applied sciences

MDPI

Article

Multiple Swarm Fruit Fly Optimization Algorithm Based Path Planning Method for Multi-UAVs

Kunming Shi [1,2], Xiangyin Zhang [1,2,3,4,*] and Shuang Xia [1,3]

1 Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China; kunming1002@emails.bjut.edu.cn (K.S.); xias@emails.bjut.edu.cn (S.X.)
2 Beijing Key Laboratory of Computational Intelligence and Intelligent System, Beijing 100124, China
3 Engineering Research Center of Digital Community, Ministry of Education, Beijing 100124, China
4 Beijing Laboratory for Urban Mass Transit, Beijing 100124, China
* Correspondence: xy_zhang@bjut.edu.cn

Received: 27 February 2020; Accepted: 15 April 2020; Published: 19 April 2020

Abstract: The path planning of unmanned aerial vehicles (UAVs) in the threat and countermeasure region is a constrained nonlinear optimization problem with many static and dynamic constraints. The fruit fly optimization algorithm (FOA) is widely used to handle this kind of nonlinear optimization problem. In this paper, the multiple swarm fruit fly optimization algorithm (MSFOA) is proposed to overcome the drawback of the original FOA in terms of slow global convergence speed and local optimum, and then is applied to solve the coordinated path planning problem for multi-UAVs. In the proposed MSFOA, the whole fruit fly swarm is divided into several sub-swarms with multi-tasks in order to expand the searching space to improve the searching ability, while the offspring competition strategy is introduced to improve the utilization degree of each calculation result and realize the exchange of information among various fruit fly sub-swarms. To avoid the collision among multi-UAVs, the collision detection method is also proposed. Simulation results show that the proposed MSFOA is superior to the original FOA in terms of convergence and accuracy.

Keywords: multiple unmanned aerial vehicles (multi-UAVs); fruit fly optimization algorithm (FOA); path planning; multi-swarms

1. Introduction

Unmanned aerial vehicles (UAVs) have become an area of great concern to many governmental and military organizations around the world. The autonomy level of the UAVs depends on the methodology used to control the vehicle and plan its flight path [1]. Therefore, path planning, which is to generate a feasible path between two points that has an optimal or near-optimal performance satisfying constraint conditions, is one of the most important challenges in the autonomous navigation process of UAVs [2]. Typically, in increasingly complex mission environments, only one UAV usually cannot meet the mission requirements, so there are often multiple UAVs (multi-UAVs) on the same mission at the same time [3]. This paper mainly discusses the path planning of multi-UAVs cooperation. To solve the cooperative path planning problem of multi-UAVs, an approach is proposed: Priority is set to each UAV. The first priority UAV does not need a plan. Its sub-goal is the final goal. The UAV with second priority plans the path to avoid collision with the first priority UAV. The UAV with the third priority needs to plan its path to avoid collision with the former UAVs [4]. It is proven that finding the optimal path is an non-deterministic polynomial complete problem [2]. As the scale of the problem increases, the complexity of the problem increases rapidly. In many existing works, the optimal path is always considered as the shortest path, and the deterministic search algorithm is used to find the shortest path. However, the optimal path of the UAV should be considered to be related to the

minimum distance path, average height, fuel consumption, radar exposure, and so on, which make the problem more complex [5]. In order to reduce the computational complexity, many approaches have been applied to the path planning problem, which can be classified into traditional methods and bionic inspired algorithms. The traditional path planning methods contain the A* search algorithm [6], the polynomial optimization method [7], and artificial potential field based methods [8], while the bionic inspired algorithms contain the simulated annealing algorithm (SAA) [9], genetic algorithm (GA) [5], differential evolution (DE) [10], artificial bee colony (ABC) [11], particle swarm optimization (PSO) algorithm [12,13], and ant colony optimization (ACO) [14]. Among these bionic inspired algorithms, ACO is usually applied to the combinational optimization problems [14,15], while the others are usually used to solve the continuous optimization problems. As a kind of population-based random search algorithm, PSO has been widely applied to UAV path planning [16]. The basic idea of PSO comes from the behavior of birds that randomly search for food within an area. To simulate this situation, the particle swarm was developed as a useful computational technique for solving the optimization problem [17].

As a novel bio-inspired technique, the fruit fly optimization algorithm (FOA) was first proposed by Pan in 2011 [18], inspired by the food finding behavior of fruit flies. FOA is well-known for its simple principle, few adjustable parameters, strong global optimization capability, and fast convergence speed [19]. So far, FOA has been used in a variety of applications, such as general regression neural network parameters optimization [20,21] and the tuning of controller parameters [22]. Several improved versions of FOA were proposed by researchers to enhance the search efficiency and global search ability of the basic FOA; these include the multi-swarm fruit fly optimization algorithm (MFOA) [23] and improved fruit fly optimization (IFFO) [24], to name a few.

In this paper, the new multiple swarm fruit fly optimization algorithm (MSFOA) is proposed. In the MSFOA, the whole fruit fly swarm are divided into several sub-swarms with multi-tasks. In the early searching stage, the sub-swarms are inclined to global searching, while at the end of the searching stage, they are inclined to local searching. The strategy can expand the searching space and improve the global and local searching ability. Then, the offspring competition strategy is introduced to improve the utilization degree of each calculation result and realize the exchange of information among various fruit fly sub-swarms. A path planner for multi-UAVs based on the proposed MSFOA is designed to find the optimal path for each UAV in complex three-dimensional environments. A collision detection method is proposed to avoid collision among multi-UAVs. The simulation results show that the proposed MSFOA-based path planner can effectively solve the coordinated path planning problem among multi-UAVs.

The rest of the paper is arranged as follows: The Section 2 introduces the basic FOA technique. In Section 3, the proposed MSFOA and its application to multi-UAV path planning are presented. Then, the comparison experiment results are listed and discussed in Section 4. Section 5 gives the conclusion.

2. Overview of Fruit Fly Optimization Algorithm

Fruit flies have a superior sense of smell and vision [18]. In their search for food, fruit flies can not only use their sense of smell to detect smells in the air, but also can use their sense of sight to locate food and other flies. In the FOA, the flies first use their sense of smell to find the direction of the food source, and then fly toward the direction with the greatest smell concentration. After that, the fruit flies use their vision to find the location of the food and other flies and fly to the direction obtained by their vision.

Without loss of generality, the global optimization problem in the continuous domain can be summarized as

$$\begin{aligned} &\min F(x) \\ &\text{s.t.} x_j \in [X_{\min}, X_{max}], j = 1, 2, \ldots, D \end{aligned} \tag{1}$$

where $F(\boldsymbol{x})$ is the objective function with the D-dimensional decision variable $\boldsymbol{x} = (x_1, x_2, \ldots, x_D)$ and $X_{\min}$ and X_{max} are the lower and upper bounds of the decision space, respectively.

The parameters of FOA are the population size M_{pop} and the maximum number of iterations NC. Firstly, the fruit fly swarm location (x_{best}, y_{best}) is randomly initialized in the decision space. The main process of the FOA contains two parts, namely olfactory search and visual search [1,18].

(1) Olfactory search

In the olfactory search process, the fruit fly swarm generates in total M_{pop} new locations of the food source that is randomly around its current location using random strategy, which is given by

$$\begin{cases} x_i = x_{\text{best}} + \text{random}(-1, 1) \\ y_i = y_{\text{best}} + \text{random}(-1, 1) \end{cases} \tag{2}$$

where x_i and y_i, $i = 1, 2, \ldots, M_{\text{pop}}$, denote the location coordinate of the i-th fruit fly and random(−1,1) is the random value in the range of (−1,1).

For each location produced in the olfactory search process, its smell concentration judgement S_i is the reciprocal of the distance between the location and the origin, as follows

$$\begin{cases} D_i = \sqrt{x_i^2 + y_i^2} \\ S_i = 1/D_i \end{cases} \tag{3}$$

In the FOA framework, the smell concentration judgement S_i corresponds to the candidate solution of the objective function $F(.)$ in the decision space. The smell concentration of the food source can be calculated using the smell concentration judgement as follows

$$smell_i = F(S_i) \tag{4}$$

(2) Visual search

The visual search process of the fruit fly swarm can be regarded as a greedy selection procedure, where the fruit fly swarm observes all the locations generated via the above olfactory search and finds out the best location that has the minimum smell concentration, as follows

$$Index = \underset{i}{\text{argmin}}(smell_i) \tag{5}$$

Then the best smell concentration $smell_{index}$ generated via the olfactory search is compared with the smell concentration $smell_{\text{best}}$ of the original fruit fly swarm location (x_{best}, y_{best}). If $smell_{index} < smell_{\text{best}}$, then the original location (x_{best}, y_{best}) will be replaced by the newly generated location (x_{index}, y_{index}) and the fly swarm will fly towards the new location by using the visual search. The visual search can be described as follows

$$\begin{cases} x_{\text{best}} = x_{Index} \\ y_{\text{best}} = y_{Index} \\ smell_{\text{best}} = smell_{Index} \end{cases} \quad \text{if} smell_{Index} < smell_{\text{best}} \tag{6}$$

The olfactory and visual search processes are repeated many times until a program termination condition is reached. The general process of the FOA in solving the minimum problem min$F(\boldsymbol{x})$ is described in Algorithm 1.

