Hindawi
International Journal of Aerospace Engineering
Volume 2021, Article ID 7667173, 19 pages
https://doi.org/10.1155/2021/7667173

Research Article

A Novel Three-Dimensional Path Planning Method for Fixed-Wing UAV Using Improved Particle Swarm Optimization Algorithm

Chen Huang [1,2,3]

[1]*College of Civil Aviation, Shenyang Aerospace University, Shenyang 110136, China*
[2]*Shenyang Academy of Instrumentation Science Co., Ltd., Shenyang 110136, China*
[3]*College of Mechanical Engineering, Dalian Jiaotong University, Dalian 116028, China*

Correspondence should be addressed to Chen Huang; huangchen054@163.com

Received 9 May 2021; Revised 6 June 2021; Accepted 8 July 2021; Published 31 July 2021

Academic Editor: Chen Pengyun

This paper proposed an improved particle swarm optimization (PSO) algorithm to solve the three-dimensional problem of path planning for the fixed-wing unmanned aerial vehicle (UAV) in the complex environment. The improved PSO algorithm (called DCA∗PSO) based dynamic divide-and-conquer (DC) strategy and modified $A*$ algorithm is designed to reach higher precision for the optimal flight path. In the proposed method, the entire path is divided into multiple segments, and these segments are evolved in parallel by using DC strategy, which can convert the complex high-dimensional problem into several parallel low-dimensional problems. In addition, $A*$ algorithm is adopted to generated an optimal path from the particle swarm, which can avoid premature convergence and enhance global search ability. When DCA∗PSO is used to solve the large-scale path planning problem, an adaptive dynamic strategy of the segment selection is further developed to complete an effective variable grouping according to the cost. To verify the optimization performance of DCA∗PSO algorithm, the real terrain data is utilized to test the performance for the route planning. The experiment results show that the proposed DCA∗PSO algorithm can effectively obtain better optimization results in solving the path planning problem of UAV, and it takes on better optimization ability and stability. In addition, DCA∗PSO algorithm is proved to search a feasible route in the complex environment with a large number of the waypoints by the experiment.

1. Introduction

1.1. Background and Motivation. In recent years, with the development of sensor technology and intelligent control technology, unmanned aerial vehicle (UAV) has been widely used in military and civil fields, such as reconnaissance [1, 2], surveillance [3], target prosecution [4], wireless communications [5, 6], and oilfield inspection [7]. As a basic technology for autonomous navigation of the UAV system [8], the path planner is an important component to ensure the successful completion of the complex missions. The objective of path planning is to seek an optimal or near-optimal flight route from the starting position to the destination in the mission space under the required constraint conditions [9].

The core of the path planning system is route planning algorithm, which has been actively researched for decades. Over the last few decades, a variety of approaches have been proposed to deal with path planning problem for UAV or autonomous robot. Most traditional path planning methods based on the graph are adaptive to 2-dimensional planning problems, such as Voronoi diagram algorithm [10], $A*$ algorithm [11], and probabilistic roadmap algorithm. These methods are relatively easy to implement, but UAV kinematic and dynamic constraints are seldom considered by these algorithms. Meanwhile, these methods need more time and huge storage memory in a large space. Potential field-based methods, such as artificial potential field algorithm [12], can obtain a smooth path and has little calculation time.

Its disadvantage lies that it is easy to trap into local minimum and even cannot find a feasible path when the distance between the target and the obstacles is near. Hence, potential field-based methods can seldom be used for the practical situations. Compared with the traditional methods, evolutionary computation (EC) algorithms have a strong ability to obtain high-quality solutions in path planning and are easier to implement [13]. The problem of trajectory planning is often treated as a nonlinear NP-hard optimal problem to be solved by evolutionary algorithms. It is well known that the ECs have made much progress and have been deeply and extensively studied in recent years [14–16]. EC algorithms, including evolutionary algorithms (EAs) and swarm intelligence algorithms (SIs), such as genetic algorithm (GA) [17], differential evolution (DE) [18, 19], particle swam optimization (PSO) [20–25], and ant colony optimization (ACO) [26], have been widely used in handling global optimization problems.

In the application of route planning of the three-dimensional (3D) environment, it is necessary to require a higher ability to avoid collisions with terrain and much more complicated calculation than two-dimensional path planning. Therefore, how to further enhance convergence speed and solution quality of the planner will be a main motivation in a 3D complex environment. Moreover, as the environment and the planning task become increasingly complex, intelligent optimization algorithms have difficulty in dealing with the high dimension problems.

1.2. Related Work. ECs have been applied to find global or approximate global optimum path. Yang et al. [27] discovered the drawback that the high-quality waypoints in previous search can hardly be further exploited in evolutionary algorithm-based due to all the waypoints of a path as an integrated individual. Hence, a new idea of separately evaluating and evolving waypoints is proposed for two-dimensional UAV path planning. On the basis of Ref. [14], Huang [28] proposed an improved PSO algorithm based on the competition of global best solution and applied the path planner to a three-dimensional UAV to demonstrate. Zhang et al. [29] formulated an improved fruit fly optimization (MAFOA) based on the phase angle-encoded connected with mutation adaptation mechanisms to solve the unmanned aerial vehicle (UAV) path planning problem. Du et al. [30] proposed a hybrid evolutionary algorithm included the main algorithm and a subalgorithm. The main algorithm was to evolve a population of main solutions, and the subalgorithm was used to optimize each UAV path. Yang and Yoo [31] proposed a joint genetic algorithm and ant colony optimization to find an optimal flight path in accordance with sensing, energy, time, and risk utilities.

Inspired by the foraging behavior of birds, PSO was proposed by Eberhart and Kennedy in 1995 [32]. PSO is a swarm intelligent optimization algorithm, which has many advantages, such as strong robustness, low sensitivity to population size, less adjustment parameters, and better optimization effects. In fact, the path planner is aimed at searching a group of waypoints, which is a NP optimization problem. In this case, PSO and its variants are used to solve the complex optimization problems. In the preliminary study, the PSO-based approach has been used to solve the path planning for fix-wing UAV, which showed that PSO can provide good guidance for the search of path planning owing to the globally best particle of the entire population. Therefore, this paper studies the PSO-based approach with dynamic divide-and-conquer (DC) strategy and modified $A*$ algorithm for solving the path planning problem.

However, the traditional PSO algorithm has some shortcomings, such as premature convergence, slow convergence speed in the later evolution, and it is easy to fall into local extreme point. Meanwhile, with the increase of problem size or dimension, PSO may be not sufficient in large-scale optimization problem due to large search space and exponential growth of local optimization. Recently, some researches have focused on these challenges. Zhang et al. [21] proposed a cooperative coevolutionary bare-bones particle swarm optimization (CCBBPSO) with function independent decomposition (FID), where binary encoding of the original model is converted to integer encoding to reduce the dimension. Cheng and Jin [33] proposed a novel competitive swarm optimizer (CSO) to solve large-scale optimization problem. Li and Yao [34] proposed a new cooperative coevolving particle swarm optimization (CCPSO2) algorithm, where the coevolving subcomponent sizes of the variables are determined dynamically. Liang and Suganthan [35] developed a modified dynamic multiswarm particle swarm optimizer (DMS-PSO), where the swarms are dynamic. Wang et al. [24] proposed a dynamic group learning distributed particle swarm optimization (DGLDPSO) for large-scale optimization. Cheng and Jin [36] developed a social learning PSO (SL-PSO), where social learning mechanisms are introduced into particle swarm optimization (PSO).

1.3. Contribution and Organization. To deal with the challenges of the traditional PSO algorithm, a dynamic DC strategy and modified $A*$ algorithm are proposed for the path planning problem of UAV. Specifically, three major novel designs and advantages that help DCA$*$PSO find the feasible path under minimizing the cost are described as follows.

On the basis of analyses of the cost function and constraints for UAVs, a new path planner is formulated to enhance the ability of solving high-dimensional route planning problem in complex 3D environment. The complex route planning problem is decomposed into a serious of small-scale subproblems based on the DC strategy. For each subproblem, only the coordinates of a few waypoints need to be concerned.

A new subsegment evaluate function is proposed for estimating the optimal solution. The subsegment evaluate function provides a reference for judging the equality of the whole path according to some waypoints in a path.

$A*$ algorithm is firstly adopted to optimize the particle composition and enhance the equality of the particles, which is embedded into the evolution process of the PSO algorithm. The $A*$PSO algorithm can provide a significant improvement in the convergence speed of the PSO algorithm and avoid the local optimum entrapment.

This paper is organized as follows. Section 2 introduces the environment and trajectory representation of route

planning for UAV. The optimization model of the path planning problem for fixed-wing UAV is established in Section 3. In Section 4, an improved PSO algorithm based on dynamic DC strategy and improved $A*$ algorithm is proposed. The improved PSO algorithm is used to solve the path planning problem in Section 5. In Section 6, the data simulation and analysis are introduced in detail. Finally, the conclusion is offered, and the future research direction is discussed in Section 7.

2. Environment and Trajectory Representation

2.1. Environment Representation. The representation of the environment space, e.g., the terrain and the danger zones, directly affects the efficiency of the planning algorithm and the quality of planning result. The terrain map of the mission space is described by a 2D matrix. The rows and columns of this matrix represent the x-coordinate and y-coordinate in a 3D space, respectively. The elevation data of the map is represented by the corresponding element of the matrix as follows:

$$\text{terrain} = \begin{pmatrix} e_{11} & e_{12} & \cdots & e_{1n} \\ e_{21} & e_{22} & \cdots & e_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ e_{m1} & e_{m2} & \cdots & e_{mn} \end{pmatrix}, \quad (1)$$

where e_{ij} is the altitude of the environment terrain.

If the coordinate of a waypoint is defined as (x, y, z), hence, the flying space for UAV can be given as

$$\begin{cases} x_{\min} \le x \le x_{\max}, \\ y_{\min} \le y \le y_{\max}, \\ z_{\min} \le z \le z_{\max}, \end{cases} \quad (2)$$

where $x_{\min}, x_{\max}, \mathrm{y}_{\min}, y_{\max}, z_{\min}, z_{\max}$ represent the boundaries of the 3D coordinate, respectively.

When a UAV flies into the high-risk threat of the radar, the defense scope of the radar may be considered as omnidirectional. The mathematical model of the threat source is denoted as a geometric sphere, as seen from Figure 1, which is described by the following matrix:

$$\text{danger zones} = \begin{bmatrix} x_1 & y_1 & z_1 & r_1 \\ x_2 & y_2 & z_2 & r_2 \\ \cdots & \cdots & \cdots & \cdots \\ x_d & y_d & z_d & r_d \end{bmatrix}, \quad (3)$$

where (x_k, y_k, z_k) is the center of the k-th radar threat, and r_k is the radius of the k-th threat source.

2.2. Trajectory Representation. There are two common representations to describe the trajectory: the first one is a continuous smooth trajectory, which is based on the description of UAV kinematic and dynamic characteristics. The smoothness of the trajectory may be omitted by this representation method. The other is represented by the waypoints, which are connected by straight line segments between the adjacent waypoints. The advantage of the second trajectory representation is that the desired accuracy can be achieved by adjusting the number of waypoints. Subsequently, the path planning problem can be taken as the optimization of the waypoint coordinate.

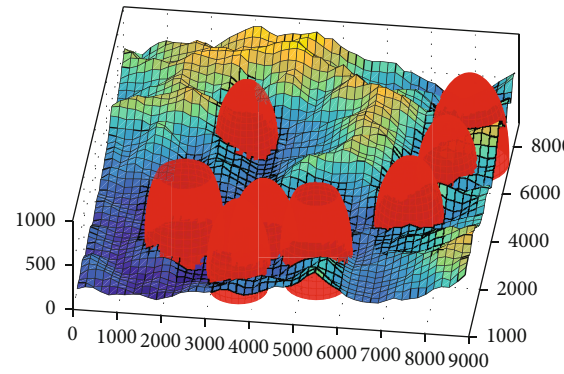


Figure 1: The environment with the terrain and the danger zones.

