sensors

MDPI

Article

Intelligent Beetle Antennae Search for UAV Sensing and Avoidance of Obstacles

Qing Wu [1], Xudong Shen [1], Yuanzhe Jin [2], Zeyu Chen [1], Shuai Li [3] and Ameer Hamza Khan [3] and Dechao Chen [1,*]

[1] School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China; wuqing@hdu.edu.cn (Q.W.); 172050050@hdu.edu.cn (X.S.); zeno181050017@hdu.edu.cn (Z.C.)
[2] Department of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310018, China; 3150100835@zju.edu.cn
[3] Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong 999077, China; shuaili@polyu.edu.hk (S.L.); ameer.h.khan@connect.polyu.hk (A.H.K.)
* Correspondence: chdchao@hdu.edu.cn; Tel.: +86-13777481992

Received: 22 February 2019; Accepted: 10 April 2019; Published: 12 April 2019

Abstract: Based on a bio-heuristic algorithm, this paper proposes a novel path planner called obstacle avoidance beetle antennae search (OABAS) algorithm, which is applied to the global path planning of unmanned aerial vehicles (UAVs). Compared with the previous bio-heuristic algorithms, the algorithm proposed in this paper has advantages of a wide search range and breakneck search speed, which resolves the contradictory requirements of the high computational complexity of the bio-heuristic algorithm and real-time path planning of UAVs. Besides, the constraints used by the proposed algorithm satisfy various characteristics of the path, such as shorter path length, maximum allowed turning angle, and obstacle avoidance. Ignoring the z-axis optimization by combining with the minimum threat surface (MTS), the resultant path meets the requirements of efficiency and safety. The effectiveness of the algorithm is substantiated by applying the proposed path planning algorithm on the UAVs. Moreover, comparisons with other existing algorithms further demonstrate the superiority of the proposed OABAS algorithm.

Keywords: UAVs; path planning; obstacle avoidance; MTS; optimization algorithms

1. Introduction

Unmanned aerial vehicles (UAVs) are increasingly being used in military and civilian environments to perform critical tasks [1,2] because of their ability to complete missions under dangerous conditions or extreme weather [3]. UAVs are often better suited to perform dangerous missions and require real-time updates to avoid drones being discovered or crashed, so we chose drones as our experimental subjects. Computationally efficient planning that generates high quality paths is important for UAVs, which are time-varying nonlinear dynamic systems [4]. Since the moving aircraft and vehicles have high-speed dynamics [5,6], it is an essential requirement to meet the low complexity of planning [7], i.e., to plan the required path in a short time.

UAV path planning can be divided into five types: sampling methods, node-based, mathematical models, bio-heuristic algorithms and multi-fusion algorithms. Sampling based methods include probabilistic roadmaps (PRM) [8], Voronoi maps [9], corridor map [10] and artificial potential field (APF) [11]. These types of algorithms usually need to pre-process the map, grid or sample the map, and then randomly search for paths. These types of path planning algorithms have a time complexity of $O(n \log n)$ to $O(n^2)$, which is reasonably fast and can be applied to real-time path planning. Node-based path planning algorithms methods include Dijkstra [12], A* [8], lifelong planning

algorithm (LPA) [13], harmony search [14], and theta star [15]. These types of path planning have a time complexity between $O(\log n)$ and $O(n^2)$. Due to the low time complexity of these algorithms, real-time path planning is possible. Path planning algorithms based on mathematical models have mixed integer algorithms [16], binary linear algorithm [17] and nonlinear algorithm [18]. The time complexity of such algorithms is described by a polynomial equation and are generally time-consuming, therefore most of these types of algorithms are used in offline planning. The fourth category is bio-heuristic algorithms, including neural networks (NN) [19], genetic algorithm (GA) [20], particle swarm optimization (PSO) [21], ant colony optimization (ACO) algorithm [22] and hybrid leapfrog optimization [23]. These types of path planning have a time complexity of $O(n^2)$, which can only be applied in static environments for offline path planning. The last category is multi-fusion based algorithms, including PRM node based on optimal algorithm [8], geography informations system (GIS) based 3D path planning algorithm [24], visibility node based on optimal algorithm [25] and 3D path planning robust algorithm [26]. Their time complexity is greater than $O(n \log n)$. Their low time complexity makes them applicable to online path planning.

The heuristic algorithm can solve the NP-hard problem that traditional linear programming can't solve. We found that heuristic algorithms are often applied to static path planning. Because heuristic algorithms take a long time, it is difficult to cope with real-time dynamic path planning scenarios. Therefore, a bio-heuristic algorithm with fast computation speed and low computational complexity has positive significance for solving the path planning problem of UAV.

1.1. Related Work

The path planning of the UAVs needs to meet multiple constraints in a complex environment, and the purpose of these constraints is to enable UAVs to find short and safe paths efficiently. Bio-heuristic algorithms are the methods inspired from biological organisms to solve a problem. People have summarized the behavioral characteristics of organisms that have been continuously evolved by natural selection, so this type of algorithms are efficient.