Algorithm 1: Process of the FOA

/*Initialization*/
Set the population size M_{pop} and the max number of iterations NC.
Randomly initialize the fruit fly swarm's location (x_{best}, y_{best}) in the search space.
/* *Iterative search* */
for n = 1: NC
  /* *Olfactory search* */
  for i = 1: M_{pop}
    $x_i = x_{best}$ + random(–1,1)
    $y_i = y_{best}$ + random(–1,1)
    /* *Calculate the smell concentration* */
    $$D_i = \sqrt{x_i^2 + y_i^2}$$
    $S_i = 1/D_i$
    $$smell_i = F(S_i)$$
  end for
/* *Visual search* */
  $Index = \underset{i}{\operatorname{argmin}}(smell_i)$
  if $smell_{Index}$ is smaller than $smell_{best}$ of the fruit fly swarm's location (x_{best}, y_{best}) **then**
    Fruit flies move to the location (x_{Index}, y_{Index}) and update the smell concentration:
    $$x_{best} = x_{Index}$$
    $$y_{best} = y_{Index}$$
    $$smell_{best} = smell_{Index}$$
  end if
end for

3. UAV Path Planning Method Based Modified Fruit Fly Optimization Algorithm

In this section, the MSFOA is proposed to enhance the searching ability of the original FOA and is applied to the path planning for multi-UAVs. The symbol description of the main variables appearing in the proposed MSFOA is listed in Abbreviations.

3.1. MSFOA

3.1.1. Motivation

The FOA converges quickly when applied to a single UAV path planning. Yet when it is applied to planning the path of multi-UAVs, it is easy to get lost in the local optimal solution. The original FOA is random when searching the solution, which means the convergence speed is fast in the early stage, but it is difficult to converge towards the optimal solution in the later stage. When the fruit fly's path is near the optimal solution, the algorithm may ignore the path, because the path length and the threating level and other constraints are considered. The optimal path is often between the edge of the threating cylinder or the valley. When the fly finds a path near the optimal solution, the path usually falls into the threating cylinder or through the interior of the mountain. This path with a high smell concentration is easy to eliminate but near the optimal solution. When the algorithm is used to plan the path for a simple scenario, it can be repeated many times to get the optimal solution, but when the algorithm is applied to some complex scenarios, such as multi-UAVs path planning or path planning in complex scenarios, the convergence ability of FOA is obviously insufficient.

In FOA, there is no communication between fruit flies and the direction of the path depends only on the given random direction and the random distance, so the ability to converge in solving complex problems is insufficient. In order to overcome the shortfalls of the FOA, increasing the searching ability and communication between fruit flies can improve the convergence ability of the algorithm. At the

same time, in order to avoid the problem of possible collision when planning multiple UAVs paths, this algorithm also adds the method of checking for collisions.

3.1.2. Multi-Swarm with Multi-Tasks Strategy

The searching process can be separated by global searching and local searching. Global searching occurs in the early stage of searching, while local searching occurs in the end stage of searching. In this paper, global searching and local searching are considered as different tasks, which should employ different searching strategies. This paper uses a threshold to confirm the two different stages. When the cost is larger than the threshold value, the algorithm is in the global searching period, when the cost is smaller than the threshold value, the algorithm is in the local searching period.

In the global searching period, it is important to enhance the searching space to improve global searching ability since the size of the searching space has a significant impact on the results. FOA only uses the single swarm approach to search for all fruit flies from a single location, thus when the searching space is small, the searching capacity is insufficient. In MSFOA, the searching space of the algorithm can be expanded linearly by using multi-swarm fruit fly at the same time and starting from many different positions. The fruit fly population M_{pop} is divided equally into G swarms. G is the total amount of swarms. Every fruit fly intends to have a big step in searching as shown in Equations (10) and (11). $\sin((\pi/2) * \text{random}(-1, 1))$ is introduced to map $\text{random}(-1, 1)$ near -1 and 1.

$$X_i^{(g)} = x_{\text{best}}^{(g)} + \sin((\pi/2) * \text{random}(-1, 1)) \tag{7}$$

$$Y_i^{(g)} = y_{\text{best}}^{(g)} + \sin((\pi/2) * \text{random}(-1, 1)) \tag{8}$$

where $X_i^{(g)}$ and $Y_i^{(g)}$, $i = 1, 2, \ldots, M_{pop}/G$, $g = 1, 2, \ldots, G$ denote the position of the i-th fruit fly in the g-th swarm, while $x_{\text{best}}^{(g)}$ and $y_{\text{best}}^{(g)}$ are the concentration position of the g-th swarm.

In the local searching period when fruit fly swarms are near the global optimum, it is important to reduce searching space for further convergence; fruit flies should pay attention to the points near themselves to have a better result. The step of searching is dwindled.

$$X_i^{(g)} = x_{\text{best}}^{(g)} + R * \text{random}(-1, 1) \tag{9}$$

$$Y_i^{(g)} = y_{\text{best}}^{(g)} + R * \text{random}(-1, 1) \tag{10}$$

where R is the searching step dwindled parameter.

3.1.3. Competitive Strategies of Offspring

In order to avoid multi-swarm falling into local optimum, each swarm is independent in visual search and olfaction search, while the information exchange occurs in the competition of offspring, which is explained in detail below. In this way, not only can the multi-swarms not get into the local optimum at the same time, but also the information exchange between the species cannot be guaranteed.

The competition strategy of the offspring uses the optimal individual to cross with other flies at random to generate a new swarm at the same scale and select a new optimal individual. That is, the swarm in the same number has two optimal individuals, one comes from the old swarm, the other from the new swarm. The optimal solution of the swarm is obtained after competition between two optimal individuals.

The effect of this method is when fruit fly A is near the optimal solution but its smell concentration is high, but the individual from A hybridized with random fruit fly B has the path information of both fruit fly A and B; the path information of the fruit fly can be obtained by distributing the weight of the path data of the two flies. This strategy can increase the probability that the fruit fly near the optimal

solution will be selected by the algorithm. The procedure for generating the i-th new individual of the g-th swarm is as follows:

$$\begin{cases} S^{(g)}_{\text{old},\,i} = \sqrt{\left(X^{(g)}_i\right)^2 + \left(Y^{(g)}_i\right)^2} \\ smell^{(g)}_{\text{old},\,i} = F(S^{(g)}_{\text{old},\,i}) \end{cases} \tag{11}$$

$$Index^{(g)}_{\text{old}} = \underset{i}{\text{argmin}}(smell^{(g)}_{\text{old},\,i}) \tag{12}$$

$$x^{(g)}_{\text{oldbest}} = X^{(g)}_{Index^{(g)}_{\text{old}}} \quad y^{(g)}_{\text{oldbest}} = Y^{(g)}_{Index^{(g)}_{\text{old}}} \tag{13}$$

$$X^{(g)}_{\text{new},\,i} = coe_1 * x^{\text{random}(1,G)}_{\text{oldbest}} + coe_2 * X^{(g)}_i \quad y^{(g)}_{\text{new},\,i} = coe_1 * y^{\text{random}(1,G)}_{\text{oldbest}} + coe_2 * Y^{(g)}_i \tag{14}$$

where $\text{random}(1,\ G)$ produces random integers between 1 and G. $X^{(g)}_{\text{new},\,i}$ and $Y^{(g)}_{\text{new},\,i}$ mean the position of the i-th fly in the g-th new swarm. coe_1 and coe_2 are the weights of parental path data, satisfying:

$$coe_1 + coe_2 = 1 \tag{15}$$

The calculation of best smell concentration is given by:

$$\begin{cases} S^{(g)}_{\text{new},\,i} = \sqrt{\left(X^{(g)}_{\text{new},\,i}\right)^2 + \left(Y^{(g)}_{\text{new},\,i}\right)^2} \\ smell^{(g)}_{\text{new},\,i} = F(S^{(g)}_{\text{new},\,i}) \end{cases} \tag{16}$$

$$Index^{(g)}_{\text{new}} = \underset{i}{\text{argmin}}(smell^{(g)}_{\text{new},i}) \tag{17}$$

Then all the fruit flies in every swarm fly to the point with the minimum smell concentration of each swarm. If the smell concentration of the offspring is less than the smell concentration of their parent ($smell^{(g)}_{\text{new},\,Index^{(g)}_{\text{new}}} < smell^{(g)}_{\text{old},\,Index^{(g)}_{\text{old}}} < smell^{(g)}_{\text{best}}$), then:

$$\begin{cases} x^{(g)}_{\text{best}} = X^{(g)}_{\text{new},\,Index^{(g)}_{\text{new}}} \\ y^{(g)}_{\text{best}} = Y^{(g)}_{\text{new},\,Index^{(g)}_{\text{new}}} \\ smell^{(g)}_{\text{best}} = smell^{(g)}_{\text{new},\,Index^{(g)}_{\text{new}}} \end{cases} \tag{18}$$

Otherwise ($smell^{(g)}_{\text{old},\,Index^{(g)}_{\text{old}}} < smell^{(g)}_{\text{new},\,Index^{(g)}_{\text{new}}} < smell^{(g)}_{\text{best}}$):

$$\begin{cases} x^{(g)}_{\text{best}} = X^{(g)}_{Index^{(g)}_{\text{old}}} \\ y^{(g)}_{\text{best}} = Y^{(g)}_{Index^{(g)}_{\text{old}}} \\ smell^{(g)}_{\text{best}} = smell^{(g)}_{\text{old},\,Index^{(g)}_{\text{old}}} \end{cases} \tag{19}$$

if $smell^{(g)}_{\text{best}}$ is smaller than $smell^{(g)}_{\text{old},\,Index^{(g)}_{\text{old}}}$ or $smell^{(g)}_{\text{new},\,Index^{(g)}_{\text{new}}}$, $x^{(g)}_{\text{best}}$ and $y^{(g)}_{\text{best}}$ remain unchanged; where $x^{(g)}_{\text{best}}$ and $y^{(g)}_{\text{best}}$ are the location of the g-th fruit fly swarm.