Define $P = \{S, W_1, W_2, \cdots, W_i, W_{i+1}, \cdots, W_{N_w}, T\}$ as a trajectory in the flying space, where S is the starting point, T is the target point, and $W_1, W_2, \cdots, W_i, W_{i+1}, \cdots, W_n$ are the waypoints between the start point and the target point. The segment composed by two adjacent waypoints W_i, W_{i+1} is divided into N_d parts equally, as shown in Figure 2. Suppose the set of the dividing points is $D = \{D_{11}, D_{12}, \cdots, D_{1N_d}, D_{21}, D_{22}, \cdots, D_{2N_d}, \cdots, D_{N_w1}, D_{N_w2}, \cdots, D_{N_wN_d}\}$.

The coordinates of dividing points are calculated as:

$$\begin{cases} x_{ik} = x_{i-1} + k \cdot (x_i - x_{i-1})/N_d, \\ y_{ik} = y_{i-1} + k \cdot (y_i - y_{i-1})/N_d, \\ z_{ik} = z_{i-1} + k \cdot (z_i - z_{i-1})/N_d, \end{cases} \quad (4)$$

where (x_{ik}, y_{ik}, z_{ik}) is the coordinate of the k-th division point corresponding to the i-th waypoint. $(x_{i-1}, y_{i-1}, z_{i-1})$ and (x_i, y_i, z_i) are the $(i-1)$-th waypoint and the i-th waypoint, respectively.

3. Cost Function of Path Planning for Fixed-Wing

For the UAV path planning problem, the cost function is a series of optimization criteria and constraints to evaluate the quality of the flight path. The smaller the cost function fitness value is, the better the quality of the flight path is. To determine the cost function, it is necessary to consider that all the factors can affect the route performance, such as path length, flight safety altitude, threat probability, UAV dynamic constraint, and environment constraints. In general, the cost function mainly includes two parts: the objective function and the constraint function. The cost function F of path planning for fixed-wing UAV is the sum of

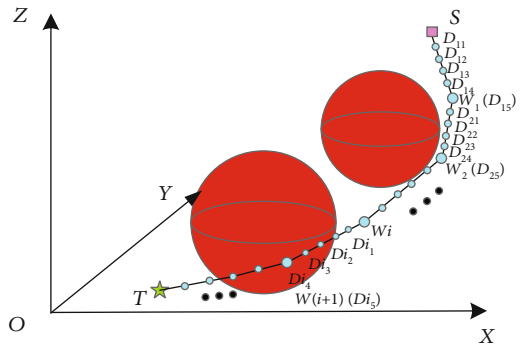


Figure 2: The flight trajectory in 3D space.

the objective function and the constraint function, which is given as

$$F = \sum_{k=1}^{3} J_k + \sum_{n=1}^{3} C_n, \tag{5}$$

where $\sum_{k=1}^{3} J_k$ is the sum of objective function, $\sum_{k=1}^{3} C_n$ is the sum of the constraint function.

3.1. Objective Function. The objective function enables the UAV to obtain the maximum profit under the premise of satisfying the constraints. Taking the route length, the radar threat, and flight height into account, the objective cost is calculated as follows

$$\sum_{k=1}^{3} J_k = J_{\text{length}} + J_{\text{altitude}} + J_{\text{threat}}, \tag{6}$$

where J_{length}, J_{altitude}, and J_{threat} represent the costs of path length, the threat, and the height, respectively.

(1) Route length cost

In general, shorter routes can save more fuel consumption. To describe the length cost more accurately, path length ratio (PLR) is utilized to measure the route length cost J_{length}, which is given as follows

$$\begin{aligned} J_{\text{length}} &= \frac{L_{\text{traj}}}{L_{\text{ST}}} = \frac{\sum_{i=1}^{N_w+1} \left\| \overrightarrow{W_{i-1} W_i} \right\|}{L_{\text{ST}}} \\ &= \frac{\sum_{i=1}^{N_w+1} \sqrt{(x_i - x_{i-1})^2 + (y_i - x_{i-1})^2 + (z_i - z_{i-1})^2}}{\sqrt{(x_T - x_S)^2 + (y_T - x_S)^2 + (z_T - z_S)^2}}, \end{aligned} \tag{7}$$

where L_{ST} is the length of the straight line connecting the starting point S and the destination T. L_{traj} is the length of the actual route. $\|\cdot\|$ denotes the Euclidean distance of a vector. When the actual route length is approximately closer to the length of the straight line, a shorter route will be obtained. The range of J_{length} is given as follows

$$J_{\text{length}} \in [1, 1.5]. \tag{8}$$

The length of the path with good performance is considered within the scope of 1.5 times the distance between start and destination according to [15]. Hence, the value of 1.5 is the preference of feasible PLR.

(2) Flight height cost

A lower-height path can reduce the risk of being detected by radar and strengthen the threat to the enemy on the ground for flying missions. The term J_{altitude} is calculated as follows

$$J_{\text{altitude}} = \sum_{i=1}^{N_w+1} \frac{A_{\text{route}.i} - Z_{\min}}{Z_{\max} - Z_{\min}}, \tag{9}$$

where $A_{\text{route}.i}$ represents the average flight altitude of i-th route segment, $Z_{\min}$ and $Z_{\max}$ are the lower and higher bounds of the flight height in mission space, respectively.

Therefore,

$$J_{\text{altitude}} \in [0, 1]. \tag{10}$$

(3) Threat cost

During the process of UAV flying, it is essential to avoid into the detection range of the radar, where it may encounter the risk of being discovered or being attacked. The threat cost is computed according to the route length which goes into the threat sphere, which is calculated by

$$J_{\text{threat}} = \frac{\sum_{i=1}^{N_w+1} L_{\text{inside}.i}}{\sum_{k=1}^{N} D_k}, \tag{11}$$

where L_{inside} is the length of i-th segment that goes inside the threat circle, D_k indicates the diameter of k-th threat, and N is the number of threat sources. It is noted that J_{threat} is limited to $[0, 1]$.

3.2. Constraint Function. The constraint function to evaluate the candidate route should take the environment constraint and UAV dynamic constraints into account. The environment constraint focuses on the terrain of the flight altitude constraint. UAV dynamic constraints mainly refer to the turning-angle constraint and the climb/dive angle constraint. The constraint function $\sum_{n=1}^{3} C_n$ is descripted as

$$\sum_{n=1}^{3} C_n = C_{\text{collision}} + C_{\text{turn}} + C_{\text{slope}}, \tag{12}$$

where $C_{\text{collision}}$ is introduced to penalize the candidate routes colliding with the terrain, C_{turn} is introduced to penalize the route with a turning angle beyond the maximum, and C_{slope} is introduced to penalize the route with the climbing/diving

angle beyond the constraint. The constraint function can be depicted as follows in detail.

(1) Terrain constraint

If the flight height is lower than the terrain, then, UAV will collide with the terrain. So the term $C_{\text{collision}}$ associated with the terrain collision is described as follows

$$C_{\text{collision}} = \begin{cases} 0, & \text{if} \quad \sum_{i=1}^{N_w+1} L_{\text{under}.i} = 0, \\ P_e + \sum_{i=1}^{N_w+1} \dfrac{L_{\text{under}.i}}{L_{\text{traj}.i}}, & \text{if} \quad \sum_{i=1}^{N_w+1} L_{\text{under}.i} > 0, \end{cases} \tag{13}$$

where $L_{\text{under}.i}$ is the length of the i-th segment in a route which located below the altitude of the terrain, $L_{\text{traj}.i}$ is the length of the i-th actual route segment. Considering the above objective optimization value, the penalty constant P_e is selected to 3.5.

(2) Turning angle constraint

In view of the physical characteristics of UAV, the turning angle for UAV is required to be less than or equal to the maximum turning angle. The turning angle constraint function C_{turning} can be expressed by

$$C_{\text{turning}} = \begin{cases} 0, & \text{if} \quad \sum_{i=1}^{N_w} a_i = 0 \\ e + \dfrac{\sum_{i=1}^{N_w} a_i}{N_w} & \text{if} \quad \sum_{i=1}^{N_w} a_i > 0 \end{cases} \quad \text{with } a_i = \begin{cases} 0, & \text{if} \quad \theta_i < \theta_{\max}, \\ 1, & \text{if} \quad \theta_i > \theta_{\max}, \end{cases} \tag{14}$$

where $\theta_{\max}$ is the maximum turning angle, θ_i is the corresponding turning angle of the i-th waypoint. The turning angle θ_i is calculated according to

$$\theta_i = \arccos\left(\frac{(x_{i+1} - x_i, y_{i+1} - y_i) \cdot (x_i - x_{i-1}, y_i - y_{i-1})^T}{\|(x_{i+1} - x_i, y_{i+1} - y_i) \cdot (x_i - x_{i-1}, y_i - y_{i-1})\|}\right). \tag{15}$$

(3) Slope angle constraint

The slope angle for UAV at each waypoint is limited into the range between the maximum slope angle and the minimum slope angle. The slope angle constraint referring to the vertical direction is calculated as

$$C_{\text{slope}} = \begin{cases} 0, & \text{if} \quad \sum_{i=1}^{N_w+1} b_i = 0 \\ e + \dfrac{\sum_{i=1}^{N_w+1} b_i}{N_w + 1} & \text{if} \quad \sum_{i=1}^{N_w+1} b_i > 0 \end{cases} \quad \text{with } b_i = \begin{cases} 0, & \text{if} \quad \varphi_{\min} < \varphi_i < \varphi_{\max}, \\ 1, & \text{else}, \end{cases} \tag{16}$$

where $\varphi_{\min}$ and $\varphi_{\max}$ are, respectively, the minimum and maximum slope angle, φ_i is the corresponding slope angle of the i-th waypoint, which can be calculated as

$$\theta_i = \arctan\left(\frac{z_i - z_{i-1}}{\|(x_i - x_{i-1}, y_i - y_{i-1})\|}\right). \tag{17}$$

4. Improved PSO Algorithm

4.1. PSO Algorithm. PSO is a typical search algorithm based on group cooperation, which comes from the basic concept of the study on simulating the foraging behavior of birds. In the particle swarm optimization algorithm, each bird in the swarm is regarded as a particle. The basic idea of PSO is to find the optimal solution through cooperation and information sharing among the individuals in the group. The process is simplified as follows. First, a group of random particles are generated at the initialization stage, and then to search the optimal solution through iteration. Each particle updates its position and velocity by tracking two extreme values (*pbest* and *gbest*) at one iteration. In n-dimensional search space, a population of m particles is $X = \{x_1, x_2, \cdots, x_i, \cdots, x_m\}$, the position of the i-th particle is $x_i = (x_{i1}, x_{i2}, \cdots, x_{\text{in}})^T$, and its velocity is $v_i = (v_{i1}, v_{i2}, \cdots, v_{\text{in}})^T$. The individual best solution of the population is $pbest_i = (pbest_{i1}, pbest_{i2}, \cdots, pbest_{\text{in}})^T$, and the global best solution is $gbest_i = (gbest_{i1}, gbest_{i2}, \cdots, gbest_{\text{in}})^T$. The velocity and position of each particle is updated according to the following formula

$$\begin{aligned} v_{id}(t+1) &= w \cdot v_{id}(t) + c_1 r_1 (pbest_{id}(t) - x_{id}(t)) \\ &\quad + c_2 r_2 (gbest_{id}(t) - x_{id}(t)), \\ x_{id}(t+1) &= x_{id}(t) + v_{id}(t+1), \end{aligned} \tag{18}$$

where m is the population size, t is the current iteration, r_1 and r_2 are random numbers in [0,1]. c_1 and c_2 are, respectively, the personal influence parameter and the social influence parameter, and w is the inertia coefficient.