At present, the primary investment sources and applications of UAVs are in the field of national defense, but they also have great potential for development in the civilian sector. Kroumov et al. [27] proposed a novel potential field based 3D path planning technique for differential drive mobile robots, moving in known environment. Passino [28] provides a tutorial on biology of bacterial foraging optimization (BFO), including an overview of the biology of bacterial foraging. Yang et al. [29] propose a new metaheuristic method, the bat algorithm (BA), based on the echolocation behaviour of bats. These algorithms have strong search performance but have the downside of large time-consumption. Li et al. [30] proposed distributed recurrent neural networks for cooperative control of manipulators. Gawel et al. [31] use UAVs to perform aerial data mapping on heterogeneous grounds using point clouds. In recent years, more and more researches have begun to use bio-heuristic algorithms for path planning of UAVs. This kind of planning method does not need to build a complex environment model but provides a very effective way for path optimization. Researchers have explored the path planning of UAVs in 2D and 3D space. Xu et al. proposed the application of chaotic artificial bee colony (ABC) methods to path planning [32], Duan et al. proposed the application of the hybrid ACO approach to path planning in 3D space [33]. Duan et al. [34] proposed the application of max-min adaptive ACO to the path planning of multiple UAVs. Mittal et al. [35] proposed an offline path planning method based on a multi-objective evolutionary algorithm. Roberge et al. [36] compared the performance of UAV path planning implemented by PSO and GA.

Our algorithm works similarly to PSO and genetic algorithms, so we briefly introduced how they work. The PSO algorithm is group-based and moves individuals in the group to a suitable area based on fitness to the environment. Genetic algorithm is a search algorithm used in computational mathematics to solve optimization. It is a kind of evolutionary algorithm. The intelligent algorithm uses all the path point sets as the optimization target and obtains the best path with the fitness value through the respective search methods.

1.2. Organization and Contributions

The structure of this paper is as follows. The problem description is introduced in Section 2, including the definition of path planning and the selection of obstacles. In Section 3, we introduce the methodology, including the path representation, the definition of the cost function and the description of the obstacle avoidance beetle antennae search (OABAS) algorithm. In Section 4, we show the simulation results and compare the performance of this algorithm with other evolutionary algorithms in path planning. In Section 5, we summarize the full paper. Before the end of this section, the main contributions of this article are listed as follows.

- This paper solves the shortest path and obstacle avoidance problem by proposing a novel path planning algorithm, termed OABAS algorithm.
- A linear loss function is designed as a cost function of the OABAS algorithm according to the constraints of UAVs.
- The proposed algorithm is applied to the UAV model and compared with the existing algorithms to verify the feasibility and efficiency.

2. Preliminaries

Path planning has a long history of research in robotics. Before the advent of the electric-driven robots, many classic path planning problems, such as the traveling salesman problem (TSP) [37] and the Chinese postman problem [38], have been a focus of academic research. In response to these classic problems, various solutions have been proposed. Since then, along with the emergence of various types of robotic arms [39,40], wheeled robots [41], and humanoid robots [42,43], there is a requirement to study the path planning problems of such devices.

2.1. Path Definition and Generation

In this section, we need to present two keywords that are used in robot navigation.

> Navigation: the task of navigation is to move a robot from one location to another. In the process of navigation, the robot movement needs a path that is gradually determined. During the navigation, it is often necessary to rely on the sensor data to update the information about position of the surrounding obstacles and to update the position and direction information of the robot.
>
> Path planning: in general, the work that path planning needs to do is to collect relevant data information and generate available paths based on constraints. We use the cost function to constrain the path. The cost function can generally be determined by the length of the path, possibility of collides with obstacles, the load of the robot, and other conditions that cause danger. From this, it can be inferred that reducing the numerical value of the cost function can plan an efficient, feasible and safe path.

According to the working environment with mobile robots, we can divide path planning into two types. One type is the global path planning of the model in a state where the environmental conditions are all known, and can also be called offline path planning. The other is local path planning, also known as online path planning. Local path planning uses the concept of navigation, mentioned above, to update the information about surrounding environment based on the sensor data and perform path planning in real-time to reach a specified destination.

We combine the concept of minimum threat surface (MTS) to ignore the optimization of the flight height value which means that the altitude value of the aircraft is determined by the environment and threat information and the tasks performed by the aircraft. This operation reduces the range of optimization required, and the generation time of the path is significantly reduced.

We specify the starting point and the target point of the UAV and the path planning is performed according to the specified points. This method is described in detail in Section 3, which generates

whole path at a time, then uses the proposed OABAS algorithm to adjust the points of the path until these points meet the planning requirements of the path.

2.2. Obstacle Sensing and Avoidance in Path Planning

In the global path planning, we can obtain some static target information in the environment, such as mountains and air defense threats. In order to not lose the generality, we used many different types of obstacle environments. We used obstacle-free, a single regular obstacle, multiple regular obstacles, a single irregular obstacle, multiple irregular obstacles and mixed obstacles (including multiple regular and irregular obstacles) for simulations. Such an obstacle environment setting can effectively prevent the algorithm from being useful only in a specific obstacle environment. Since the algorithm can reflect whether the obstacle is successfully avoided (there is a corresponding penalty value to record if it crosses the obstacle successfully), we can count the success rate and compare it with other path planning algorithms. In Section 4.1, we used a bit representation method to represent the presence of a virtual obstacle. In Section 4.4, we used a static cylindrical range to represent the air defense threat in space. In dynamic obstacles, such as abnormal weather areas and flocks, because of their shape uncertainty, we create a minimum enveloping cylinder to represent their range of influence. The moving direction of these dynamic obstacles is to simulate the movement of birds and unusual weather in nature to move.