3.1.4. The Process of MSFOA

The whole process of the MSFOA can be described by Algorithm 2.

Algorithm 2: MSFOA

/*Initialization*/
Set the number of the fruit fly population M_{pop}, the max number of iterations NC, swarms' initial position $[x_{\text{best}}, y_{\text{best}}]$, swarm number g, the *threshold*, genetic coefficient of hybridization coe_1 and coe_2, the parameter R.
for $nc = 1: NC$
  for $g = 1: G$
  /* M*ulti-swarm with multi-task searching strategy* */
    for $i = 1: M_{\text{pop}}/G$
      if $smell_{\text{best}}^{(g)}$ is larger than *threshold*
        $X_i^{(g)} = x_{\text{best}}^{(g)} + \sin((\pi/2) * \text{random}(-1, 1))$
        $Y_i^{(g)} = y_{\text{best}}^{(g)} + \sin((\pi/2) * \text{random}(-1, 1))$
      else
        $X_i^{(g)} = x_{\text{best}}^{(g)} + R * \text{random}(-1, 1)$
        $Y_i^{(g)} = y_{\text{best}}^{(g)} + R * \text{random}(-1, 1)$
      end
      /* *Calculate smell concentration* */
      $S_{\text{old},\,i}^{(g)} = \sqrt{\left(X_i^{(g)}\right)^2 + \left(Y_i^{(g)}\right)^2}$
      $smell_{\text{old},\,i}^{(g)} = F(S_{\text{old},\,i}^{(g)})$
      /*Vision searching in old swarms*/
      $Index_{\text{old}}^{(g)} = \underset{i}{\text{argmin}}(smell_{\text{old},i}^{(g)})$
    end for
  end for
  /*Competitive strategies of offspring*/
  for $g = 1: G$
    for $i = 1: M_{\text{pop}}/G$
      $X_{\text{new},\,i}^{(g)} = coe_1 * x_{\text{oldbest}}^{\text{random}(1,G)} + coe_2 * X_i^{(g)}$
      $Y_{\text{new},\,i}^{(g)} = coe_1 * y_{\text{oldbest}}^{\text{random}(1,G)} + coe_2 * Y_i^{(g)}$
      /* *Vision searching in new swarms* */
      $S_{\text{new},\,i}^{(g)} = \sqrt{\left(X_{\text{new},\,i}^{(g)}\right)^2 + \left(Y_{\text{new},\,i}^{(g)}\right)^2}$
      $smell_{\text{new},\,i}^{(g)} = F(S_{\text{new},\,i}^{(g)})$
      $Index_{\text{new}}^{(g)} = \underset{i}{\text{argmin}}(smell_{\text{new},i}^{(g)})$
    end for
  end for
  /* *Competitive strategy of offspring* */
  for $g = 1: G$
    Competition of offspring and visual searching in each swarm by formulas (18) and (19)
  end for
end for
/*Output*/
Find the best individual from the G swarms and export it.

3.2. MSFOA-Based Path Planning

3.2.1. Problem Modeling

UAV path planning needs to adapt to current terrain and physical world constraints, such as terrain undulation, gravitational acceleration, etc. In addition, the path of the UAV cannot exceed the limits of the UAV itself, such as turning radius, glide angle, flying speed, minimum flying height,

and so on. Within these limits, an optimal flight path is planned for the UAV, which can avoid a specific area and reach the target point with less energy consumption in a shorter time. The collision risk between UAVs needs to be considered in the three-dimensional (3D) cooperative path planning of multi-UAVs.

Path representation is a key problem in path planning, and the grid-based map [15] and spline curve [1,5] are two common methods. Zhang proves that by using the B-spline-based path representation strategy, the path can be constructed using a relatively smaller number of parameters than using a complete geometric description of the path [1]. Thus, in charting our course, the algorithm uses very few control points to record the path. When the algorithm needs to use the full path, it will use the control points to generate the whole path.

In this paper, the objective function divides the path constraints into the following categories: path length, threat cost, altitude cost, terrain constraint, corner constraint, and descent angle constraint. All the constraints constitute the objective function. The algorithm uses the objective function to express the quality of a path. Before calculating the objective function, it is necessary to generate a complete path with path points before calculating. The objective function is calculated as follows:

$$J = \sum JH + JL + JT + CH + CT + CG + JP \tag{20}$$

where the objective function includes the altitude cost JH, the length cost JL, the threat cost JT, the terrain constraint CH, the corner constraint CT, the climb down constraint CG, and the collision risk JP, which are described in detail as follows.

The altitude cost is determined by all points along the path, which reflects the flying altitude of the UAV above the ground. When flying at a low altitude, the UAV can benefit from terrain masking to avoid unknown radars [1]. Suppose the whole UAV path is represented by k points, if the variable z_n is the height of the n-th point in the path, and h_{min} is the minimum flying height of the UAV, then the altitude cost JH adds up the height of all the points along the path on the ground as follows:

$$JH = \sum_{n=1}^{k} (z_n - h_{min}) \tag{21}$$

The path length cost JL reflects the flight length of the entire UAV path, which calculates and adds the distance between any two adjacent points to get the total length as follows:

$$\text{with} \quad JL = \sum_{n=1}^{k-1} l_n, \quad l_n = \sqrt{(x_{n+1} - x_n)^2 + (y_{n+1} - y_n)^2 + (z_{n+1} - z_n)^2} \tag{22}$$

where the variable x_n is the x coordinate of the n-th point in the path, y_n is the y coordinate of the n-th point in the path, and z_n is the z coordinate of the n-th point in the path.

The terrain constraints are determined by the length of the path in the underground section. This symbol is used to punish those paths that are not entirely on the ground. The variable p is the information of the map, and function H(p) means the height of the current point in map p.

$$\text{with} \quad CH = \sum_{n=1}^{k} T_n, \quad T_n = \begin{cases} H(p) - z_n & , \text{ if } z_n < H(p) \\ 0 & , \text{ if } z_n > H(p) \end{cases} \tag{23}$$

In this paper, the threat around the threat point is only related to the distance, so the threat area is approximated as a cylinder, within a certain range; the closer to the threat point, the greater the degree of threat. The variable n_r is the amount of threaten points. $T_{\mathrm{de}}(i)$ is the coordinate of the i-th threaten points.

$$JT = \sum_{n=1}^{k-1} l_n \cdot \sum_{i=1}^{n_R} T_{\mathrm{de}}(i) \tag{24}$$

The minimum angle of turning is determined by the maximum speed and gravitational acceleration. The turning angle of each point in the path is calculated. If current turning angle is greater than the max turning angle, and the turning angle constraint increases iteratively. Iterations are added to quickly filter out individuals who do not exceed the limit. The calculation is as follows:

$$\begin{gathered} CT = \sum_{n=1}^{k-1} g_n \\ \text{with} \\ g_n = \begin{cases} CT, \text{ if } \varphi_n > \varphi_{max} \\ 0, \text{ if } \varphi_n < \varphi_{max} \end{cases} \end{gathered} \tag{25}$$

where φ_k denotes the turning angle in the n-th point in the path and φ_{max} denotes the maximum turning angle of the UAV.

CG is used to constrain the slope at the current point to meet the maximum climb slopes α and minimum climb slopes β.

$$\begin{gathered} CG = \sum_{n=1}^{k-1} g_S^n \\ \text{with} \\ g_S^n = \begin{cases} C, \text{ if } \beta_n \le S_n \le \alpha_n \\ 0, \text{ if } \beta_n \ge S_n \text{ or } S_n \ge \alpha_n \end{cases} \end{gathered} \tag{26}$$

where

$$\begin{gathered} \alpha_n = -1.5377 \times 10^{-10} z_n^2 - 2.6997 \times 10^{-5} z_n + 0.4211 \\ \beta_n = 2.5063 \times 10^{-9} z_n^2 - 6.3014 \times 10^{-6} z_n - 0.3257 \\ S_n = \frac{z_{n+1} - z_{kn}}{\sqrt{\left(x_{n+1} - x_n\right)^2 + \left(y_{n+1} - y_n\right)^2}} \end{gathered} \tag{27}$$

The cost JP is used to assess the risk of collisions between UAVs. The risk of collision between UAVs is determined by the time it takes to reach two points very close together. When planning the path of multi-UAVs, first, these UAVs are numbered according to the degree of importance and then the UAVs with the former number are given priority of passage on the map because each UAV has to avoid the UAV with greater priority.

$$JP_n = \sum_{k=1}^{N-1} PC \cdot \Delta_{time\_k} \tag{28}$$

where PC is the penalty factor, N is the number of current UAV, and N is the number of points in the flight path generated by the B-spline curve. Δ_{time_k} is the time difference between the two UAVs when they arrive at the point where a possible collision could occur.