4.2. Improved PSO Algorithm Based on Dynamic DC Strategy. When the number of obstacles, the threat sources increase, or the environment becomes complex, the number of waypoints should increase accordingly. However, the larger the dimension of the particle, and the worse the search ability of the intelligent algorithms becomes. Therefore, a large amount of waypoints will result in failure of planning. The planners based on intelligent optimization algorithms may be difficult to explore a feasible route in a mission space. An efficient way to solve this problem is to simplify the original problem into lower dimensions, where existing intelligent optimization algorithms can well suit. In consideration of computational

accuracy and efficiency, dynamic DC strategy is introduced in PSO to find an optimal route when the number of the waypoints increases. According to this idea, the grouping strategy based on dynamic dimension is adopted to divide multiple waypoints into different groups. The number of the group in DC strategy is adjusted according to the cost function value.

A new subcost function is established to evaluate the quality of subpath. The group cost function is defined as

$$F_g = \begin{cases} \sum_{k=1}^{3} \sum_{b(g-1)-2}^{bg+2} J_{gk} + \sum_{n=1}^{3} \sum_{b(g-1)-2}^{bg+2} C_{gn}, & \text{if } b(g-1) > 2 \& bg+2 \le N_w, \\ \sum_{k=1}^{3} \sum_{S}^{bg+2} J_{gk} + \sum_{n=1}^{3} \sum_{S}^{bg+2} C_{gn}, & \text{if } b(g-1) \le 2 \& bg+2 \le N_w, \\ \sum_{k=1}^{3} \sum_{S}^{T} J_{gk} + \sum_{n=1}^{3} \sum_{S}^{T} C_{gn}, & \text{if } b(g-1) \le 2 \& bg+2 > N_w, \\ \sum_{k=1}^{3} \sum_{b(g-1)-2}^{T} J_{gk} + \sum_{n=1}^{3} \sum_{b(g-1)-2}^{T} C_{gn}, & \text{if } b(g-1) > 2 \& bg+2 > N_w, \end{cases} \tag{19}$$

where b is the number of the waypoints in each group, J_{gk} is the objective function, and C_{gn} is the constrain function. J_{gk} and C_{gn} can be denoted as

$$\sum_{gs}^{gt} J_{gk} = J_{\text{length}.g} + J_{\text{altitude}.g} + J_{\text{threat}.g} = \frac{\sum_{i=gs}^{gt} \| W_{i\text{-}1} W_i \|}{L_{ST}} + \sum_{i=gs}^{gt} \frac{A_{\text{route}.i} - Z_{\min}}{Z_{\max} - Z_{\min}} + \frac{\sum_{i=gs}^{gt} L_{\text{inside}.i}}{\sum_{k=1}^{n} D_k}, \tag{20}$$

$$\sum_{gs}^{gt} C_{gn} = \begin{cases} 0, & \text{if } \sum_{i=gs}^{gt} L_{\text{under}.i} = 0 \& \sum_{i=gs}^{gt} a_i = 0 \& \sum_{i=gs}^{gt} b_i = 0, \\ P_e + \sum_{i=gs}^{gt} \frac{L_{\text{under}.i}}{L_{\text{traj}.i}} + P_e + \frac{\sum_{i=gs}^{gt} a_i}{gt - gs + 1} + P_e + \frac{\sum_{i=gs}^{gt} b_i}{gt - gs + 1}, & \text{if } \sum_{i=gs}^{gt} L_{\text{under}.i} > 0 \& \sum_{i=gs}^{gt} a_i > 0 \& \sum_{i=gs}^{gt} b_i > 0, \\ P_e + \sum_{i=gs}^{gt} \frac{L_{\text{under}.i}}{L_{\text{traj}.i}} + P_e + \frac{\sum_{i=gs}^{gt} a_i}{gt - gs + 1}, & \text{if } \sum_{i=gs}^{gt} L_{\text{under}.i} > 0 \& \sum_{i=gs}^{gt} a_i > 0 \& \sum_{i=gs}^{gt} b_i = 0, \\ P_e + \frac{\sum_{i=gs}^{gt} a_i}{gt - gs + 1} + P_e + \frac{\sum_{i=gs}^{gt} b_i}{gt - gs + 1}, & \text{if } \sum_{i=gs}^{gt} L_{\text{under}.i} = 0 \& \sum_{i=gs}^{gt} a_i > 0 \& \sum_{i=gs}^{gt} b_i > 0, \\ P_e + \sum_{i=gs}^{gt} \frac{L_{\text{under}.i}}{L_{\text{traj}.i}} + P_e + \frac{\sum_{i=gs}^{gt} b_i}{gt - gs + 1}, & \text{if } \sum_{i=gs}^{gt} L_{\text{under}.i} > 0 \& \sum_{i=gs}^{gt} a_i = 0 \& \sum_{i=gs}^{gt} b_i > 0, \\ P_e + \sum_{i=gs}^{gt} \frac{L_{\text{under}.i}}{L_{\text{traj}.i}}, & \text{if } \sum_{i=gs}^{gt} L_{\text{under}.i} > 0 \& \sum_{i=gs}^{gt} a_i = 0 \& \sum_{i=gs}^{gt} b_i = 0, \\ P_e + \frac{\sum_{i=gs}^{gt} a_i}{gt - gs + 1}, & \text{if } \sum_{i=gs}^{gt} L_{\text{under}.i} = 0 \& \sum_{i=gs}^{gt} a_i > 0 \& \sum_{i=gs}^{gt} b_i = 0, \\ P_e + \frac{\sum_{i=gs}^{gt} b_i}{gt - gs + 1}, & \text{if } \sum_{i=gs}^{gt} L_{\text{under}.i} = 0 \& \sum_{i=gs}^{gt} a_i = 0 \& \sum_{i=gs}^{gt} b_i > 0. \end{cases} \tag{21}$$

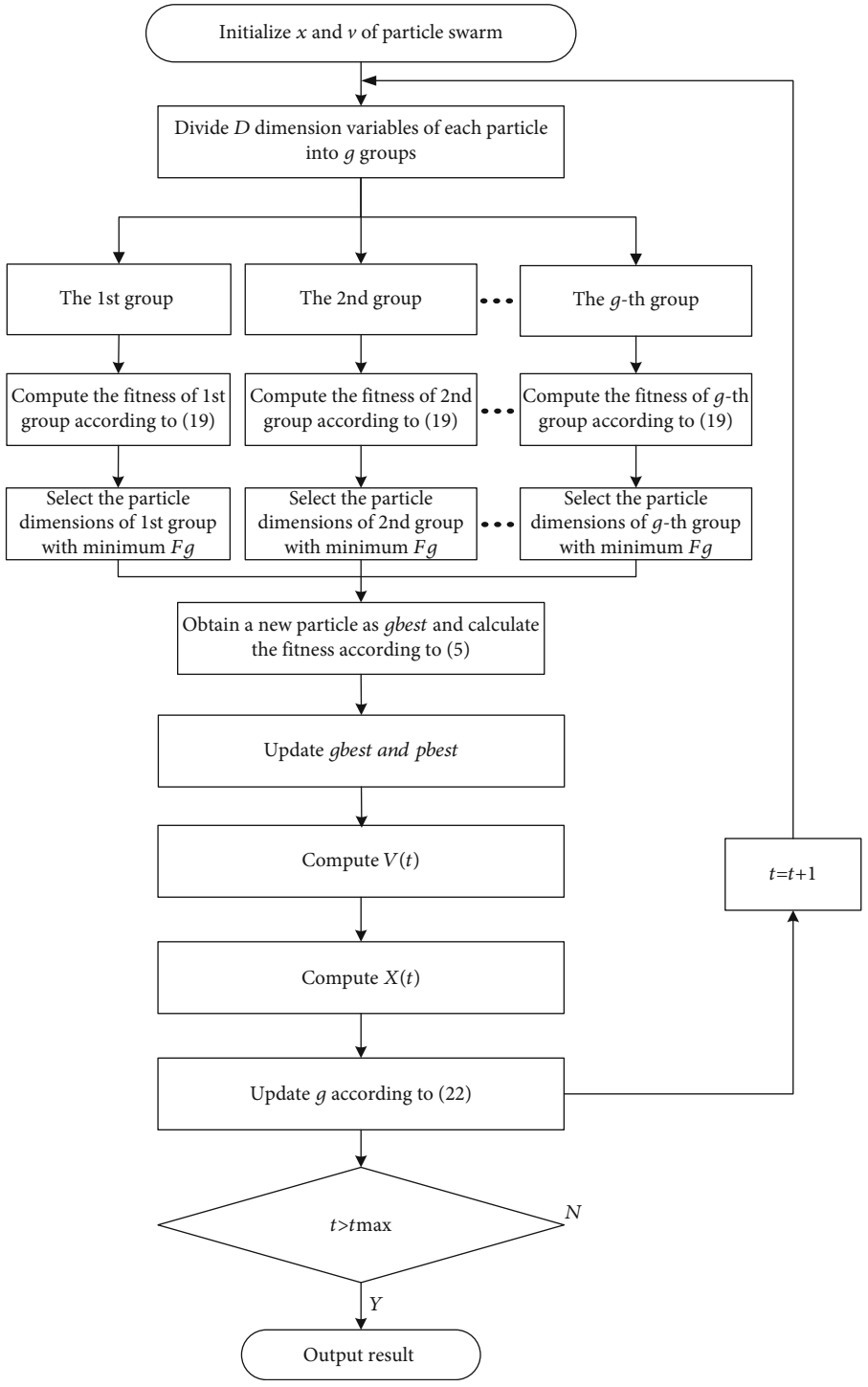


Figure 3: The flow chart of PSO with divide-and-conquer (DC) strategy.

The individual best solution of each group is selected by comparing the group cost function F_g of previous parent and offspring. The global best solution based on dynamic DC strategy is generated by choosing the best group of all the groups according to F_g value. In order to explain the strategy, the flow chart of PSO with DC strategy is given in Figure 3. At the starting phase, the positions and speeds of m particles are initialized, and then each particle is divided into g groups based on DC strategy, so each group contains D/g dimensions. D/g dimensions of each particle form a group, which can be evaluated by the group cost function F_g. According to the result of the group cost function value, the groups with minimum group function fitness will be found and compose the global best particle. The parallel

search based on DC strategy is developed to find the whole particle swarm and complete an iteration of search. During the iteration of path search, the group number g is adjusted dynamically by

$$\begin{cases} g(t+1) = N\left(g(t), \sigma^2\right), & \text{if} \quad F_{g\text{-best}}(t) < F_{g\text{-best}}(t-1), \\ g(t+1) = \text{randint}(a, b), & \text{if} \quad \text{else}, \end{cases} \tag{22}$$

where $N(g(t), \sigma^2)$ is the normal distribution with mean $g(t)$ and standard deviation σ^2. randint(a, b) represents an integer chosen within the range of $[a, b]$, $a \geq 1$, $b \leq D$.

Thus, the individual best solution of each particle is selected to participate in the next iteration operation.

4.3. An Improved PSO Algorithm Based on A ∗ Algorithm. In order to improve the convergence accuracy and convergence speed, $A*$ algorithm is introduced into PSO. The route obtained by $A*$ search is regarded as a particle to optimize the performance of the DCPSO algorithm.