3. Methodology

Now, we will present a method for path planning in a known environment as shown in Figure 1. First, we read the map information and generated a random initial path, then used this path as input to the OABAS algorithm which adjusted the waypoints until the path met the convergence condition. Finally, we output the qualified path to the UAV for execution. The core of this method is the formulation of the cost function and the optimization of the fitness value by OABAS algorithm. The following content is divided into three parts to introduce this method. Section 3.1 introduces the path generation and the acquisition of environmental information, Section 3.2 explains the structure and meaning of the cost function in detail, and Section 3.3 introduces the proposed OABAS algorithm.

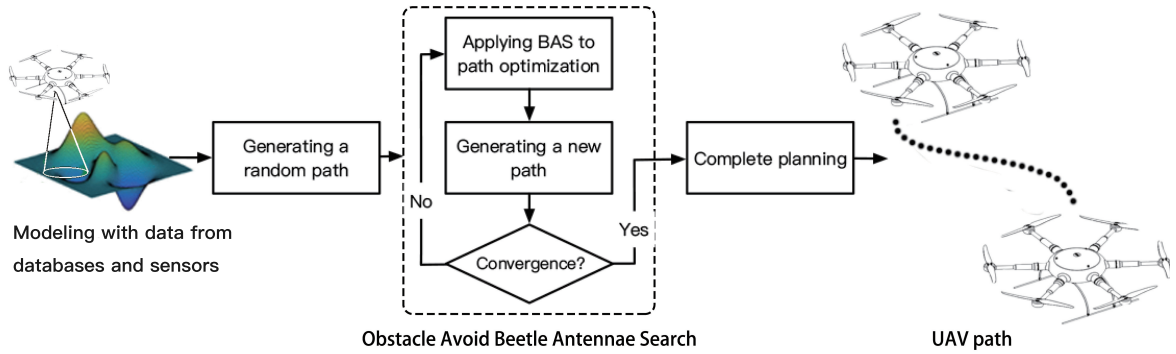


Figure 1. Flowchart of obstacle avoidance beetle antennae search (OABAS) algorithm applied to path planning of unmanned aerial vehicles (UAVs).

3.1. Path Representation

Search techniques (e.g., simulated annealing [44], A* [45]) are generally used to find the optimal solution for a particular function, but large-area path planning in a high-dimensional space is a typical large-scale optimization problem. With the expansion of the search space, the computational cost of searching for the best path in 3D space exponentially increases, therefore this problem is usually called a non-deterministic polynomial-time (NP)-hard problem. When the drone is performing a mission, we can assume that the aircraft maintains a minimum safety clearance with the land plane. The literature [46,47] proposes the concept of the MTS during the movement of the drone, indicating

that flying on this surface will have the property that the terrain threat is minimal and the path of the drone is on this surface. This assumption transforms the original 3D path planning problem to a 2D path planning on the MTS. Therefore, in the path planning and path recording of the drone, it is not necessary to record the altitude information of the aircraft. In other words, the altitude information of the aircraft does not need to be involved and recorded in the coding process of the path planning.

$$F(x,y) = h_u + f(x,y) + Q(x,y). \tag{1}$$

Equation (1) enables the UAV to have a minimal threat surface so that it can avoid obstacles in the vertical direction. Note that $F(x,y)$ is the height of the aircraft, $f(x,y)$ is the terrain profile, and h_u is the specified terrain clearance. The distance h_u between the ground and the aircraft can be a special function of the downrange. Also, the obstacles can be defined as $Q(x,y)$. As expressed in Equation (1), the minimum threat surface generated by MTS takes into account terrain, threats, and obstacles. If the ground suddenly rises, the $F(x,y)$ value at this position becomes very large. So according to the concept that our aircraft only fly on the smallest threat surface, we are not inclined to cross large obstacles. From the concept of the MTS mentioned above, the path we need to plan must be on this surface. Since we fixed the starting point and target points of the flight path and connect the path points of each step, the resulting path is also a curve. By projecting the curve onto the MTS according to the corresponding relationship of the projection, the track corresponding to the optimal path on the MTS can be easily obtained. According to the references [48–50], we define the drone path planning here. In this paper, we assume that the UAV's flight path is projected onto a 2D plane, making this space as workspace $\mathbb{U} \in \mathbb{R}^{k\times k}$ (k is the size of map). There are often obstacles in the workspace, so we need to define them. We defined obstacles as $\mathbb{O} \subset \mathbb{U}$, and the space of $\mathbb{U}$ that is free of obstacles is represented as $\mathbb{U}_{\text{fre}} = \mathbb{U} \setminus \mathbb{O}$. The starting point is $x_{\text{sta}} \in \mathbb{U}_{\text{fre}}$, and the target point is $x_{\text{end}} \in \mathbb{U}_{\text{fre}}$. We define the path [51] as a vector $\vec{\gamma} \in \mathbb{R}^s$ (s is the number of path points) with the following definitions.