3.2.2. Application of MSFOA to Multi-UAVs Path Planning

(1) Collision detection method

Collision avoidance is an important part of multi-UAVs path planning. Due to the huge space in the 3D map, this method only detects the possibility of collision after the path is generated. When the

n-th UAV's flight path is going to check for collision, this method will check it in pairs with the previous $n-1$ UAV's flight path. The n-th path is called the path to be detected, and the previous $n-1$ is called the air_path_{n-1}. For each point on the detection path, the distance from each point between the detection path and air_path_{n-1} is calculated. If the distance is less than the minimum safe distance, the two points will be called a possible collision point, and the path will be punished. The mathematical descriptions are as follows:

$$JP = \sum_{m=1}^{n} \sum_{i=1}^{n} JP_i$$
$$\text{with} \quad \Delta_{time\_k} = \begin{cases} C * {t_{crash}}^3, & t_{crash} < 0.5 \\ 0, & t_{crash} \geq 0.5 \end{cases} \tag{29}$$

where JP_i is the collision risk between the i-th UAVs and the m-th UAVs, the parameter C is the time penalty factor, and t_{carsh} is the time difference between the two UAVs at the point of possible collision, determined by the following formula:

$$t_{carsh} = v/(dis_1 - dis_2) \tag{30}$$

(2) MSFOA based multi-UAVs path planning

In order to further explain the MSFOA based path planning, the specific steps of multi-UAVs path planning for MSFOA is elaborated. The algorithm records the location of the d-th control point of the i-th individual fruit fly in swarm g with $X_i^{(g),\,d}$ and $Y_i^{(g),\,d}$, $d = 1, \ldots, D$, where D denotes the number of control points.

$$S_i^{(g),\,d} = \sqrt{{(X_i^{(g),d})}^2 + (Y_i^{(g),d})^2} \tag{31}$$

where $S_i^{(g),\,j}$ is the smell concentration judgment. The path is generated by

$$\begin{cases} path_i^{(x)}[k] = S_i^{(g),\,k} * P_{max}^{(x)} \\ path_i^{(y)}[k] = S_i^{(g),\,k+D} * P_{max}^{(y)} \\ path_i^{(z)}[k] = S_i^{(g),\,k+2*D} * P_{max}^{(z)} \end{cases} \tag{32}$$

where $P_{max}^{(x)}$, $P_{max}^{(y)}$, $P_{max}^{(z)}$ is the boundary of the map in three dimensions. $path_i^{(x)}$ is the coordinate on the X axis of all the control points in the path, $path_i^{(y)}$ is the coordinate on the Y axis of all the control points in the path, and $path_i^{(z)}$ is the coordinate on the Z axis of all the control points along this path. k represents the number of control points, where k values range from 2 to D–1. Then add the starting and ending points:

$$\begin{cases} path_i^{(x)}[1] = X_{S,N} \\ path_i^{(y)}[1] = Y_{S,N} \\ path_i^{(z)}[1] = Z_{S,N} \\ path_i^{(x)}[D+2] = X_T \\ path_i^{(y)}[D+2] = Y_T \\ path_i^{(z)}[D+2] = Z_T \end{cases} \tag{33}$$

where $X_{S,N}$, $Y_{S,N}$, $Z_{S,N}$ is the beginning of the path of the N-th UAV, and X_T, Y_T, Z_T is the end of the path of the UAV. There is no essential difference between the multi-origin and multi-destination model in this paper. The MSFOA for multi-UAVs path planning is implemented as follows:

Step 1: Determine information of the UAVs mission, including starting coordinates $(X_{S,\,N}, Y_{S,\,N}, Z_{S,\,N})$ and ending coordinates (X_T, Y_T, Z_T). The number of UAVs required for the

mission Num, maximum velocity v_{max}, maximum turn overload n_{max}, and minimum altitude h_{min}. Mission map information p, boundary P^x_{max}, P^y_{max}, P^z_{max}, and the number D of control points that each UAV needs to plan out.

Step 2: Identify enemy ground weapons, including threat types such as radar, missiles, anti- aircraft guns, weapon positions ($X_{threat,\,j}$, $Y_{threat,\,j}$), and their respective threat ranges.

Step 3: Set up the parameters of MSFOA, including maximum iteration number NC, the number of new swarms G, the fruit fly population size M_{pop}, the *threshold*, the searching step dwindled parameter R, and cross genetic coefficient coe_1 and coe_2.

Step 4: Let $num = 1$.

Step 5: Let $nc = 1$ and randomly generate the location for each swarm, $(x^{(1),1}_{\text{best}}, y^{(1),1}_{\text{best}}) \dots (x^{(1),\,D}_{\text{best}}, y^{(1),\,D}_{\text{best}}) \dots (x^{(G),1}_{\text{best}}, y^{(G),1}_{\text{best}}) \dots (x^{(G),\,D}_{\text{best}}, y^{(G),\,D}_{\text{best}})$.

Step 6: Let $g = 1$, $i = 1$, $d = 1$.

Step 7: Generate the random locations for the food source using the olfactory according to Equation (34) if $smell^{(g)}_{\text{best}}$ is larger; otherwise, generate the new locations according to Equation (35), as follows

$$X^{(g),\,d}_i = x^{(g),\,d}_{\text{best}} + \sin((\pi/2) * \text{random}(-1,1)) Y^{(g),\,d}_i = y^{(g),\,d}_{\text{best}} + \sin((\pi/2) * \text{random}(-1,1)) \tag{34}$$

$$X^{(g),\,d}_i = x^{(g),\,d}_{\text{best}} + R * \text{random}(-1,1) Y^{(g),\,d}_i = y^{(g),\,d}_{\text{best}} + R * \text{random}(-1,1) \tag{35}$$

Step 8: Update the smell concentration judgement according to the formula below:

$$S^{(g),\,d}_{\text{old},\,i} = \sqrt{\left(X^{(g),\,d}_i\right)^2 + \left(Y^{(g),\,d}_i\right)^2} \tag{36}$$

if $S^{(g),\,d}_{\text{old},\,i}$ is bigger than 1, jump to step 7, otherwise jump to step 9.

Step 9: if $d < D$, let $d = d + 1$, jump to step 7, otherwise jump to step 10.

Step 10: Generate the path of this fruit fly $path_i$ using Equations (31)–(33). Update smell concentration by the formula below

$$smell^{(g)}_{\text{old},\,i} = J(pathi) \tag{37}$$

where J is described in Equation (20). If $i < M_{pop}/G$, then let $i = i + 1$, $d = 1$ and jump to step 7, otherwise jump to step 11.

Step 11: Find out the old best in the g-th swarm $(x^{(g),1}_{\text{oldbest}}, y^{(g),1}_{\text{oldbest}}), \dots, (x^{(g),\,D}_{\text{oldbest}}, y^{(g),\,D}_{\text{oldbest}})$ using Equations (12) and (13). If $g < G$, then let $g = g + 1$, $d = 1$, $i = 1$ and jump to step 7, otherwise jump to step 12.

Step 12: Let $g = 1$, $i = 1$, $d = 1$.

Step 13: Generate fruit fly in new swarms according to the equation below:

$$\begin{aligned} X^{(g),\,d}_{\text{new},\,i} &= coe_1 * x^{\text{random}(1,G),\,d}_{\text{oldbest}} + coe_2 * X^{(g),\,d}_i \\ Y^{(g),\,d}_{\text{new},\,i} &= coe_1 * y^{\text{random}(1,G),\,d}_{\text{oldbest}} + coe_2 * Y^{(g),\,d}_i \end{aligned} \tag{38}$$

Step 14: Update the smell concentration judgement according to the equation below:

$$S^{(g),\,d}_{\text{new},\,i} = \sqrt{\left(X^{(g),\,d}_{\text{new},\,i}\right)^2 + \left(Y^{(g),\,d}_{\text{new},\,i}\right)^2} \tag{39}$$

if $S^{(g),\,d}_{\text{new},\,i}$ is bigger than 1, jump to step 6, otherwise jump to step 8.

Step 15: if $d < D$, let $d = d + 1$, jump to step 13, otherwise jump to step 16.

Step 16: Generate the path of this fruit fly $path_i$ by using Equations (31)–(33). Update smell concentration by the equation below:

$$smell^{(g)}_{\text{new},\,i} = J(pathi) \tag{40}$$

If $i < M_{pop}/G$, then let $i = i + 1$, $d = 1$ and jump to step 13, otherwise jump to step 17.

Step 17: Find out the index of new best in the g-th new swarm $Index^{(g)}_{\text{new}}$ by Equation (17).

Step 18: All the fruit flies in the g-th swarm fly to the location with the minimum smell concentration according to Equations (18) and (19). If $g < G$, then let $g = g + 1$, $d = 1$, $i = 1$ and jump to step 13, otherwise jump to step 19.

Step 19: If $nc < NC$, let $nc = nc + 1$ and jump to step 5, otherwise jump to step 20.

Step 20: If $num < Num$, generate and output the path of the num-th UAV then let $num = num + 1$ and jump to step 5; otherwise, end and output the UAV paths.

4. Results and Discussion

This section discusses the simulations used to assess the performance of our proposed algorithm in solving the UAV path planning problem. Seven algorithms are tested under three scenarios, including MSFOA, FOA, PSO, DE, ABC, IFFO, and MFOA. The 3D stereo displays of three scenarios are shown in Figure 1. The algorithm is coded using Matlab-2018b. Owing to their stochastic nature, evolutionary algorithms may arrive at better or worse solutions than they previously reached during their search for new solutions. For this reason, every tested algorithm is run 40 times independently for each scenario and the statistical results are used for performance evaluation and comparison. We coded the above algorithms and set the parameters according to their references. The main parameters of the seven tested algorithms are detailed in Table 1. The best values of each scenario are highlighted using boldface. The parameter configurations are all based on the suggestions in the corresponding references; where these values have a fair comparison, the same maximum iteration number is used for all algorithms as the stopping criterion, while the population size is set as 100 for all algorithms.

Table 1. The parameters used in experiments of different algorithms.