$A*$ algorithm is an effective and direct search algorithm in static environment, which determines search direction from the starting point to the surrounding through the heuristic function. In $A*$ algorithm, the open list and close list are used to realize the node expansion and the best choice in the process of path planning. The function of open list is to save the extended nodes which needed to be checked in the search process. The node with the smallest value is selected as the current node and put into close list, then all the neighboring nodes of the current node is added to the open list according to the neighbor node rule. The process is repeated until a path is found from the start point to the end point. During the search process, because each node on the path has the minimum cost, the cost of the path is the least.

The heuristic function of $A*$ algorithm is aimed at evaluating the equality of the expandable nodes. The heuristic function is defined as follows

$$f(n) = g(n) + h(n), \tag{23}$$

where n is the node to be expanded; $g(n)$ is the actual cost from the initial node to the node n in the space; $h(n)$ is the cost from node n to the target node.

For fix-wing UAV path planning problem, the heuristic function is denoted as

$$f(n) = \sum_{k=1}^{3} f_k(n) + \sum_{v=1}^{3} fC_V(n), \tag{24}$$

where $\sum_{k=1}^{3} f_k(n)$ is cost function of node n, $\sum_{v=1}^{3} fC_v(n)$ is the constraints of node n.

The cost function of node n is given by

$$\sum_{k=1}^{3} f_k(n) = f_{\text{length}}(n) + f_{\text{altitude}}(n) + f_{\text{threat}}(n), \tag{25}$$

where $f_{\text{length}}(n)$, $f_{\text{altitude}}(n)$, and $f_{\text{threat}}(n)$ are the objective functions of the length, the altitude, and the risk of radar detection for node n, respectively.

(1) Length cost of node n

For node n, $f_{\text{length}}(n)$ can be given as

$$\begin{aligned} f_{\text{length}}(n) &= \frac{\left\|\overrightarrow{(n-1)n}\right\| + \left\|\overrightarrow{n\text{T}}\right\|}{\left\|\overrightarrow{(n-1)\text{T}}\right\|} \\ &= \frac{\|x_n - x_{n-1}, y_n - y_{n-1}, z_n - z_{n-1}\| + \|x_T - x_n, y_T - y_n, z_T - z_n\|}{\|x_{\text{T}} - x_{n-1}, y_{\text{T}} - y_{n-1}, z_{\text{T}} - z_{n-1}\|}, \end{aligned} \tag{26}$$

where T is the goal.

Therefore,

$$f_{\text{length}}(n) \in [1, 1.5]. \tag{27}$$

(2) Flight altitude cost of node n

For node n, $f_{\text{altitude}}(n)$ is described as

$$f_{\text{altitude}}(n) = \frac{A(n) - Z_{\min}}{Z_{\max} - Z_{\min}}, \tag{28}$$

where $A(n)$ is the average flight of the segment between node $n-1$ and node n.

Therefore,

$$f_{\text{altitude}}(n) \in [0, 1]. \tag{29}$$

(3) Threat cost of node n

For node n, $f_{\text{threat}}(n)$ can be calculated as

$$f_{\text{threat}}(n) = \frac{dL_{\text{inside}}}{\sum_{k=1}^{N} D_k}, \tag{30}$$

where dL_{inside} is the length of the segment between node n and node $n-1$ that go through the threat zones. Therefore,

$$f_{\text{threat}}(n) \in [0, 1]. \tag{31}$$

The Constraint Expression of node n

$$\sum_{v=1}^{3} fC_v(n) = fC_{\text{terrain}}(n) + fC_{\text{turning}}(n) + fC_{\text{slope}}(n). \quad (32)$$

(1) Terrain constraint for node n

For node n, $fC_{\text{collision}}(n)$ can be given as

$$fC_{\text{terrain}}(n) = \frac{\sum_{k=1}^{N_d} s1(n)}{N_d},$$
$$\text{with } s1(n) = \begin{cases} 1, & \text{if } z_{n-1,k} \le \text{terrain}\left(x_{n-1,k}, y_{n-1,k}\right), \\ 0, & \text{else}, \end{cases} \quad (33)$$

where $(x_{n-1,k}, y_{n-1,k}, z_{n-1,k})$ is the k-th division point coordinate of the node $n-1$, $z_{\text{terrain}}(x_{n-1,k}, y_{n-1,k})$ is the terrain altitude of the corresponding coordinate $(x_{n-1,k}, y_{n-1,k})$ on the map.

(1) Turning angle constraint for node n

For node n, $fC_{\text{turning}}(n)$ is calculated by

$$fC_{\text{turning}}(n) = \begin{cases} C, & \text{if } \theta_n > \theta_{\max}, \\ 0, & \text{else}, \end{cases} \quad (34)$$

where the constant C is used to penalize the segment with the turning angle more than the maximum turning angle.

(2) Slope constraint

For node n, $fC_{\text{slope}}(n)$ can be given by

$$fC_{\text{slope}}(n) = \begin{cases} 0, & \text{if} \varphi_{\min} \le \varphi_n \le \varphi_{\max}, \\ C, & \text{else}. \end{cases} \quad (35)$$

4.4. Improved PSO Algorithm Based on A ∗ Algorithm. The process of improved PSO algorithm based on $A*$ algorithm is described in Figure 4.

From the starting point S, put S into the open list. The first waypoints of $m-1$ particles are considered as the adjacent nodes of the starting point S. The adjacent nodes with S are checked to join the open list. Set the starting point as parent node and remove S from the open list to the close list. The first waypoint of $m-1$ particles with minimum fitness will be searched and add it into the close list. Then, this waypoint in close list is seen as the parent of the next node, and the adjacent nodes with the first waypoint are the second waypoints of $m-1$ particles. These nodes are put into the open list, repeat the selection process based on the $A*$ heuristic function value. Until the last waypoint, a whole route is generated by $A*$ algorithm at one iteration in PSO. The

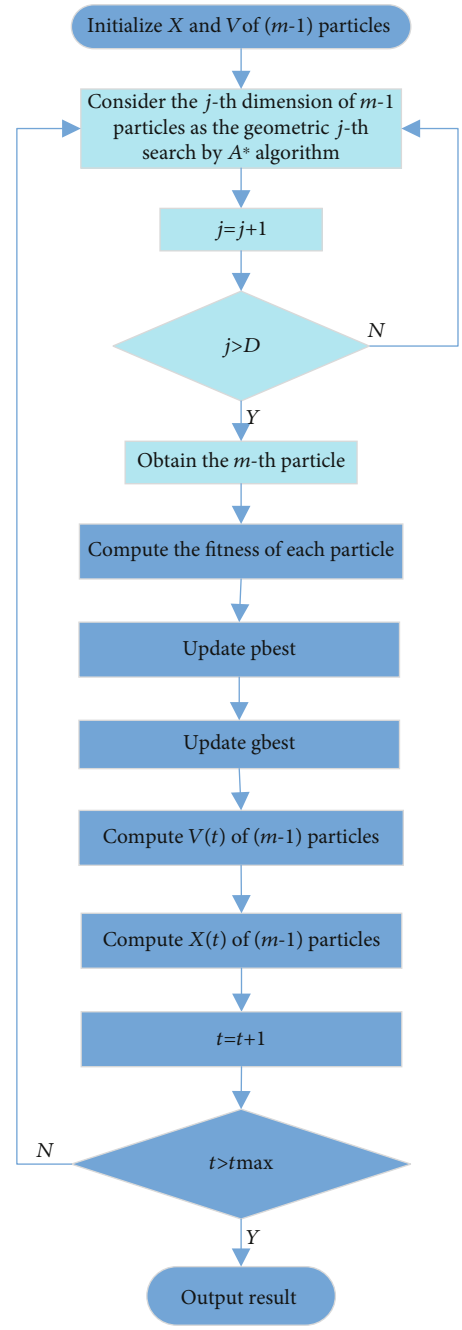


Figure 4: The flow chart of PSO with $A*$ algorithm.

waypoints in the close list generate m-th particle. The path found by $A*$ algorithm is seen as a particle in the particle swarm. The $m-1$ particles in PSO will be updated according to the evolution equation. The $A*$ algorithm of the new generated particles will be continued. The new route obtained by $A*$ algorithm from all the particles can optimize the equality of the particles, enrich the diversity of particle swarm, and enhance the convergence speed.

4.5. The Steps of DCA∗PSO Algorithm. The steps of the DCA∗PSO algorithm based on DC strategy and A∗ algorithm are described as follows.

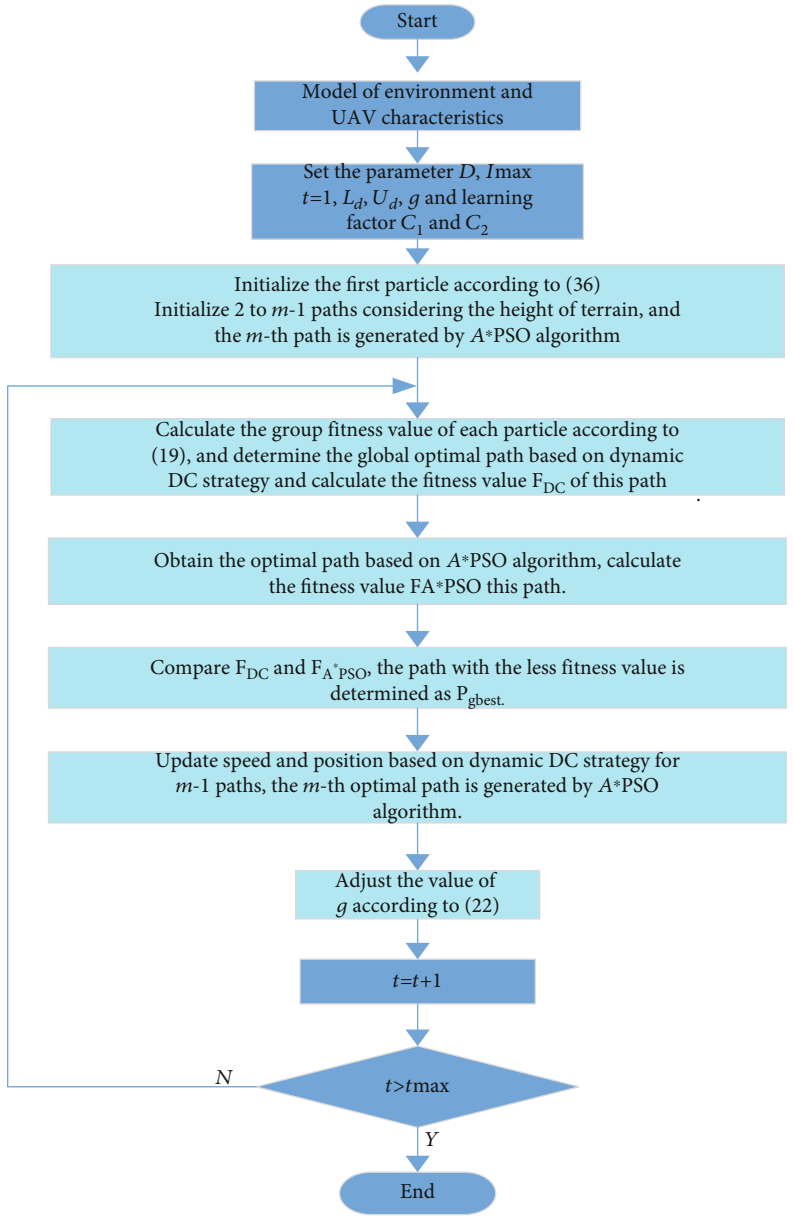


Figure 5: The flow chart of the DCA∗PSO algorithm.

Step 1. Set the parameters and initialize each particle.

Set population size m, the maximum number of iterations, particle dimension, the upper and lower bounds of velocity and position, and learning factor c_1 and c_2. Randomly generate the initial position and the initial velocity of each particle.

Step 2. Evaluate the particle based on group cost function.