Definition 1. *Path plan: initialize a path $\vec{\gamma}$, where $\gamma_1 = x_{sta}$ and $\gamma_s = x_{end}$. If there is a process $\phi(\cdot) : \mathbb{U} \to \mathbb{R}^s$ that satisfies $\vec{\gamma}(\tau) \in \mathbb{U}_{fre}$, for all $\tau \in 1, 2, \ldots, s$, then ϕ is called path planning.*

Definition 2. *Path optimization: We give an initialization path $\vec{\gamma}$. Under the constraint of cost function Equation (2), if we satisfy the Definition 1, we find a path $\vec{\gamma}'$, and $f(\boldsymbol{\gamma}') = \min\{f(\vec{\gamma}), \vec{\gamma} \in \Gamma = \{\gamma^1, \gamma^2, \ldots, \gamma^m\}\}$, m is the number of iterations of path planning, then $\vec{\gamma}'$ is the optimized path, and ϕ' is the path planning optimization.*

3.2. Cost Function

Since the relationship between the UAVs being detected, destroyed, and the state of the aircraft has not been explicitly studied so far, there is no accepted mathematical expression for the UAV's path cost function. From the cost function of cruise missiles [52–55], we make corresponding modifications, and apply the previous research results to the trajectory planning cost function of the aircraft, and then summarize the following constraints. First, the trajectory of the aircraft should as short as possible. This constraint reduces flight time, enhances safety and improves energy efficiency. Additionally, the UAVs usually have a maximum turning angle [4]. The maximum turning angle imposes more performance constraint on the aircraft. Last, the aircraft path cannot be too close to obstacles or known threats, because the close spacing between the aircraft and the obstacles can easily lead to accidental collisions. In accordance with these limitations, we propose an improved aircraft path planning cost function

$$f(\vec{\gamma}) = \sum_{i=2}^{s} (k_1 Li + k_2 H_i + k_3 T_i). \tag{2}$$

In Equation (2), k_1 represents the penalty coefficient of the distance, k_2 represents the penalty coefficient of the maximum corner, and k_3 represents the penalty coefficient of the obstacle collision.

$$L_i = \sqrt{x^2 + (\gamma_i - \gamma_{i-1})^2}. \tag{3}$$

Equation (3) denotes the contents of L_i which represents the distance cost of the ith path point, where x represents the displacement of the aircraft moving forward each time step (they are equal between themselves). The values γ_i and γ_{i-1} represent the projected length of the distance between the two path points of the aircraft in the y-axis direction

$$H_i = \begin{cases} 1, & \text{if } x \geqslant L_i \cos\theta, \\ 0, & \text{otherwise}, \end{cases} \tag{4}$$

where H_i is a corner penalty factor. When a certain flight path forms an angle larger than the specified maximum angle θ of the aircraft, this value is activated (set to be unity). We assume that the direction at which the aircraft starts is parallel to the positive x-axis direction. This imposes a penalty on path cost if the angle become larger than θ.

$$T_i = \begin{cases} 1, & \text{if } z_i \in \mathbb{O}, \\ 0, & \text{otherwise}, \end{cases} \tag{5}$$

where T_i is an obstacle penalty parameter. This value is activated (set to be unity) when there are obstacles near the path to keep away from the obstacles. The parameter z_i is the detection range. Due to the concept of MTS mentioned above, the flight altitude is guaranteed to be optimal, and other threat indices depend directly on the constraints. Also, all node coordinates of the path $\vec{\gamma}$ restrict each other. Therefore, the path planned by the proposed method not only enables the aircraft to reach the destination in a short time under the condition of safely avoiding the obstacle, but also meet the performance index constraint of the aircraft.

3.3. OABAS Algorithm

Recently, a new metaheuristic strategy called beetle antenna search (BAS) for function optimization was proposed in [56]. After setting the appropriate parameters, this strategy can quickly find the fitness value close to optimal of the objective function. Compared with the bio-heuristic algorithms mentioned in Section 1.1, it has tremendous search speed. We believe that the fast search performance of the BAS strategy fits well with the real-time requirement of the path planning. We improved the path planning algorithm on the basis of BAS and name it OABAS algorithm, and the description of OABAS algorithm is as follows. Figure 2 shows the flow chart of our proposed OABAS algorithm.

First, we get the initial path $\vec{\gamma}$ and calculate the objective fitness value $f(\vec{\gamma})$ to be optimized according to Equation (6).

$$f(\vec{\gamma}) = \sum_{i=2}^{s} (k_1 \sqrt{x^2 + (\gamma_i - \gamma_{i-1})^2} + k_2 H_i + k_3 T_i). \tag{6}$$