Algorithms	Population Size	Control Parameters
MSFOA	$M_{pop} = 100$	The swarm amount $G = 5$, the mutation parameters $coe_1 = 0.8$, $coe_2 = 0.2$.
FOA [18]	$M_{pop} = 100$	No more parameters.
PSO [13]	$M_{par} = 100$	The inertial weight coefficient w varies from 0.9 to 0.4, the acceleration coefficients $c_1 = c_2 = 2$.
DE [10]	$M_{sol} = 100$	The scaling factor F is set as a random value in [0.2, 0.9], the crossover factor $cr = 0.9$.
ABC [11]	$N_e = 50$, $N_u = 50$	The largest local searching times $Limit = 20$.
IFFO [24]	$M_{pop} = 100$	The maximum radius $\lambda_{max} = 0.5$, the minimum radius $\lambda_{\min} = 0.01$.
MFOA [23]	$M_{pop} = 100$	The multi-swarm amount $M = 5$.

Abbreviations: MSFOA, multiple swarm fruit fly optimization algorithm; FOA, fruit fly optimization algorithm; PSO, particle swarm optimization; DE, differential evolution; ABC, artificial bee colony; IFFO, improved fruit fly optimization; MFOA, multi-swarm fruit fly optimization algorithm.

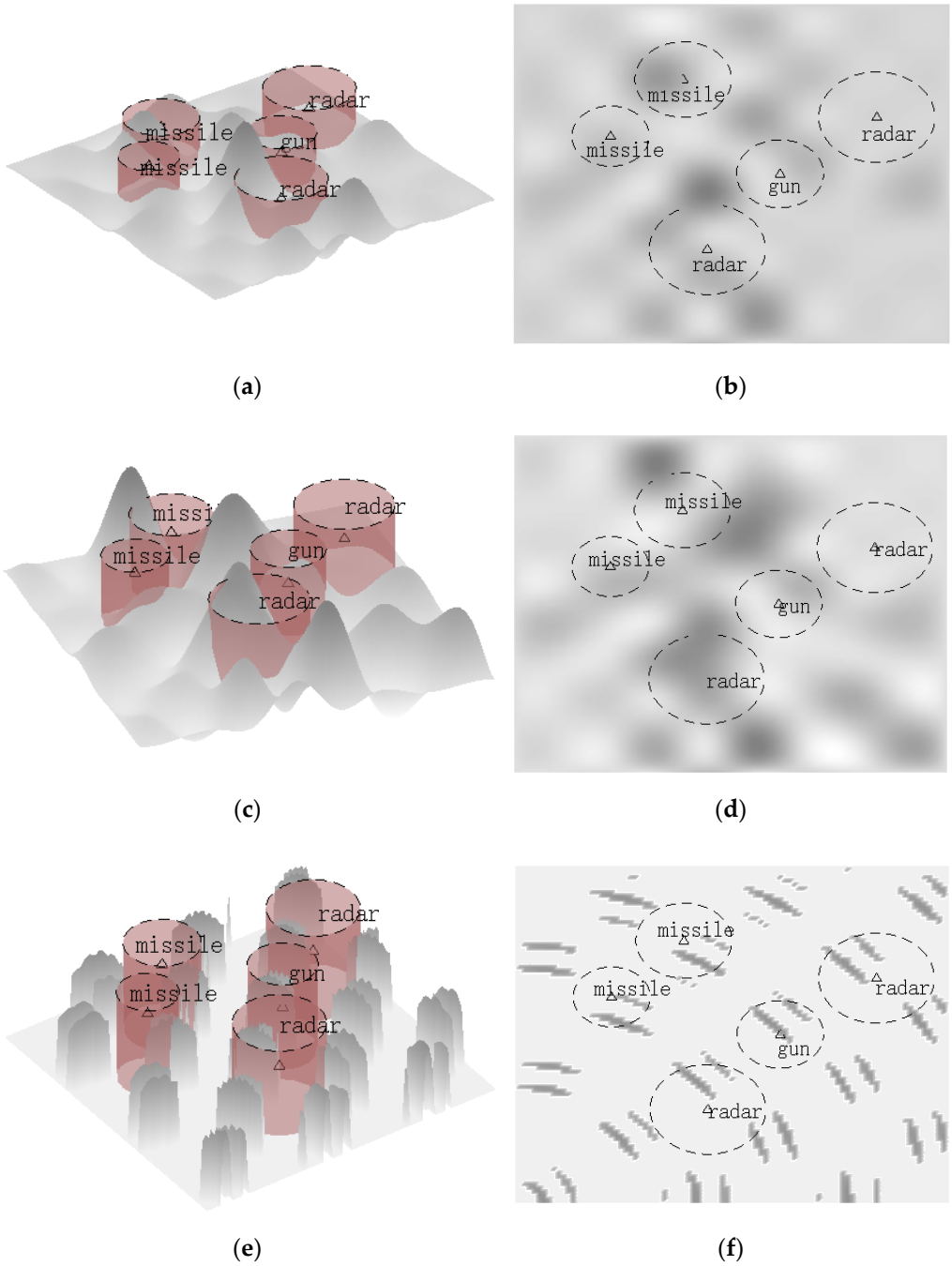


Figure 1. (**a**,**c**,**e**) are the three-dimensional (3D) displays of the three simulation scenarios, which show the terrain and threat sources in the mission map; (**b**,**d**,**f**) are the corresponding 2D views of the simulation scenarios.

The 3D stereo displays of the seven UAV paths in the three scenarios are shown in Figures 2–4, where (a)–(g) correspond to MSFOA, FOA, PSO, DE, ABC, IFFO, and MFOA, respectively. The statistical results of MSFOA, FOA, PSO, DE, ABC, IFFO, and MFOA during 40 runs on the three different scenarios are listed in Table 2, where the best of the cost values of each UAV are recorded.

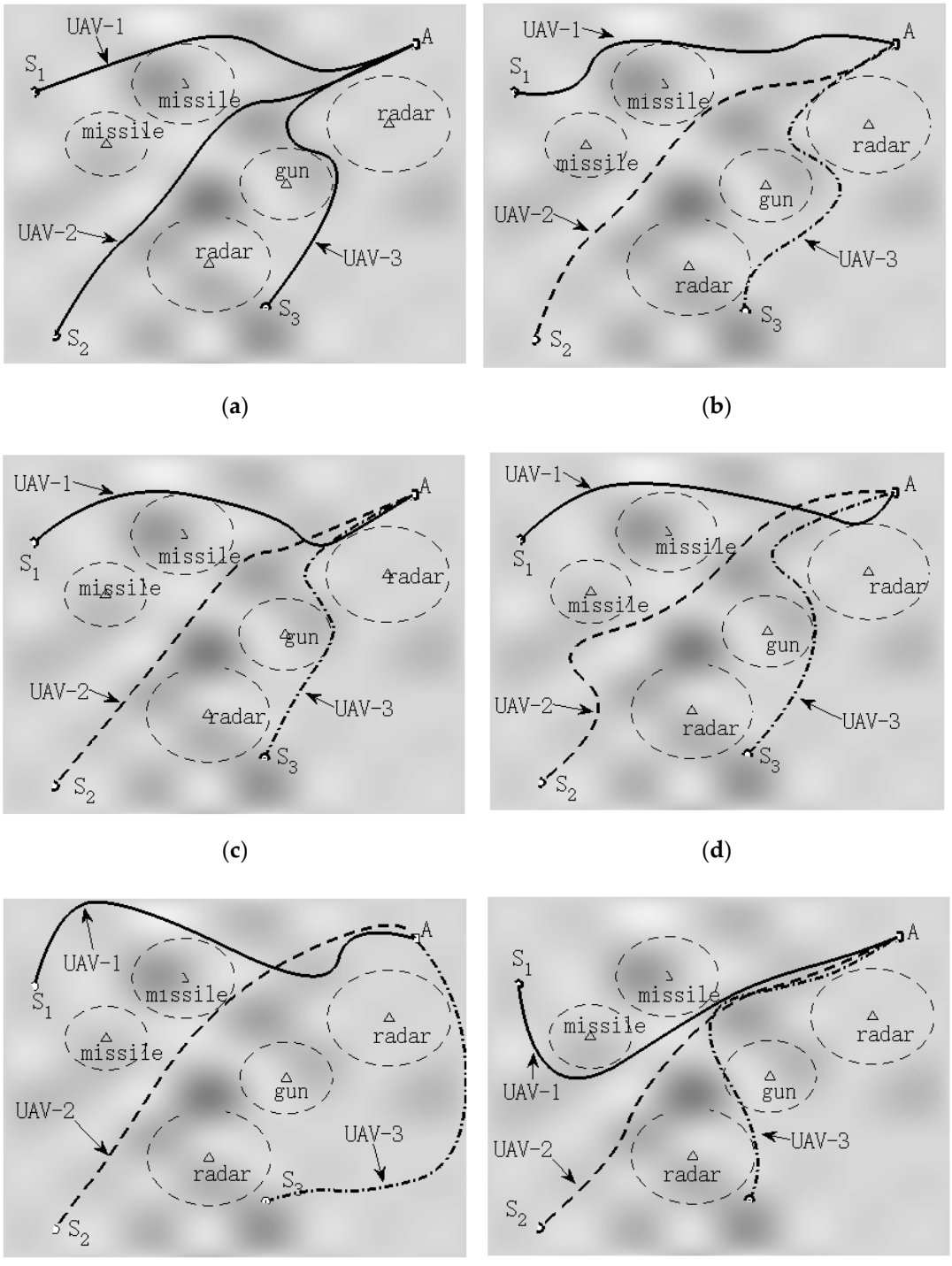
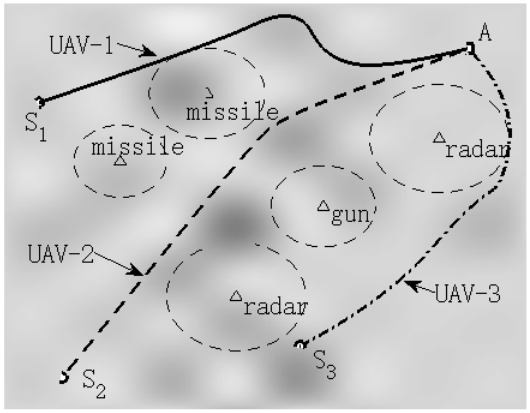
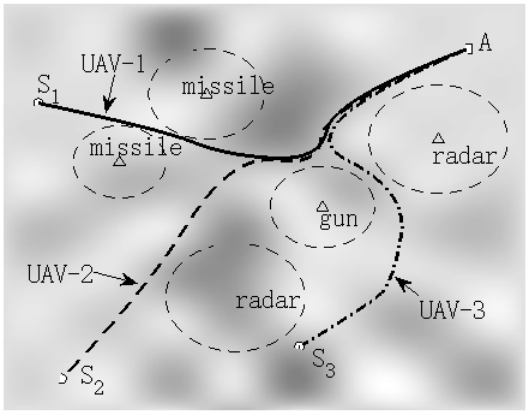