Generate the group number g of DC strategy according to (22). It means that the dimensions are divided into g groups. Calculate the group cost function value of each group. If the value is better than that of the current value, this value is regarded as the new best value of the individual. The global best solution based on dynamic DC strategy is obtained by combining the groups with the best group cost function value.

Step 3. Find the global best solution.

A path is searched by $A*$ PSO algorithm. Calculate the fitness of this path and compare with that of the global best solution based on DC strategy. Select the path with the less fitness value as the global best path.

Step 4. Update speed and position.

Update the velocity and position in each group according to (18).

Table 1: The statistical results of three cases.

Item	N_w	Min cost	Mean cost	Std. dev.	G_c	$\widetilde{G_c}$	FR (%)
Case I	5	1.4851	1.6951	0.5470	1	6	96.67
	7	1.5056	1.6799	0.1374	1	6	100
	10	1.5352	1.6443	0.0836	1	5	100
	15	1.5432	1.7130	0.1322	5	10	100
	20	1.5315	1.6243	0.0739	5	12	100
	25	1.6022	2.0725	0.9112	9	23	93.33
Case II	5	2.2425	3.3362	1.2896	4	15	70
	7	1.6101	2.2201	0.4413	3	15	96.67
	10	1.6249	1.8212	0.1102	2	6	100
	15	1.5371	1.7924	0.2645	5	16	100
	20	1.5255	1.7033	0.6325	4	12	96.67
	25	1.5985	1.9863	1.0627	4	13	90
Case III	50	1.4152	1.4785	0.0877	4	10	100
	60	1.3522	1.3913	0.0343	4	9	100
	70	1.4252	1.4492	0.0420	5	12	100
	80	1.3631	1.7131	0.9523	5	13	90
	90	1.3813	1.8634	0.9499	5	14	90
	100	1.3916	1.7671	0.9376	7	24	90

Step 5. Determine the end condition.

Determine whether the end conditions are met. If the end condition is met, the global optimum value and position of the particle are saved. Otherwise, turn to Step 2.

5. Optimization Model of Path Planning

The proposed DCA∗PSO algorithm, which has stronger optimization performance, is used to solve the optimization model of path planning for UAV. The detailed process is shown in Figure 5. The steps are described as follows:

Step 1. Establish the environment model.

The environment information is established, mainly including the planning space, the obstacle information, the center, and the size of the threat sources.

Step 2. Initialize the parameters.

Initialize the parameters of the DCA∗PSO algorithm. The parameters include the population size, the maximum number of iteration, the number of waypoints, the maximum number of iterations $I_{\max}$, the upper and lower bounds of the velocity of the position, and the learning factor and inertial coefficient.

Step 3. Initialize the route.

For the first particle, x, y, z coordinates of the waypoints are initialized as

$$\begin{cases} x_i = i \times \dfrac{x_T - x_S}{N_w + 1}, \\ y_i = i \times \dfrac{y_T - y_S}{N_w + 1}, \\ z_i = Z_{\text{terrain}}(x_i, y_i) + D_{\text{safe}}. \end{cases} \tag{36}$$

From the second particle to $(m-1)$-th particle, x coordinates of the waypoints are given by evenly distributed on x-axis of the coordinate from the starting point S to the goal for $m-1$ particles, and y coordinates of the waypoints are selected randomly within the bounds for $m-1$ particles. Due to the altitude of the terrain information is known before the planning, take full advantage of the altitude of the terrain to optimize the initial routes. So z-coordinate of i-th waypoint is given as follows

$$z_i = Z_{\text{terrain}}(x_i, y_i) + D_{\text{safe}}, \tag{37}$$

where $Z_{\text{terrain}}(x_i, y_i)$ represents the corresponding terrain altitude of the coordinate (x_i, y_i), and D_{safe} denotes a safety reference. The m-th path is obtained by using $A*$PSO algorithm according to the cost function of $A*$ algorithm.

Step 4. Evaluate the particle.

Calculate the group fitness value of each group. If the new fitness value is better than that of the current value, this value is regarded as the new best fitness value of individual. The path based on DC strategy is found by combining the group with the best group fitness value in all groups. Calculate the fitness value $F_{A*\text{PSO}}$ of the path based on $A*$PSO algorithm. Compare F_{DC} and $F_{A*\text{PSO}}$, then, the route with a smaller fitness value is set as the current global best value F_{gbest}. This route with global best fitness is taken in regard as the global best route.

Step 5. Update the velocity and position.

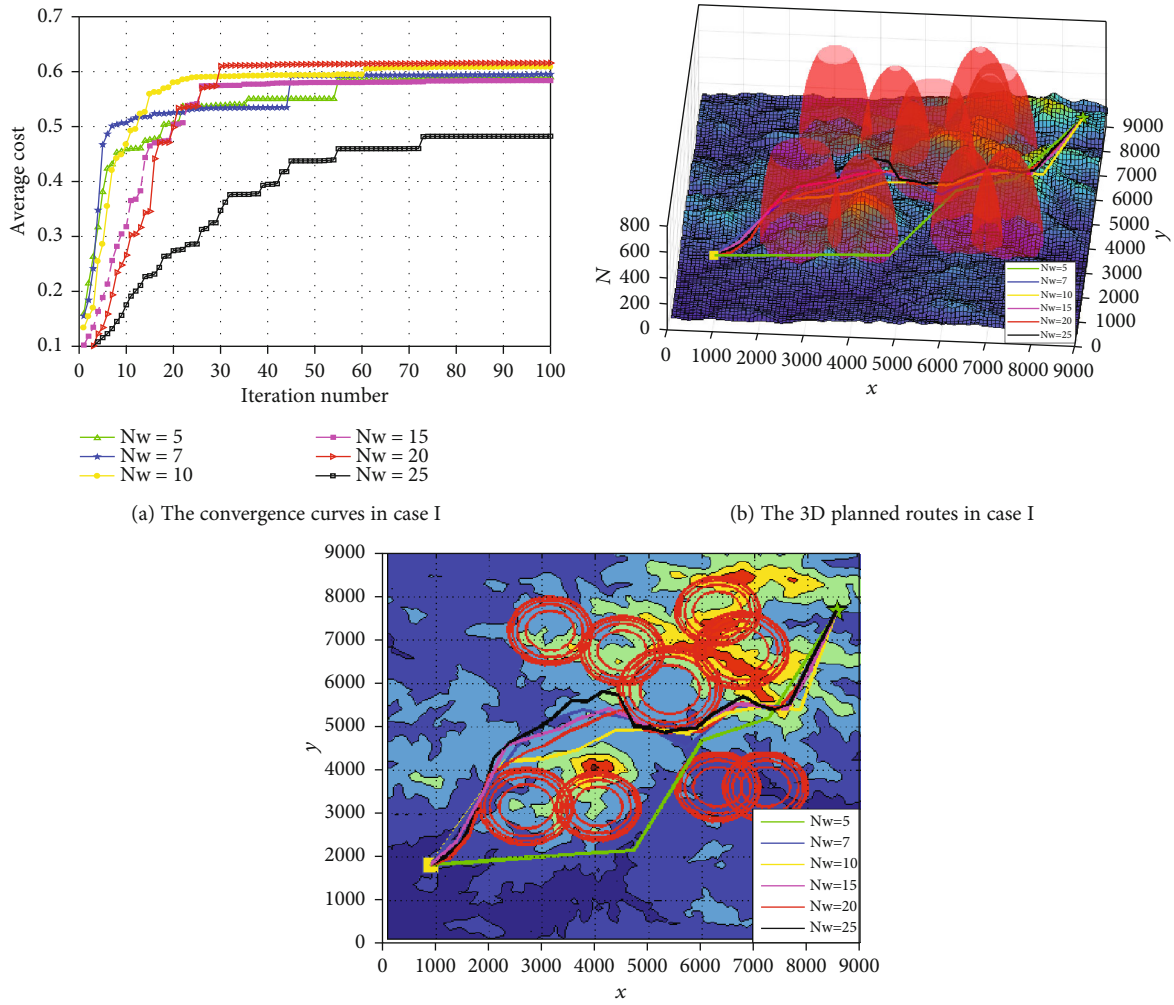
Update the velocity and the position of $m-1$ particles by (18) from the first group to g-th based on DC strategy. $A*$ PSO algorithm is adopted to obtain m-th optimal path.

Step 6. Determine the end condition.

Determine whether the termination condition is met, then output the global optimum value and the corresponding route. Otherwise, turn to Step 4.

6. The Data Simulation and Analysis

In order to demonstrate the performances of the DCA∗PSO algorithm for the path planning problem, a series of experiments with the realistic environment maps [37] were implemented on a PC with Intel Core (TM) i7-9700 CPU



(a) The convergence curves in case I

(b) The 3D planned routes in case I

(c) The contour of planned routes in case I

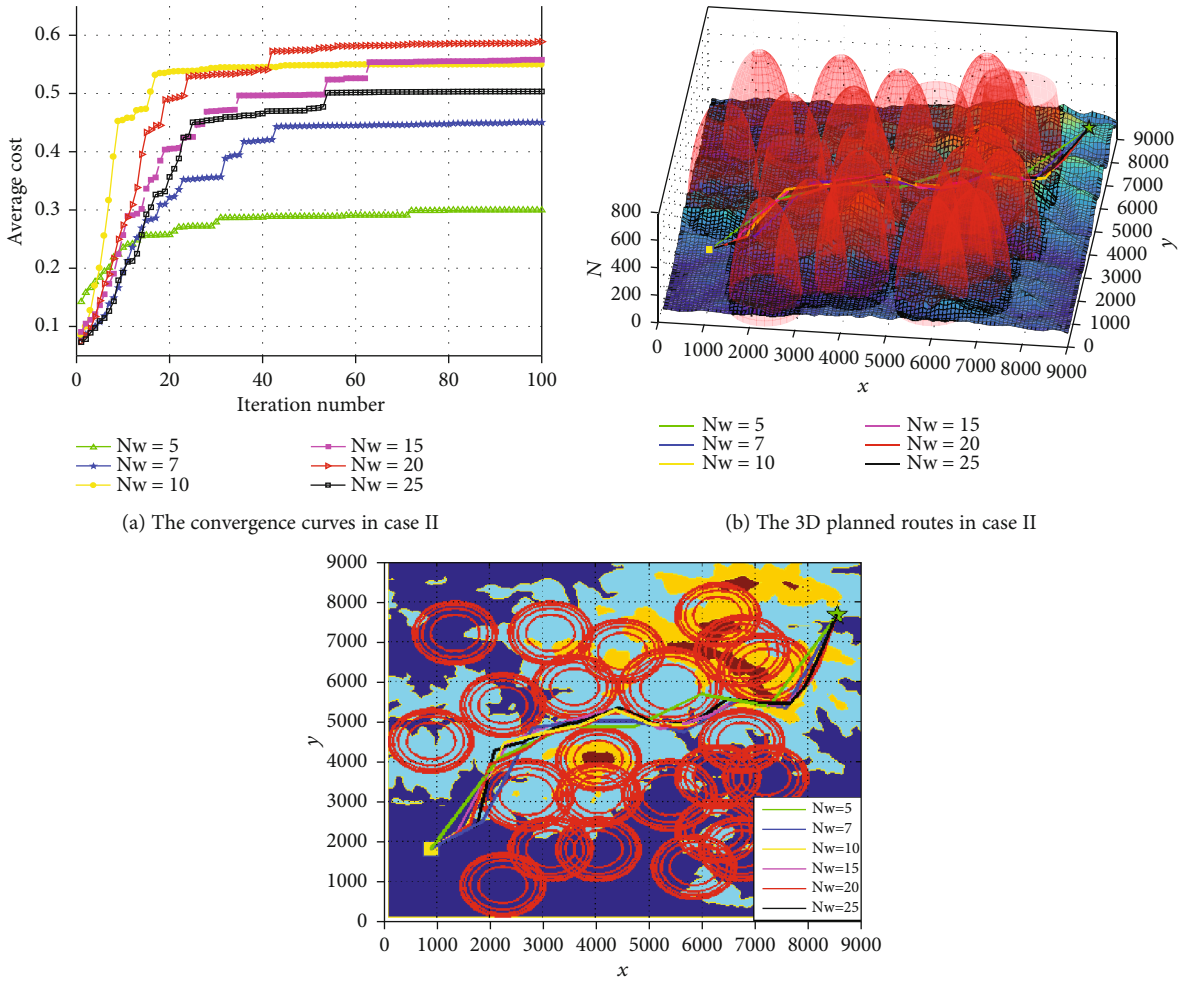
FIGURE 6: The comparison of results with different numbers in case I.