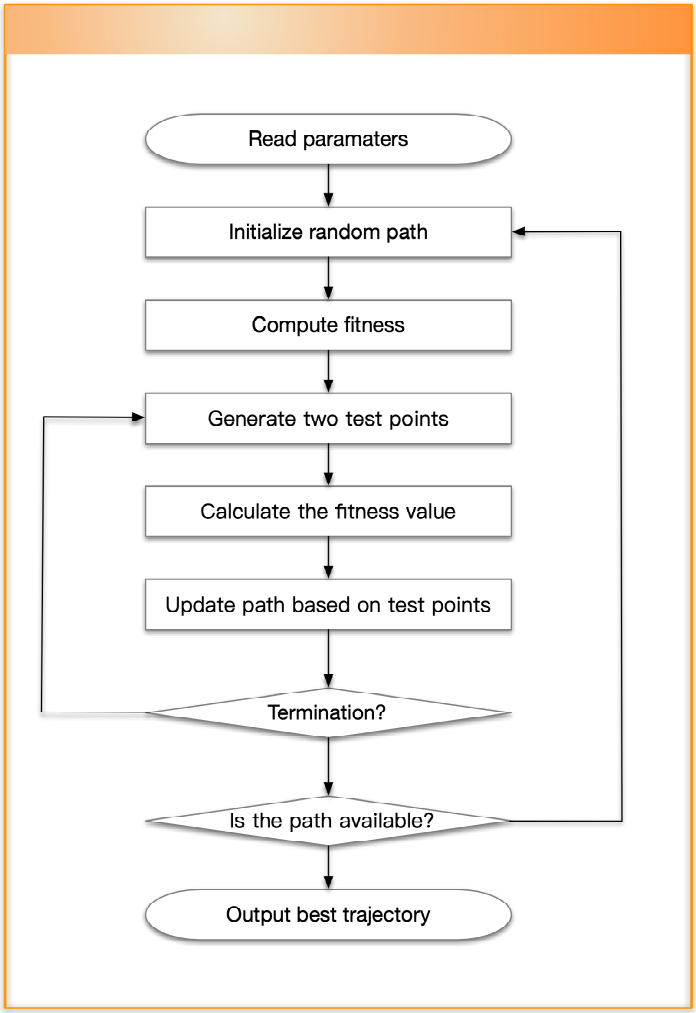


Figure 2. Flowchart of the proposed OABAS algorithm.

The more the path points s are used, the higher the accuracy of the planned path, but the path planning will take a long time. Randomly generate the beetle's search direction $\vec{d}$ and normalize it. In each dimension, the beetle has a corresponding exploration direction. The random direction is

$$\vec{d} = \frac{\text{rands}\,(s,1)}{\|\text{rands}\,(s,1)\|_2}, \tag{7}$$

where rands$(\cdot)$ is a function that generates s-dimensional random numbers, through which we obtain the normalized direction vector. Then we compare the left and right antenna's value of the beetle by the following equations

$$\begin{cases} \vec{\gamma}_{\text{rig}}^{m} = \vec{\gamma}^{m-1} + \delta^{m}\vec{d}, \\ \vec{\gamma}_{\text{lef}}^{m} = \vec{\gamma}^{m-1} - \delta^{m}\vec{d}, \end{cases} \tag{8}$$

where

$$\delta^{m} = \zeta^{m} c, \tag{9}$$

where δ^{m} is the length of the beetle's step size after m iterations, and setting this initial value appropriately can make the algorithm have better search performance.

$$\zeta^{m} = \zeta^{m-1}\eta, \tag{10}$$

where ζ^m is the antennae length after m iterations, which is a value that changes as the number of iterations increases. The change of δ is determined by the decay rate η (typically between 0 and 1). Note that c is the ratio of the length of the step to the length of the beetle's antennae. The beetle's next exploration location is updated by

$$\vec{\gamma}^m = \vec{\gamma}^{m-1} + \text{sign}\left(f\left(\vec{\gamma}_{\text{rig}}^{m-1}\right) - f\left(\vec{\gamma}_{\text{lef}}^{m-1}\right)\right)\delta^{m-1}\vec{d}, \tag{11}$$

where $f\left(\vec{\gamma}_{\text{rig}}^{m-1}\right)$ and $f\left(\vec{\gamma}_{\text{lef}}^{m-1}\right)$ indicate the fitness values of the beetle's left and right antennae after $m-1$ iterations, and function $\text{sign}(\cdot)$ is

$$\text{sign}(x) = \begin{cases} 1, & \text{if } x > 0, \\ 0, & \text{if } x = 0, \\ -1, & \text{otherwise.} \end{cases} \tag{12}$$

The algorithm is based on BAS, and its prototype algorithm is improved to be used for UAV path planning.

In order to explain the OABAS algorithm more clearly, we have detailed the steps of the OABAS algorithm in Algorithm 1. The explanation of variables and steps is as follows: δ^0 represents the initial exploration step size, and ζ^0 represents the initial length of antennae. The optimal fitness value of the cost function is initialized to infinity, and γ^0 represents initial path, which is continuously optimized by OABAS algorithm. When $\vec{\gamma}$ satisfies the convergence condition, the optimized solution ϕ is output and the optimal fitness value f_{bes} is saved. The remainder of variables are same as as described above.

Algorithm 1: Obstacle avoidance beetle antennae search (OABAS) algorithm for path planning of UAVs.

