & Automation*, IEEE, Barcelona, Spain, April 2005.

[5] H. Oh, H. S. Shin, S. Kim, A. Tsourdos, and B. A. White, *Cooperative Mission and Path Planning for a Team of UAVs*, Springer, Berlin, Germany, 2015.

[6] J. Wang, S. D. Zhou, G. T. Zhu et al., "Research of common route planning algorithms for unmanned air vehicle," *Fire Control and Command Control*, vol. 8, p. 2, 2012.

[7] D. Marco and B. Christian, "Ant colony optimization theory: a survey," *Theoretical Computer Science*, vol. 344, pp. 243–278, 2005.

[8] X. Ma and N. Zhang, "Ant colony algorithm in route planning of cruise missile," *Ship Electronic Engineering*, vol. 33, no. 3, pp. 38–39130, 2013.

[9] L. Bo and G. Zhao, "Route planning algorithm based on mapreduce and ant colony optimization," *Computer Engineering*, vol. 41, no. 5, pp. 38–4455, 2015.

[10] Z. Qian, G. Wang, J. Wang, and Y. Shi, "Route planning of UAV based on improved ant colony algorithm," in *Proceedings of the International Conference on Logistics Engineering*, Shenyang, China, November 2015.

[11] T. Deng, Z. M. Xiong, Y. J. Liu, and Q. Z. Meng, "Research on 3D route planning for UAV in low-altitude penetration based on improved ant colony algorithm," *Applied Mechanics and Materials*, vol. 442, no. 4, pp. 556–561, 2013.

[12] Y. Fu, M. Ding, and C. Zhou, "Phase Angle-encoded and quantum-behaved particle swarm optimization applied to three-dimensional route planning for UAV," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 42, no. 2, pp. 511–526, 2012.

[13] Y. Dong, Y. Zhang, and J. Ai, "Experimental test of unmanned ground vehicle delivering goods using RRT path planning algorithm," *Unmanned Systems*, vol. 5, no. 1, pp. 45–57, 2017.

[14] Y. Dong, E. Camci, and E. Kayacan, "Faster RRT-based nonholonomic path planning in 2D building environments using skeleton-constrained path biasing," *Journal of Intelligent and Robotic Systems*, vol. 89, no. 3/4, pp. 387–401, 2018.

[15] Y. Kong, Y. Pan, and X. Chen, "A kind of two-stage RRT algorithm for robotic path planning," *International Journal of Wireless and Mobile Computing*, vol. 6, no. 1, pp. 34–38, 2013.

[16] X. Wang, Y. S. Lin, D. Wu, and C. W. Zhang, "Path planning for crane lifting based on Bi-directional RRT," *Advanced Materials Research*, vol. 446-449, pp. 3820–3823, 2012.

[17] D. Wu, Y. Sun, X. Wang, and X. Wang, "An improved RRT algorithm for crane path planning," *International Journal of Robotics and Automation*, vol. 31, no. 2, pp. 84–92, 2016.

[18] E. Taheri, M. H. Ferdowsi, and M. Danesh, "Fuzzy greedy RRT path planning algorithm in a complex configuration space," *International Journal of Control, Automation and Systems*, vol. 16, no. 6, pp. 3026–3035, 2018.

[19] W. Wang, L. Zuo, and X. Xu, "A learning-based multi-RRT approach for robot path planning in narrow passages," *Journal of Intelligent and Robotic Systems*, vol. 90, no. 1/2, pp. 81–100, 2018.

[20] O. Salzman and D. Halperin, "Asymptotically near-optimal RRT for fast, high-quality motion planning," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 473–483, 2016.

[21] K. Wei and B. Ren, "A method on dynamic path planning for robotic manipulator autonomous obstacle avoidance based on an improved RRT algorithm," *Sensors*, vol. 18, no. 2, p. 571, 2018.

[22] R. A. Fernando, G. Lopes, N. Pereira, and J. Lino, "Path planning towards non-compulsory multiple targets using Twin-RRT," *Industrial Robot*, vol. 43, no. 4, pp. 370–379, 2016.

[23] P. Pharpatara, B. Herisse, and Y. Bestaoui, "3-D trajectory planning of aerial vehicles using RRT," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 3, pp. 1116–1123, 2017.

[24] N. Lin, "An adaptive RRT based on dynamic step for UAV path planning," *Journal of Information and Computational Science*, vol. 12, no. 5, pp. 1975–1983, 2015.

[25] K. Yang, S. Moon, S. Yoo et al., "Spline-based RRT path planner for non-holonomic robots," *Journal of Intelligent & Robotic Systems: Theory & applications*, vol. 73, no. 1-4, pp. 763–782, 2014.

[26] B. Feng and Y. Liu, "An improved RRT based path planning with safe navigation," *Applied Mechanics and Materials*, vol. 494-495, pp. 1080–1083, 2014.

[27] Y. Lin, S. Gao, X. Wang, X. K. Wang, and D. Wu, "Improving RRT-connect approach for optimal path planning by utilizing prior information," *International Journal of Robotics and Automation*, vol. 28, no. 2, pp. 146–153, 2013.