@3.00GHz running Windows10. Each group of experiments was conducted 30 times independently, and the main data of the results were recorded in table for the further analysis.

6.1. The Analysis of Different Waypoint Number. The number of the waypoints is an important parameter that affects the performance of the algorithm, which has a very close relationship with accuracy of the route and algorithm computational efficiency. With the number increase of waypoints, the search difficulty of the algorithm will increase rapidly. The following experiments were implemented for testing the performance of DCA∗PSO algorithm when the number of the waypoints changes. The parameters of DCA∗PSO are set as follows: population size $N = 100$, the maximum iteration $t_{\max} = 100$, the personal influence $c_1 = 1.5$, and the social influence parameter $c_2 = 1.5$. The number of the dividing point is $N_d = 6$. Three cases are utilized to test: case I is a simple environment with nine threat sources, case II is a complicated model with 24 threat sources, and case III is a larger map environment with 32 threat sources. The statistical results of the three cases are listed in Table 1, which include the best cost, mean cost, standard deviation of the fitness value, G_c, $\widetilde{G_c}$, and FR during 30 independent runs. Among these, G_c and $\widetilde{G_c}$ denote the iteration number when the algorithm finds the feasible flight route, which reflects the convergence speed and the efficiency of the algorithm. The G_c is the smallest iteration number in 30 runs, and $\widetilde{G_c}$ is the mean iteration number in 30 runs. FR means the percentage of the feasible routes in 30 runs, which is the ratio of the runs satisfying the constraints in the total runs. The best results are marked with boldface for clarity. It is only the feasible routes can be used for the statistics of column 6-7. The typical 3D displays and contour routes with different numbers of the waypoints in case I, case II, and case III during 30 runs are plotted in Figures 6, 7, and 8, respectively. It is noted that the mean cost value is represented as the reciprocal of the mean cost value in Figures 6(a), 7(a), and 8(a). According to Ref. [18], N_w is determined as

$$\frac{\mathrm{PL}}{(N_w + 1) \cdot N_d} < \mathrm{DI}, \tag{38}$$

where PL is the route length, DI is the minimal diameter of all



(a) The convergence curves in case II

(b) The 3D planned routes in case II

(c) The contour of planned routes in case II

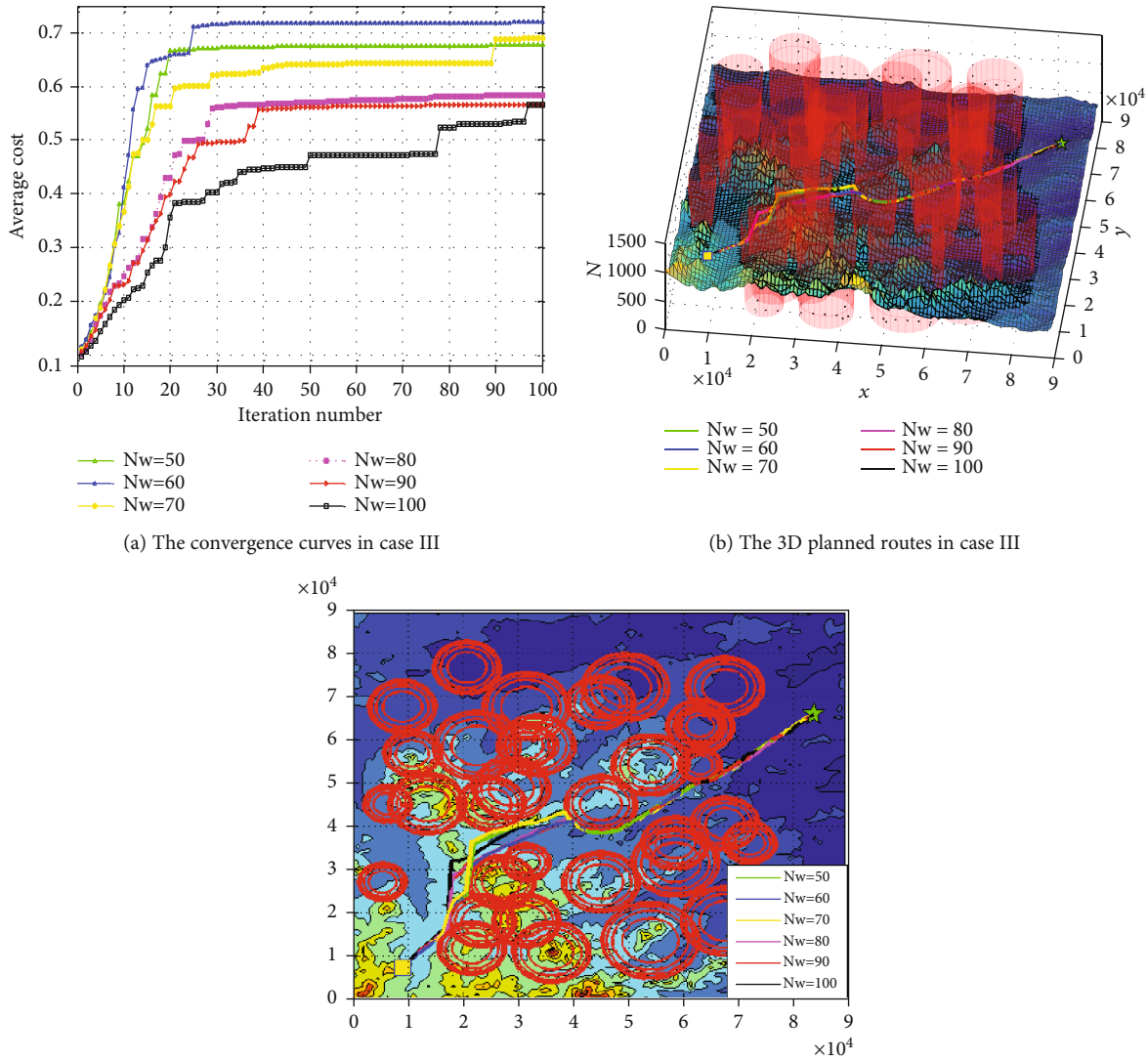
FIGURE 7: The comparison of results with different numbers in case II.

the missiles and radars in space, and N_w is the number of the waypoints (not including start point and endpoint). In general, DI = 0.5. The minimum value of N_w is determined by (38).

From Table 1, it is easy to know that the best mean fitness value and standard deviation are obtained when $N_w = 20$ in case I. The smallest value of the best cost is obtained when $N_w = 5$, and the best FR is obtained when $N_w = 7$, $N_w = 10$, $N_w = 15$, and $N_w = 20$. When $N_w = 7$ and $N_w = 10$, the indicator $\widetilde{G_c}$ is superior to when $N_w = 15$, $N_w = 20$, and $N_w = 25$. The convergence speed is faster than that of the larger number of path points. Overall, in case I, $N_w = 20$ is the most appropriate value. The FR and the mean fitness value are superior to the other N_w. In case II, as the number of the threat resources increases, the standard deviation and the mean fitness value get worse than those of case I. The best mean fitness cost is obtained when $N_w = 20$, and the standard deviation is obtained when $N_w = 10$. The best FR is obtained when $N_w = 10$ and $N_w = 15$.

In case III, the best results are obtained when $N_w = 60$. The preference of $N_w = 50$, $N_w = 60$, and $N_w = 70$ is better than that of $N_w = 80$, $N_w = 90$, and $N_w = 100$. When the number of the waypoints gets larger, the search of the path becomes more difficult and the FR will get low. On the other hand, when the number of the waypoint is too small, the dangerous zone has no way to be avoided efficiently. It is more obvious for the environment where a large number of threat sources exist. If the number of the waypoint is too small or too large, the standard deviation and the FR will get worse. Meanwhile, due to the number of threat sources increases, the security area will get narrow. The planned paths with different numbers of path points will be clustered in a narrow region. The greater the number of waypoints is, the slower convergence speed is, and more complicate the problem becomes. Figure 7 shows the experimental results in case II.

6.2. The Performance Comparison of Different Algorithms. In this section, to evaluate the performance of DCA∗PSO, some population-based optimization algorithms, such as DE [38], PSO [17], QPSO [39], AIWPSO [40], and PSOPC [41], are utilized to make several comparisons based on different indicators. Three cases are designed to test the performance of six algorithms.



(a) The convergence curves in case III

(b) The 3D planned routes in case III

(c) The contour of planned routes in case III

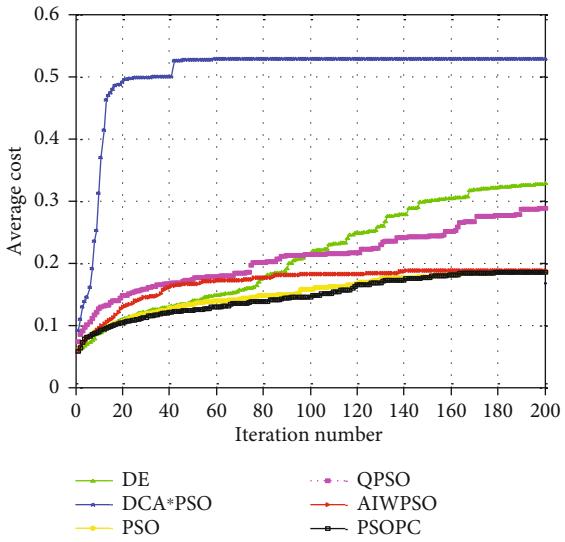
Figure 8: The comparison of results with different numbers in case III.

Table 2: The parameter values of different methods.

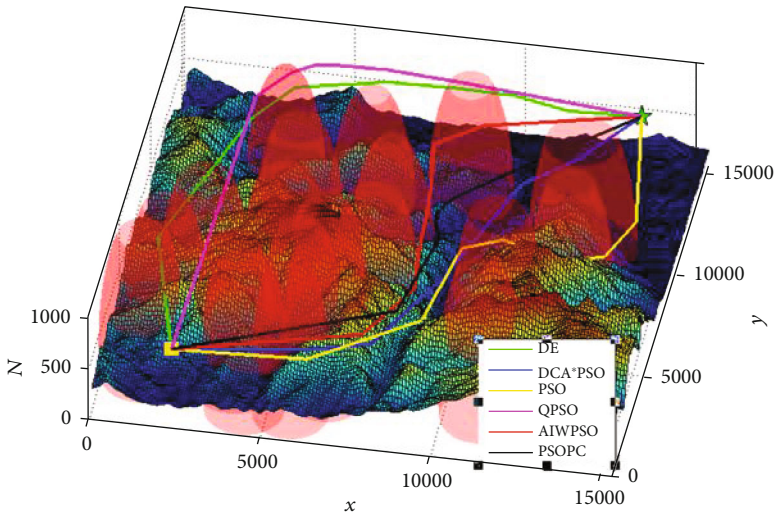
Algorithm	Parameters
DE	$F = 0.4, C_r = 0.9$
DCA∗PSO	$w = 0.73, c_1 = c_2 = 1.5$
PSO	$w = 0.7298, c_1 = c_2 = 1.4960$
QPSO	$b \in [0.3, 0.7]$
AIWPSO	$w \in [0, 1]$
PSOPC	$w \in [0.7, 0.9], c_1 = c_2 = 0.5, c_3 \in [0.4, 0.6]$

In case IV and case V, the population size and maximum number of iterations for six algorithms are the same, which are equal to 100 and 200, respectively. The number of the waypoints is set to 7 and 10 for cases IV and V, respectively. The specified parameters for the different algorithms are shown in Table 2. The simulation results for six different algorithms are given in Figure 9, Figure 10, and Table 3.