Input: Path point x_{sta} and x_{end}, domain $\mathbb{U}$ and $\mathbb{O}$, and the number of path points s;
Output: Planning path ϕ;

```
Initialize c, m, k1, k2, k3, δ^0, ζ^0, γ^0, η, f_bes, and γ⃗_bes;
for p = 1 to m do
    Generate d⃗ according to Equation (7);
    Calculate γ⃗_r and γ⃗_l according to Equation (8);
    Update γ⃗^m according to Equation (11);
    Save γ⃗^m and fitness f(γ^m) after mth iteration;
    if f_bes ⩾ f(γ⃗^m) then
        f_bes = f(γ⃗^m) ;
        γ⃗_bes = γ⃗^m ;
    Update δ^m and ζ^m according to Equations (9) and (10);
φ = γ⃗_bes;
```

4. Simulation Studies and Experimental Results

In this section, we apply the proposed OABAS algorithm to a simulated model of a UAV to test their real-world performance in path planning. In simulations, we use a volumeless and massless particle to simulate UAVs to reduce calculation time. Considering the actual volume of UAVs, we set a minimum safety gap in the algorithm to ensure that UAVs can complete the task of obstacle avoidance. The maps used in the simulation are equal in length and width, and the map information includes only two types of the domain: the feasible domain and the infeasible domain. We use black areas to represent obstacles, and the white area represent free space. In Section 4.1, we apply OABAS algorithm in different types of environments to verify that OABAS algorithm is effective in real-world scenario. In Section 4.2, we set different parameters for the algorithm to obtain the optimal algorithm

performance, and also show the statistical effect diagram of the path planning. In Section 4.3, we apply the proposed OABAS algorithm, the conventional ABC and PSO algorithm in same environments and conduct comparative simulations. In Section 4.4, we implemented dynamic obstacles of different sizes and shapes in high-resolution maps by pre-planning future paths, and we assume that these obstacles are only detected when approaching the aircraft.

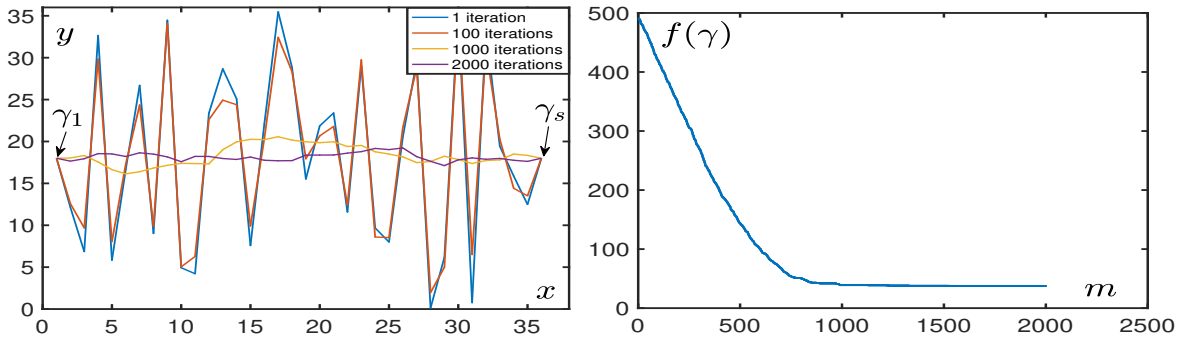
4.1. UAV Path Planning in Different Environments

In this subsection, we apply the OABAS algorithm for obstacle avoidance. Due to the complex environments in the real-world scenarios, whether obstacle avoidance is effective in various types of environments needs to be verified. We set up six types of environments for simulations: obstacle-free, a regular obstacle, multiple regular obstacles, an irregular obstacle, multiple irregular-mixed obstacles.

4.1.1. In Obstacle-Free Environment

This subsection is used to verify that the convergence of the OABAS algorithm are same as in path planning. In this part, we used a map with the length of $x = 36$ and the width of $y = 36$. The starting point γ_1 and the target point γ_s were set to the leftmost and rightmost centers of the map. The path was a straight line between two points, the theoretical optimal value was $f(\vec{\gamma_{\text{bes}}}) = 35$. After acquiring the map information, we generated a random path according to the algorithm described in Figure 1.

According to the curve in Figure 3a, as the number of iterations increased, the generated path had a trend of convergence to the optimal straight line path. We optimized this fitness value to make the UAVs follow the best path under constraint conditions. Figure 3b shows the change in fitness of the planned path.



(**a**) Comparison of the paths generated on the same map and different iterations

(**b**) Trend of optimal fitness value

Figure 3. The performance in an environment without obstacles.

From the simulation, we find that the OABAS algorithm only takes about 1000 iterations to plan a better path. Based on the results of Table 1, convergence takes time around 0.01 s. Table 1 is a statistical result for UAV path planning in an environment without obstacles. We set the step size δ to be a variable, and 100 simulations are performed using the method of controlling variables. Under different step size δ, we set the decay rate $\eta = 0.99995$. Then we count the average convergence steps m_{ave}, the average convergence time t, average fitness value f_{ave} after 100 convergences. Table 1 shows the best fitness value f_{bes} obtained in 100 tests, the standard deviation f_{Std} of the batch test, and finally we analyze the result and determine if the step is convergence to a reasonable value. The convergence criterion we use here is $f_{\text{ave}} \leqslant 50$, and if the condition is true, we conclude it to be a reasonable step size.

Table 1. Relationship between step size and OABAS algorithm performance in an environment without obstacles.