Figure 2. *Cont*.

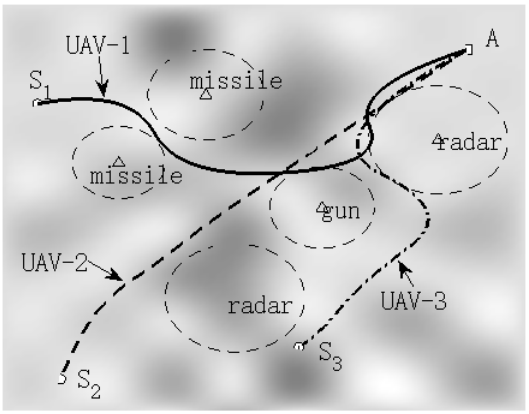


(**g**)

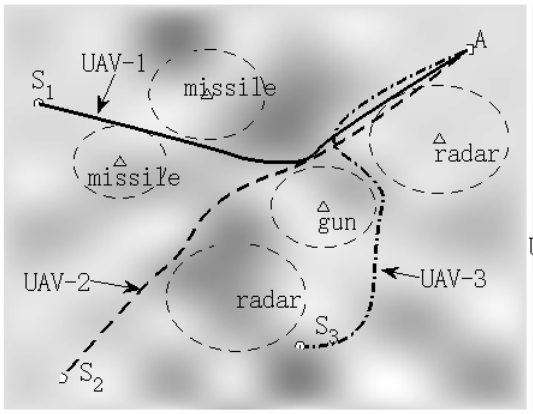
Figure 2. Two-dimensional displays of the best unmanned aerial vehicle (UAV) paths obtained by (**a**) MSFOA, (**b**) FOA, (**c**) PSO, (**d**) DE, (**e**) ABC, (**f**) IFFO, and (**g**) MFOA in the first scenario.



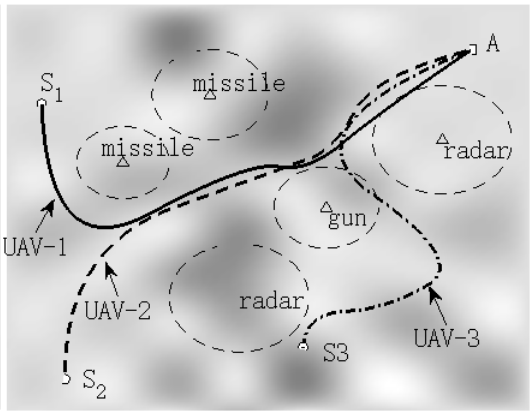
(**a**)



(**b**)



(**c**)



(**d**)

Figure 3. *Cont*.

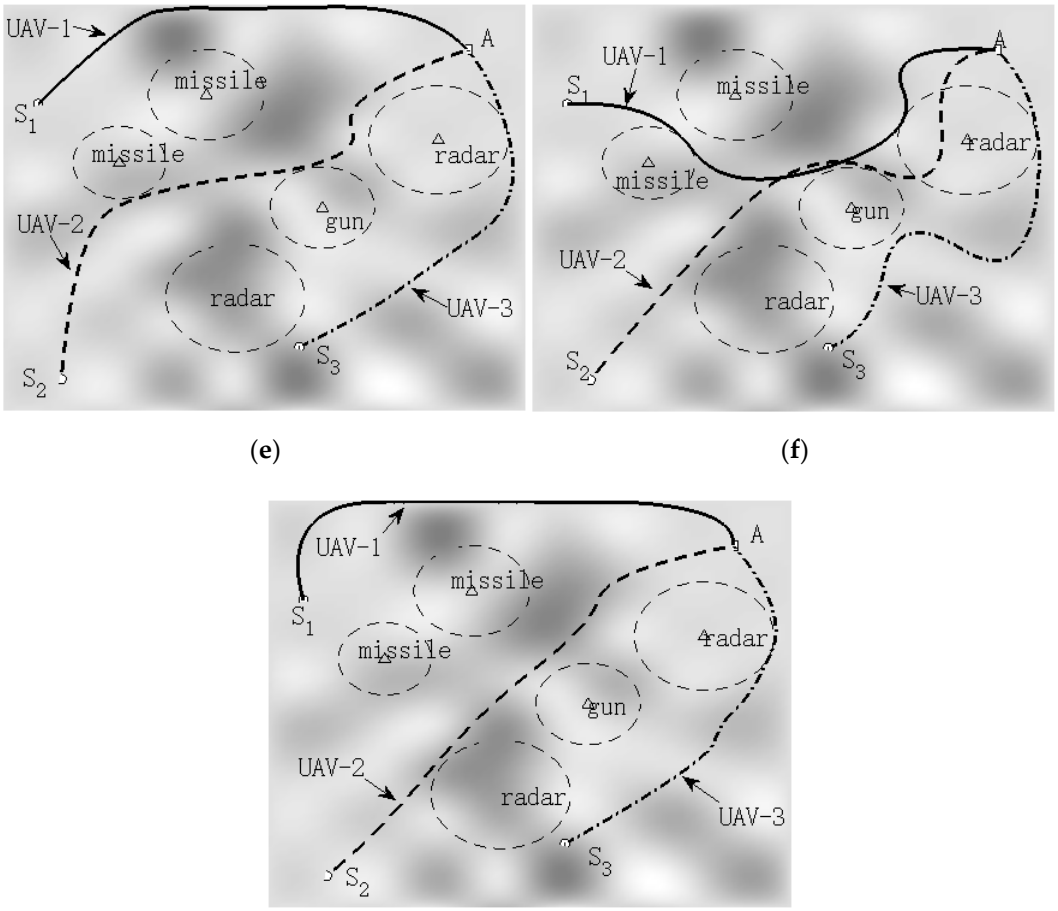


Figure 3. Two-dimensional displays of the best UAV paths obtained by (**a**) MSFOA, (**b**) FOA, (**c**) PSO, (**d**) DE, (**e**) ABC, (**f**) IFFO, and (**g**) MFOA in the second scenario.

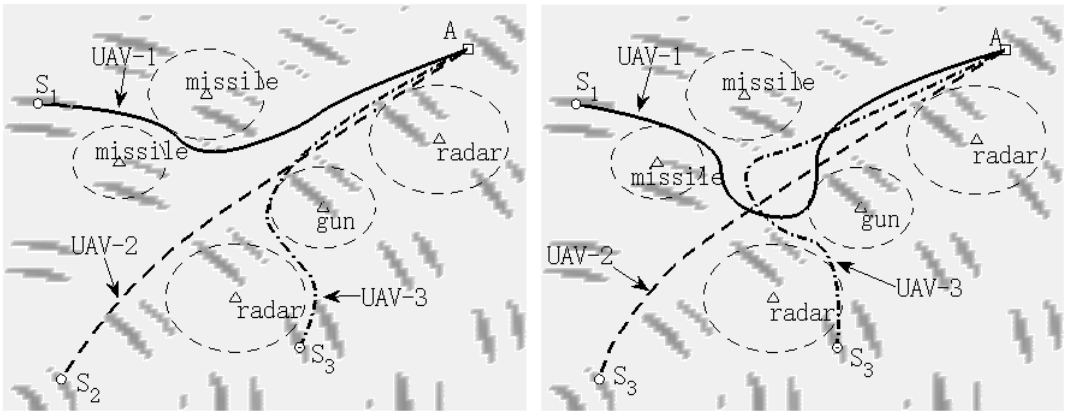


Figure 4. *Cont*.