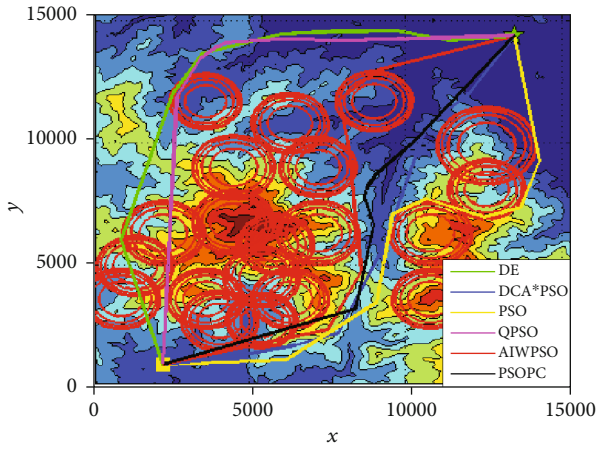
Figure 9(a) shows the convergence curves of the average fitness value during the process of 200 generations in case IV, which is a crucial indicator of algorithm performance. As can be seen from Figure 9(a), because the convergence curve of the DCA∗PSO algorithm is always above that of the other five algorithms, it is concluded that the convergence speed of the DCA∗PSO algorithm is faster than the DE, PSO, QPSO, AIWPSO, and PSOPC. The indicator G_c in Table 3 also reflects that the convergence speed of DCA∗PSO is superior to other algorithms. Moreover, the global search ability of the DCA∗PSO algorithm is better than the other five algorithms. From Table 3, it is easy to see that the DCA∗PSO algorithm has the minimum standard deviation, which indicates that the DCA∗PSO algorithm has stronger robustness. The FR, the average cost value, and standard deviation of DE and QPSO are better than PSO, AIWPSO, and PSOPC.



(a) The average cost in case IV

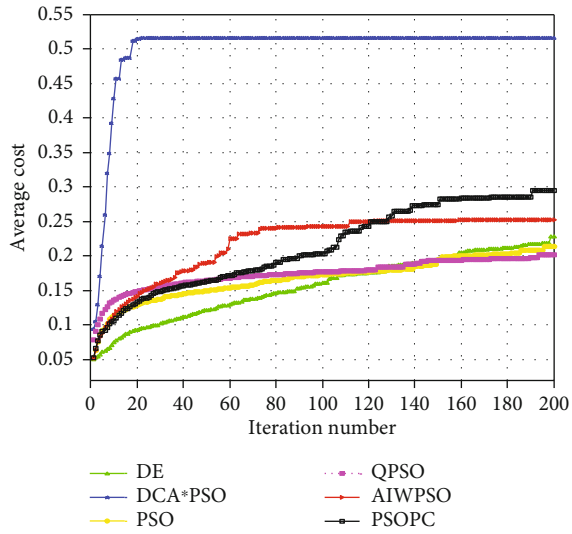


(b) The 3D UAV route in case IV

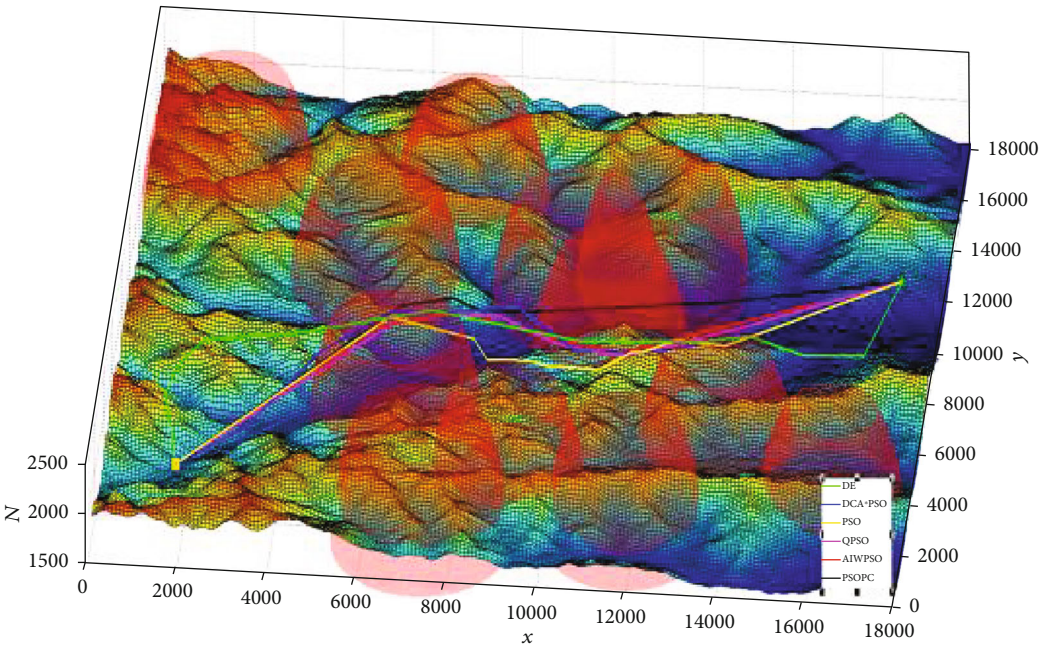


(c) The 2D UAV route in case IV

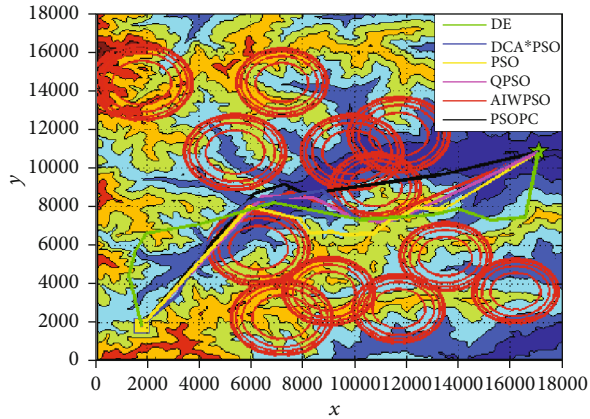
Figure 9: The comparison results of various methods in case IV.



(a) The average cost in case V



(b) The 3D UAV route in case V



(c) The 2D contour route in case V

Figure 10: The comparison results of various methods in case V.

Table 3: The statistical results of two cases.

Item	Method	Min cost	Mean cost	Std. dev.	$\tilde{G}_c$	FR (%)
Case IV	DE	1.9914	3.0480	1.4419	104	83.33
	DCA∗PSO	1.7237	1.8879	0.6125	11	96.67
	PSO	1.8511	5.3384	3.9335	115	23.33
	QPSO	1.8593	3.4639	1.5880	107	63.33
	AIWPSO	1.8275	5.3014	2.0900	67	20
	PSOPC	1.8308	5.3840	2.3791	126	23.33
Case V	DE	2.7575	4.3976	1.2025	170	26.67
	DCA∗PSO	1.6928	1.9400	0.1022	7	100
	PSO	2.2837	4.6835	1.3901	138	30
	QPSO	1.8928	4.9618	1.1591	87	10
	AIWPSO	1.8749	3.9607	1.8140	55	53.33
	PSOPC	1.8532	3.3924	1.6887	109	63.33

Table 4: The parameters of the algorithms.

Algorithm	Parameter
CSO	$m = 200, \varphi = 0.15$
CCPSO2	$m = 200, p = 0.5, S = [2, 5, 10, 20, 50, 100]$
DMSPSO	$m = 60, w = 0.729, c_1 = c_2 = 1.49445$
SLPSO	$m = 100, \alpha = 0.5, \beta = 0.01$
DCA∗PSO	$m = 200, w = 0.73, c_1 = c_2 = 1.5, N_d = 5$

The typical 3D stereo displays and contour profiles corresponding to six route planners are shown in Figures 9(b) and 9(c) in case V. It can be seen that the flight route generated by DCA∗PSO is shorter and smoother than those based on the DE, PSO, QPSO, AIWPSO, and PSOPC. Among these, the routes by the AIWPSO, PSO, PSOPC, and AIWPSO can avoid threat safely in case V. Due to the routes by DE and QPSO go through the threat, the two algorithms fail to search for a safe path. Threat-avoidance route can increase the effectiveness of performing mission for UAV and enhance the success rate of the mission.

The experimental results of case V are shown in Figure 10. The average convergence curves express that DCA∗PSO still is superior to DE, PSO, QPSO, AIWPSO, and PSOPC algorithms in Figure 10(a). From the statistical data in Table 3, it is not hard to find that the average fitness cost value, the standard deviation, $\widetilde{G_c}$, and the successful rate FR of DCA∗PSO are better than those of the other five algorithms. So DCA∗PSO can obtain more excellent performance of the search ability, robustness, and convergence speed in two cases. In case V, the performance of DE and QPSO is worse than case IV. The mean fitness cost of DE and QPSO is larger than AIWPSO and PSOPC algorithms. Figures 10(b) and 10(c) show the 3D stereo displays and contour profiles in case V, and only DE and DCA∗PSO can find the safe route to avoid the detection from the radars.

Table 5: The statistical results of different algorithms in case III.

Method	Min cost	Mean cost	Std. dev.	FR (%)
CSO	13.2990	13.5231	0.1422	0
CCPSO2	14.8774	20.7542	3.6629	0
DMSPSO	28.2125	29.2021	1.4512	0
SLPSO	23.9754	25.6341	0.7212	0
DCA∗PSO	1.3916	1.7671	0.9376	90

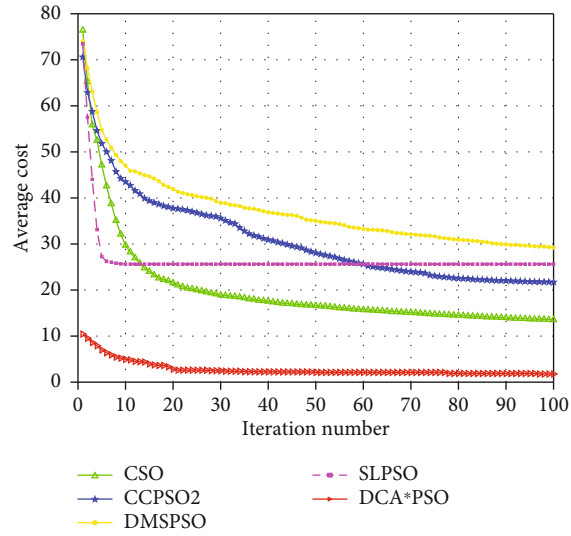


Figure 11: The average cost of different algorithms in case III.

Intelligent method is sensitive to the number of waypoint for the UAV route planning application. As the number of the waypoints increases, a feasible flight route obtained by the algorithm will become more and more difficult in high-dimensional search space. The large realistic terrain environment is utilized to test the performance of DCA∗PSO when solving the high-dimensional planning problem.

In order to further demonstrate the optimization performance of the proposed DCA∗PSO algorithm, SLPSO [33], CCPSO2 [34], DMSPSO [35], and CSO [36] are selected to solve the constructed multi-objective optimization model of route planning. The parameters of these algorithms are set in Table 4.