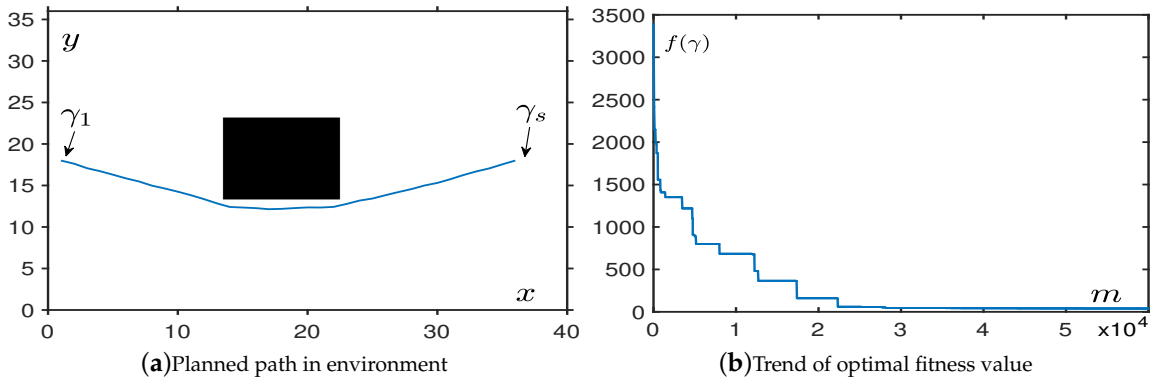
δ	$m_{\mathbf{ave}}$	T (s)	$f_{\mathbf{ave}}$	$f_{\mathbf{bes}}$	$f_{\mathbf{Std}}$	**Con.**
0.5	10,503	0.029	35.51	35.30	0.010	Yes
1.0	**20418**	**0.057**	**35.47**	**35.29**	**0.008**	**Yes**
1.5	23,873	0.067	35.60	35.40	0.010	Yes
2.0	24,077	0.067	35.92	35.54	0.035	Yes
2.5	24,122	0.067	36.40	35.85	0.077	Yes
3.0	23,957	0.067	36.97	36.03	0.105	Yes
3.5	24,128	0.067	37.53	36.31	0.265	Yes
4.0	24,031	0.067	38.24	36.98	0.366	Yes
4.5	24,243	0.068	39.11	37.70	0.816	Yes
5.0	24,103	0.067	40.05	37.50	1.131	Yes
5.5	24,085	0.067	40.76	38.48	1.429	Yes
6.0	24,186	0.067	41.59	39.56	1.145	Yes
6.5	24,115	0.067	42.60	39.36	3.033	Yes
7.0	24,039	0.067	43.74	41.14	1.910	Yes
7.5	24,105	0.067	44.82	41.76	3.693	Yes
8.0	24,141	0.067	46.79	42.03	7.127	Yes
8.5	24,269	0.068	47.21	43.74	4.717	Yes
9.0	24,195	0.067	48.56	44.02	6.020	Yes
9.5	24,155	0.067	49.69	45.13	7.214	Yes
10	24,055	0.067	50.98	45.00	8.315	No

4.1.2. In Regular Obstacles Environment

In this subsection, we test the OABAS algorithm for environments with a single regular obstacle and multiple regular obstacles.

As shown in Figure 4a, a black square in the center area is used to represent an obstacle. In order to the visually observe the effect of obstacle avoidance, we set the connection points between the starting point and the target, to pass point through the obstacle area. We can clearly understand that the path we finally planned is successfully avoiding obstacles, and there is a certain safety distance between the path and the obstacle to avoid collisions.

Figure 4b shows the number of iterations required for the algorithm to converge. It was more than the obstacle-free environment in Section 4.1.1, which was roughly around 2.2×10^4 iterations. Later, we performed a more detailed performance test at Table 2 to show the relationship between step size δ and OABAS algorithm performance in the environment with a regular obstacle. The specific indicators were the same as those used in Section 4.1.1. In Table 2, we can see that the convergence in the environment with a single obstacle was very fast, but a too small initial step size will cause the algorithm to fall into the optimum local value prematurely. The convergence criterion we used here was $f_{\text{ave}} \leqslant 100$, and if the condition was true, we concluded it to be a reasonable step size. The reason for algorithm falling into a local optimal value was that the f_{ave} value was large, proving that most of the paths were not successfully planned.



(**a**)Planned path in environment (**b**)Trend of optimal fitness value

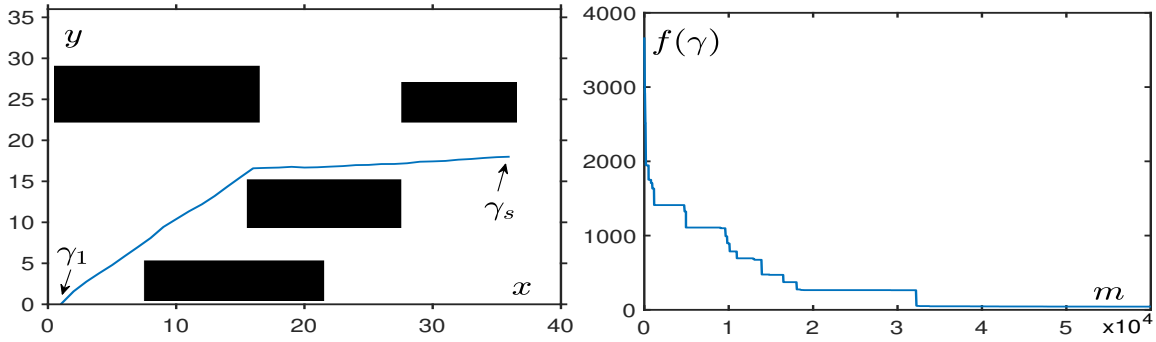
Figure 4. The performance in an environment with a regular obstacle.