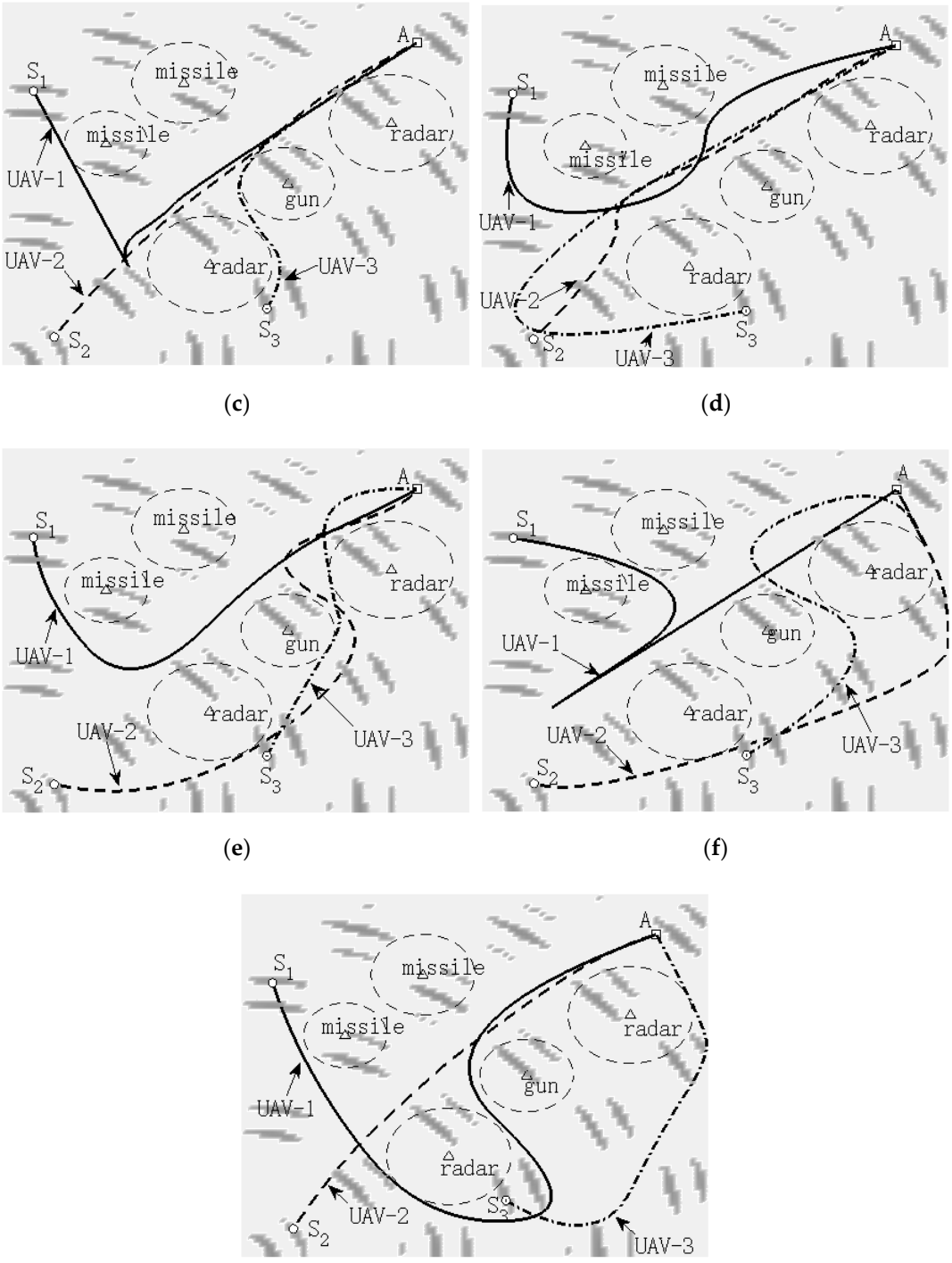


Figure 4. Two-dimensional displays of the best UAV paths obtained by (**a**) MSFOA, (**b**) FOA, (**c**) PSO, (**d**) DE, (**e**) ABC, (**f**) IFFO, and (**g**) MFOA in the third scenario.

Table 2. The objective function of each UAV obtained by various algorithms in the three scenarios.

Algorithm	First Scenario				Second Scenario				Third Scenario			
	J_{UAV_1}	J_{UAV_2}	J_{UAV_3}	$\sum_{n=1}^{3} J_{UAV\_n}$	J_{UAV_1}	J_{UAV_2}	J_{UAV_3}	$\sum_{n=1}^{3} J_{UAV\_n}$	J_{UAV_1}	J_{UAV_2}	J_{UAV_3}	$\sum_{n=1}^{3} J_{UAV\_n}$
MSFOA	133.21	138.09	145.66	**404.37**	128.79	125.52	149.90	**382.95**	152.48	145.19	156.85	**440.71**
FOA	165.22	168.24	173.69	507.15	155.16	293.51	661.07	520.27	307.26	244.29	178.91	730.46
PSO	146.27	137.72	201.16	485.14	123.72	112.26	171.77	520.27	228.07	238.56	162.09	628.71
DE	158.20	184.94	218.39	561.53	232.28	235.97	187.56	655.81	207.78	244.22	212.59	664.59
ABC	113.07	142.89	155.21	411.16	139.17	112.85	148.99	401.02	165.08	125.62	166.76	457.45
IFFO	181.34	146.80	169.14	497.29	134.56	154.59	258.29	547.44	229.20	165.71	194.94	589.85
MFOA	116.54	155.41	145.11	417.07	126.34	109.93	152.06	388.33	167.12	146.68	152.06	465.87

For the first scenario, all seven tested algorithms find the safe flight path for UAVs. It can be seen that the best path generated by MSFOA flies the smoothest. In comparing these results in the first scenario column in Table 2, it is observed that the MSFOA obtains the minimum best among the algorithms, which indicates that MSFOA has the most powerful optimization ability in the statistical sense. Figure 3 displays the experimental results of the second test scenario. For this test scenario, DE and IFFO come into the threat areas. From Table 2, it can be seen that MSFOA achieves the smallest mean cost value. Furthermore, the result of the third scenario is shown in Figure 4. It is obvious that MSFOA has the best path for UAVs. In the third scenario column of Table 2, the cost of MSFOA is significantly decreased in comparison to the other algorithms.

Above all, these experimental results demonstrate that the MSFOA still maintains a higher performance than the FOA, PSO, DE, ABC, and two other modified FOA versions.

The convergence curves of the average best cost values are displayed in Figures 5–7. In the first scenario, it can be seen that DE, FOA, and IFFO show inferior convergence. MFOA achieves a faster convergence speed and smaller cost than MSFOA in the early stages. However, in the later iterations when most algorithms come to stagnation, MSFOA still shows the ability to search for better solutions. The searching range decreases as the cost value decreases below the threshold, which can improve the local searching ability near the optimal solution, and thus leads to the best global convergence of the algorithms.

In Table 3, the best, median, and average of the cost values of the sum for three UAVs are recorded. The best values are highlighted using boldface. From this table, it can be seen that MSFOA achieves the smallest cost value in the first and third scenario and the second smallest in the second scenario. MFOA obtains a smaller average and median in the second scenario, but the distributed interval of solutions obtained by MFOA in Figure 3 is evidently worse than that obtained by MSFOA. Comprehensively comparing the statistical data of all algorithms in the three scenarios, MSFOA still shows its superiority to other algorithms in terms of searching ability and stability.

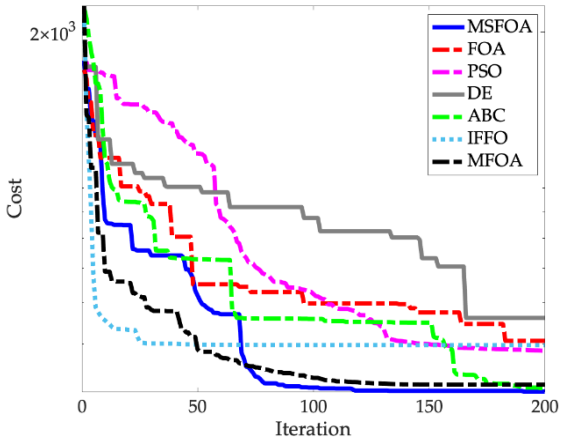


Figure 5. Convergence curves of the average best values for the first scenario.

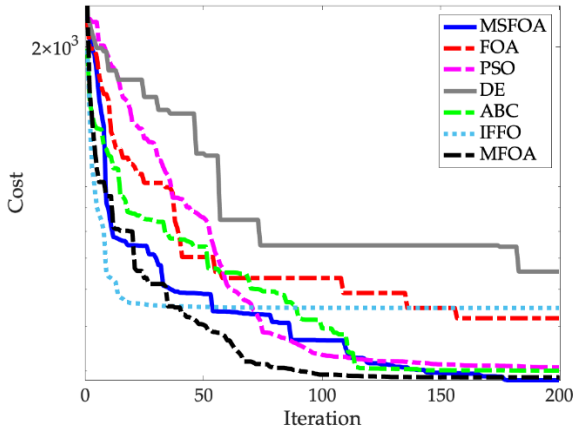


Figure 6. Convergence curves of the average best values for the second scenario.

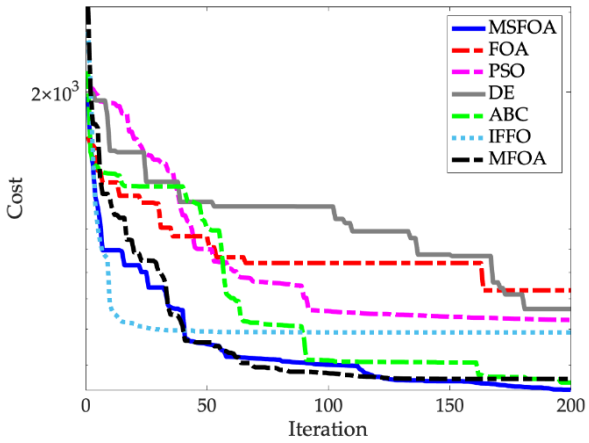


Figure 7. Convergence curves of the average best values for the third scenario.