The comparison results are shown in Table 5. It is noted that the performance of CSO, CCPSO2, DMSPSO, and SLPSO in case III cannot find the feasible route in 30 runs. As can be seen from Table 5, the minimum cost, mean cost, and FR of DCA∗PSO is best among the five algorithms. But the standard deviation is not the minimum one in five algorithms. Because there were three turns in all 30 turns experiments cannot find a feasible solution for the DCA∗PSO algorithm, the standard deviation is increased by the penalty constant.

The path planners based on CSO, CCPSO2, DMSPSO, and SLPSO failed to generate a flight route in case III. The convergence curves are shown in Figure 11.

By comparing the convergence curves shown in Figure 11, it is easy to see that as the increase of waypoint number, the initial fitness cost of the DCA∗PSO algorithm

is far better than four improved PSO algorithm. The DCA∗PSO still can obtain a more satisfactory performance than the other four PSO-based algorithms and eventually find a flight optimal path with less cost when the number of path points is large. In general, the DCA∗PSO algorithm can effectively improve the performance of route planning in a 3D complex environment. The proposed DCA∗PSO algorithm takes on the ability to escape the local minimum value and improve the global search ability of algorithm.

7. Conclusion

In this paper, considering the minimum route length, the minimum flight height, the minimum risk of being detected, the dynamic constraints of fix-wing UAV (e.g., the turning angle and slope angle), and the terrain constraint, a multiobjective optimization model of path planning problem for fix-wing UAV is constructed. A novel improved PSO called DCA∗PSO is proposed to solve the UAV route planning problem. DCA∗PSO is presented by introducing the improved $A*$ algorithm and DC strategy. The improved $A*$ algorithm can quickly find a high-quality solution from the particles of PSO, which speeds up the optimization of the algorithm. Meantime, the DC strategy enhances the search efficiency by optimizing the structure of particles when the number of path points increases. The experiment results in three typical cases show that DCA∗PSO is better than other methods in terms of convergence speed, robustness, and premature avoidance. The proposed method can effectively provide a valuable reference for route planning in complex 3D terrain. In future work, the cooperative route planning of multiple UAVs will be researched and extended. In future work, more parallel large-scale PSO algorithms and complex terrain environments should be adopted to enhance the performance of the path planner.

Data Availability

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

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was funded by the China Postdoctoral Science Foundation, grant number 2020M680991 and Doctor Research Startup Foundation of SAU, grant number 18YB36.

References

[1] Y. Wang, P. Bai, X. Liang, W. Wang, J. Zhang, and Q. Fu, "Reconnaissance mission conducted by UAV swarms based on distributed PSO path planning algorithms," *IEEE Access*, vol. 7, no. 99, pp. 105086–105099, 2019.

[2] P. Iscold, G. A. S. Pereira, and L. A. B. Torres, "Development of a hand-launched small UAV for ground reconnaissance," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 46, no. 1, pp. 335–348, 2010.

[3] H. M. Wang, Y. Zhang, X. Zhang, and Z. Li, "Secrecy and covert communications against UAV surveillance via multihop networks," *IEEE Transactions on Communications*, vol. 68, no. 1, pp. 389–401, 2020.

[4] M. Tortonesi, C. Stefanelli, E. Benvegnu, K. Ford, N. Suri, and M. Linderman, "Multiple-UAV coordination and communications in tactical edge networks," *IEEE Communications Society magazine*, vol. 50, no. 10, pp. 48–55, 2012.

[5] H. Wang, J. Wang, G. Ding, J. Chen, F. Gao, and Z. Han, "Completion time minimization with path planning for fixed-wing UAV communications," *IEEE Transactions on Wireless Communications*, vol. 18, no. 7, pp. 3485–3499, 2019.

[6] H. Wang, J. Wang, G. Ding, J. Chen, and J. Yang, "Completion time minimization for turning angle-constrained UAV-to-UAV communications," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4569–4574, 2020.

[7] Y. M. Yan, Y. T. Liang, H. Zhang et al., "A two-stage optimization method for unmanned aerial vehicle inspection of an oil and gas pipeline network," *Petroleum Science*, vol. 16, no. 2, pp. 458–468, 2019.

[8] S. Shao, Y. Peng, C. L. He, and Y. du, "Efficient path planning for UAV formation via comprehensively improved particle swarm optimization," *ISA Transactions*, vol. 97, pp. 415–430, 2020.

[9] S. Guo, X. Zhang, Y. du, Y. Zheng, and Z. Cao, "Path planning of coastal ships based on optimized DQN reward function," *Journal of Marine Science and Engineering*, vol. 9, no. 2, pp. 210–223, 2021.

[10] Y. V. Pehlivanoglu, "A new vibrational genetic algorithm enhanced with a Voronoi diagram for path planning of autonomous UAV," *Aerospace Science and Technology*, vol. 16, no. 1, pp. 47–55, 2012.

[11] R. Kala, A. Shukla, and R. Tiwari, "Fusion of probabilistic A ∗ algorithm and fuzzy inference system for robotic path planning," *Artificial Intelligence Review*, vol. 33, no. 4, article 9157, pp. 307–327, 2010.

[12] J. Y. Sun, J. Tang, and S. Y. Lao, "Collision avoidance for cooperative UAVs with optimized artificial potential field algorithm," *IEEE Access*, vol. 5, pp. 18382–18390, 2017.

[13] Y. J. Zhao, Z. Zheng, and Y. Liu, "Survey on computational-intelligence-based UAV path planning," *Knowledge Based Systems*, vol. 158, pp. 54–64, 2018.

[14] X. Cai, H. M. Zhao, S. F. Shang et al., "An improved quantum-inspired cooperative co-evolution algorithm with muli- strategy and its application," *Expert Systems with Applications*, vol. 171, article 114629, 2021.

[15] Z. H. Zhan, J. Zhang, Y. Lin et al., "Matrix-based evolutionary computation," *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2021.

[16] W. Deng, J. J. Xu, H. M. Zhao, and Y. J. Song, "A novel gate resource allocation method using improved PSO-based QEA," IEEE Transactions on Intelligent Transportation Systems, 2020.

[17] V. Roberge, M. Tarbouchi, and G. Labonte, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 132–141, 2013.

[18] W. Deng, J. Xu, X.-Z. Gao, and H. Zhao, "An enhanced MSIQDE algorithm with novel multiple strategies for global

optimization problems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2020.

[19] W. Deng, S. F. Shang, X. Cai et al., "Quantum differential evolution with cooperative coevolution framework and hybrid mutation strategy for large scale optimization," *Knowledge-Based Systems*, vol. 224, article 107080, 2021.

[20] X. F. Liu, Z. H. Zhan, Y. Gao, J. Zhang, S. Kwong, and J. Zhang, "Coevolutionary particle swarm optimization with bottleneck objective learning strategy for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 4, pp. 587–602, 2019.

[21] X. Zhang, K. J. Du, Z. H. Zhan, S. Kwong, T. L. Gu, and J. Zhang, "Cooperative coevolutionary bare-bones particle swarm optimization with function independent decomposition for large-scale supply chain network design with uncertainties," *IEEE Transactions on Cybernetics*, vol. 50, no. 10, pp. 4454–4468, 2020.

[22] X. W. Xia, L. Gui, F. Yu et al., "Triple archives particle swarm optimization," *IEEE Transactions on Cybernetics*, vol. 50, no. 12, pp. 4862–4875, 2020.

[23] Z. J. Wang, Z. H. Zhan, S. Kwong, H. Jin, and J. Zhang, "Adaptive granularity learning distributed particle swarm optimization for large-scale optimization," *IEEE Transactions on Cybernetics*, vol. 51, no. 3, pp. 1175–1188, 2021.

[24] Z. J. Wang, Z. H. Zhan, W. J. Yu et al., "Dynamic group learning distributed particle swarm optimization for large-scale optimization and its application in cloud workflow scheduling," *IEEE Transactions on Cybernetics*, vol. 50, no. 6, pp. 2715–2729, 2020.

[25] J. Y. Li, Z. H. Zhan, R. D. Liu, C. Wang, S. Kwong, and J. Zhang, "Generation-level parallelism for evolutionary computation: a pipeline-based parallel particle swarm optimization," *IEEE Transactions on Cybernetics*, 2020.

[26] Z. G. Chen, Z. H. Zhan, Y. Lin et al., "Multiobjective cloud workflow scheduling: a multiple populations ant colony system approach," *IEEE Transactions on Cybernetics*, vol. 49, no. 8, pp. 2912–2926, 2019.

[27] P. Yang, K. Tang, J. A. Lozano, and X. Cao, "Path planning for single unmanned aerial vehicle by separately evolving waypoints," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1130–1146, 2015.

[28] C. Huang and J. Y. Fei, "UAV path planning based on particle swarm optimization with global best path competition," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 32, no. 6, article 1859008, 2018.

[29] X. Zhang, X. Lu, S. Jia, and X. Li, "A novel phase angle-encoded fruit fly optimization algorithm with mutation adaptation mechanism applied to UAV path planning," *Applied Soft Computing*, vol. 70, pp. 371–388, 2018.

[30] Y. C. du, M. X. Zhang, H. F. Ling, and Y. J. Zheng, "Evolutionary planning of multi-UAV search for missing tourists," *IEEE Access*, vol. 7, pp. 73480–73492, 2019.

[31] Q. Yang and S. J. Yoo, "Optimal UAV path planning: sensing data acquisition over IoT sensor networks using multi-objective bio-inspired algorithms," *IEEE Access*, vol. 6, pp. 13671–13684, 2018.

[32] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp. 39–43, Nagoya, Japan, 1995.

[33] R. Cheng and Y. Jin, "A social learning particle swarm optimization algorithm for scalable optimization," *Information Sciences*, vol. 291, pp. 43–60, 2015.

[34] Xiaodong Li and Xin Yao, "Cooperatively coevolving particle swarms for large scale optimization," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 2, pp. 210–224, 2012.

[35] J. Liang and P. N. Suganthan, "Dynamic multi-swarm particle swarm optimizer with sub-regional harmony search," in *Proceedings of IEEE Congress on Evolutionary Computation*, pp. 522–528, Barcelona, Spain, 2011.

[36] Ran Cheng and Yaochu Jin, "A competitive swarm optimizer for large scale optimization," *IEEE Transactions on Cybernetics*, vol. 45, no. 2, pp. 191–204, 2015.

[37] A. Jarvis, H. I. Reuter, and A. Nelson, "Hole-filled seamless SRTM data V4, International Centre for Tropical Agriculture (CIAT)," 2008, https://srtm.csi.cgiar.org/.

[38] K. Y. Kok and P. Rajendran, "Differential-evolution control parameter optimization for unmanned aerial vehicle path planning," *PLoS One*, vol. 11, no. 3, article e0150558, 2016.

[39] Y. Fu, M. Ding, and C. Zhou, "Path planning for UAV based on quantum-behaved particle swarm optimization," in *In Proceedings of MIPPR 2009 - Medical Imaging, Parallel Processing of Images, and Optimization Techniques: 6th International Symposium on Multispectral Image Processing and Pattern Recognition*, vol. 7497, p. 74970B, Yichang, China, 2009.

[40] A. Nickabadi, M. M. Ebadzadeh, and R. Safabakhsh, "A novel particle swarm optimization algorithm with adaptive inertia weight," *Applied Soft Computing*, vol. 11, no. 4, pp. 3658–3670, 2011.

[41] S. He, Q. H. Wu, J. Y. Wen, J. R. Saunders, and R. C. Paton, "A particle swarm optimizer with passive congregation," *Biosystems*, vol. 78, no. 1-3, pp. 135–147, 2004.