Table 3. The statistical results of cost of the path in different scenarios.

Algorithm	First Scenario			Second Scenario			Third Scenario		
	Best	**Average**	**Median**	**Best**	**Average**	**Median**	**Best**	**Average**	**Median**
MSFOA	**404.37**	**490.23**	**466.56**	**382.95**	485.46	505.18	**440.71**	**504.20**	**492.95**
FOA	507.15	708.73	698.53	520.27	755.35	693.81	730.46	1001.69	993.81
PSO	485.14	797.76	754.04	520.27	755.35	693.81	628.71	953.93	954.92
DE	561.53	674.52	665.62	655.81	761.37	768.37	664.59	809.74	815.48
ABC	411.16	503.64	497.22	401.02	516.94	520.93	457.45	589.69	591.39
IFFO	497.29	859.11	860.97	547.44	928.37	908.26	589.85	953.56	934.38
MFOA	417.07	495.04	493.33	388.33	**482.81**	**475.26**	465.87	566.83	564.83

5. Conclusions

This paper proposed an MSFOA for cooperative path planning of multi-UAVs over three-dimensional rugged terrain. In order to ensure the convergence speed and avoid local optimization, we have improved this in many ways. The multi-swarm strategy is used to increase the searching ability of the algorithm. In order to make use of any searching information effectively, we proposed the competition strategy of offspring, which ensures the efficient use of searching information and avoids falling into local optimum. In order to plan the flight path for multi-UAVs and avoid the collision between UAVs, we proposed a method to detect the collision between UAVs. Several simulations have shown the effective performance of the proposed approach. This paper proposed a method which can be used for multi-UAV path planning and can solve the high-dimensional function optimization problem. It can be widely used in the field of robotics. In addition, the parameters

and sub-swarm amount usually affect the searching performance, and this may be improved in further work.

Author Contributions: K.S. and X.Z. proposed the algorithm and wrote the paper, as well as analyzed the data; K.S. and S.X. conceived and performed the experiments. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the Beijing Municipal Education Commission, the National Natural Science Foundation of China (No. 61703012), and the Beijing Natural Science Foundation (No. 4182010).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

$x_{\text{best}}^{(g)}$	The x coordinate of the location of the g-th old swarm.
$y_{\text{best}}^{(g)}$	The y coordinate of the location of the g-th old swarm.
$X_i^{(g)}$	The x coordinate of the i-th fruit fly in the g-th old swarm.
$Y_i^{(g)}$	The y coordinate of the i-th fruit fly in the g-th old swarm.
$S_{\text{old},\,i}^{(g)}$	The smell concentration judgement of the i-th fruit fly in the g-th old swarm.
$smell_{\text{old},\,i}^{(g)}$	The smell concentration of the i-th fruit fly in the g-th old swarm.
$Index_{\text{old}}^{(g)}$	The index of the fly with the best smell concentration in the g-th old swarm.
$x_{\text{oldbest}}^{(g)}$	The x coordinate of the fly with the best smell concentration in the g-th old swarm.
$y_{\text{oldbest}}^{(g)}$	The y coordinate of the fly with the best smell concentration in the g-th old swarm.
$X_{\text{new},\,i}^{(g)}$	The x coordinate of the i-th fruit fly in the g-th new swarm.
$Y_{\text{new},\,i}^{(g)}$	The y coordinate of the i-th fruit fly in the g-th new swarm.
$S_{\text{new},\,i}^{(g)}$	The smell concentration judgement of the i-th fruit fly in the g-th new swarm.
$smell_{\text{new},\,i}^{(g)}$	The smell concentration of the i-th fruit fly in the g-th new swarm.
$Index_{\text{new}}^{(g)}$	The index of the fly with the best smell concentration in the g-th new swarm.
G	The number of swarms.

References

1. Zhang, X.Y.; Lu, X.Y.; Jia, S.; Li, X.Z. A novel phase angle-encoded fruit fly optimization algorithm with mutation adaptation mechanism applied to UAV path planning. *Appl. Soft Comput.* **2018**, *70*, 371–388. [CrossRef]
2. Zheng, C.W.; Li, L.; Xu, F.J.; Sun, F.C.; Ding, M.Y. Evolutionary route planner for unmanned air vehicles. *IEEE Trans. Robot.* **2005**, *21*, 609–620. [CrossRef]
3. Zhu, M.N.; Zhang, X.H.; Luo, H.; Wang, G.Q.; Zhang, B.B. Optimization dubins path of multiple UAVs for post-earthquake rapid-assessment. *Appl. Sci.* **2020**, *10*, 1388. [CrossRef]
4. Wang, X.D.; Luo, X.; Han, B.L.; Chen, Y.H.; Liang, G.H.; Zheng, K.L. Collision-free path planning method for robots based on an improved rapidly-exploring random tree algorithm. *Appl. Sci.* **2020**, *10*, 1381. [CrossRef]
5. Roberge, V.; Tarbouchi, M.; Labonte, G. Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning. *IEEE Trans. Ind. Inform.* **2013**, *9*, 132–141. [CrossRef]
6. Sababha, M.; Zohdy, M.; Kafafy, M. The Eehanced firefly algorithm based on modified exploitation and exploration mechanism. *Electronics* **2018**, *7*, 132. [CrossRef]
7. Richter, C.; Bry, A.; Roy, N. Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments. *Springer Tracts Adv. Robot.* **2016**, *114*, 649–666.
8. Li, Z.H.; Yang, X.J.; Sun, X.D.; Liu, G.; Hu, C. Improved artificial potential field based lateral entry guidance for waypoints passage and no-fly zones avoidance. *Aerosp. Sci. Technol.* **2019**, *86*, 119–131. [CrossRef]
9. Xue, Y.; Sun, J.Q. Solving the path planning problem in mobile robotics with the multi-objective evolutionary algorithm. *Appl. Sci.* **2018**, *8*, 1425. [CrossRef]
10. Zhang, X.Y.; Duan, H.B. An improved constrained differential evolution algorithm for unmanned aerial vehicle global route planning. *Appl. Soft Comput.* **2015**, *26*, 270–284. [CrossRef]

11. Xu, C.F.; Duan, H.B.; Liu, F. Chaotic artificial bee colony approach to uninhabited combat air vehicle (UCAV) path planning. *Aerosp. Sci. Technol.* **2010**, *14*, 535–541. [CrossRef]
12. Zhang, X.Y.; Jia, S.M.; Li, X.Z.; Jian, M. Design of the fruit fly optimization algorithm based path planner for UAV in 3D environments. In Proceedings of the 2017 IEEE International Conference on Mechatronics and Automation, Takamatsu, Japan, 6–9 August 2017; pp. 381–386.
13. Das, P.K.; Behera, H.S.; Das, S.; Tripathy, H.K.; Panigrahi, B.K.; Pradhan, S.K. A hybrid improved PSO-DV algorithm for multi-robot path planning in a clutter environment. *Neurocomputing* **2016**, *207*, 735–753. [CrossRef]
14. Dorigo, M.; Maniezzo, V.; Colorni, A. Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Trans. Syst.* **1996**, *26*, 29–41. [CrossRef] [PubMed]
15. Duan, H.B.; Zhang, X.Y.; Wu, J.; Ma, G.J. Max-min adaptive ant colony optimization approach to multi-UAVs coordinated trajectory replanning in dynamic and uncertain environments. *J. Bionic Eng.* **2009**, *6*, 161–173. [CrossRef]
16. Masdari, M.; Salehi, F.; Jalali, M.; Bidaki, M. A survey of PSO-based scheduling algorithms in cloud computing. *J. Netw. Syst. Manag.* **2017**, *25*, 122–158. [CrossRef]
17. Das, P.K.; Behera, H.S.; Panigrahi, B.K. A hybridization of an improved particle swarm optimization and gravitational search algorithm for multi-robot path planning. *Swarm Evol. Comput.* **2016**, *28*, 14–28. [CrossRef]
18. Pan, W.T. A new fruit fly optimization algorithm: Taking the financial distress model as an example. *Knowl. Based Syst.* **2012**, *26*, 69–74. [CrossRef]
19. Li, W.T.; Zhang, Y.D.; Shi, X.W. Advanced fruit fly optimization algorithm and its application to irregular subarray phased array antenna synthesis. *IEEE Access* **2019**, *7*, 165583–165596. [CrossRef]
20. Li, H.Z.; Guo, S.; Li, C.J.; Sun, J.Q. A hybrid annual power load forecasting model based on generalized regression neural network with fruit fly optimization algorithm. *Knowl. Based Syst.* **2013**, *37*, 378–387. [CrossRef]
21. Lin, S.M. Analysis of service satisfaction in web auction logistics service using a combination of fruit fly optimization algorithm and general regression neural network. *Neural Comput. Appl.* **2013**, *22*, 783–791. [CrossRef]
22. Xing, Y.F. Design and optimization of key control characteristics based on improved fruit fly optimization algorithm. *Kybernetes* **2013**, *42*, 466–481. [CrossRef]
23. Yuan, X.F.; Dai, X.S.; Zhao, J.Y.; He, Q. On a novel multi-swarm fruit fly optimization algorithm and its application. *Appl. Math. Comput.* **2014**, *233*, 260–271. [CrossRef]
24. Pan, Q.K.; Sang, H.Y.; Duan, J.H.; Gao, L. An improved fruit fly optimization algorithm for continuous function optimization problem. *Knowl. Based Syst.* **2014**, *62*, 69–83. [CrossRef]