Table 2. Relationship between step size and OABAS algorithm performance in the environment with a regular obstacle.

δ	$m_{\mathbf{ave}}$	t (s)	$f_{\mathbf{ave}}$	$f_{\mathbf{bes}}$	f_{Std}	**Con.**
0.5	6819	0.019	858.32	557.84	1.30×10^4	No
1.0	4532	0.013	620.79	38.25	6.64×10^4	No
1.5	7898	0.022	228.38	37.28	5.69×10^4	No
2.0	12,284	0.035	104.18	37.33	1.94×10^4	No
2.5	17,209	0.049	70.35	37.83	8.45×10^3	Yes
3.0	19,065	0.054	55.54	37.71	3.21×10^3	Yes
3.5	19,984	0.057	58.37	38.61	3.09×10^3	Yes
4.0	21,096	0.060	53.17	39.17	3.45×10^3	Yes
4.5	**22,111**	**0.063**	**49.58**	**39.73**	$\mathbf{9.73 \times 10^3}$	**Yes**
5.0	22,851	0.065	53.79	39.80	3.05×10^3	Yes
5.5	22,893	0.065	53.80	41.49	1.48×10^3	Yes
6.0	23,313	0.066	51.11	41.58	1.62×10^3	Yes
6.5	23,143	0.066	68.48	42.31	4.73×10^3	Yes
7.0	23,556	0.067	68.72	42.66	4.44×10^3	Yes
7.5	23,436	0.066	59.53	43.64	1.87×10^3	Yes
8.0	23,476	0.067	75.87	44.90	3.98×10^3	Yes
8.5	23,664	0.067	86.04	45.65	6.28×10^3	Yes
9.0	23,605	0.067	114.23	47.07	2.35×10^4	No
9.5	23,595	0.067	132.36	50.07	1.28×10^4	No
10	23,508	0.067	153.99	51.96	1.10×10^4	No

As shown in Figure 5a, we had set up a number of rectangular obstacles of different specifications. In order to achieve the visual effect of obstacle avoidance, we set the connection points of the starting point and the target point, to pass through several obstacle areas.

From Figure 5a, we can clearly understand that the path we finally planned is successfully avoiding obstacles. Figure 5b shows that the algorithm converged at roughly the same speed as the performance in an environment with a regular obstacle. Later, we performed a more detailed performance test as shown in Table 3, and we calculated the effect of different step sizes on the performance of OABAS algorithm in the case of multiple regular obstacles. As shown in Table 3, the OABAS algorithm converged very quickly with a reasonable step size δ in the environment with multiple obstacles, but an excessive initial step size δ made it challenging to converge to a better value later. Therefore, the step size of the algorithm should be set within a reasonable range to avoid setting a huge initial step size for a wide range of searches. The convergence criterion we use here was $f_{\mathrm{ave}} \leqslant 100$, and if the condition was true, we concluded this it to be a reasonable step size.



(**a**) Planned path in the environment (**b**) Trend of optimal fitness value

Figure 5. The performance in the environment with multiple regular obstacles.

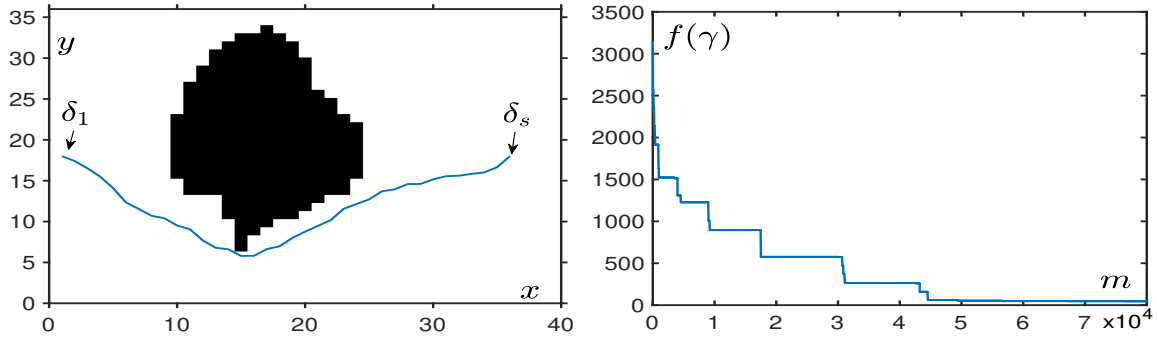
Table 3. Relationship between step size and OABAS algorithm performance in an environment with multiple rule obstacles.

δ	$m_{\mathbf{ave}}$	t (s)	$f_{\mathbf{ave}}$	$f_{\mathbf{bes}}$	$f_{\mathbf{Std}}$	**Con.**
0.5	4897	0.014	46.04	42.41	2.99×10^2	Yes
1.0	6292	0.018	46.31	42.78	4.02×10^2	Yes
1.5	9382	0.027	51.73	42.83	6.31×10^2	Yes
2.0	12,627	0.036	50.30	42.38	4.47×10^2	Yes
2.5	**15,906**	**0.045**	**48.06**	**42.84**	$\mathbf{2.23 \times 10^2}$	**Yes**
3.0	18,766	0.053	51.87	42.77	8.54×10^2	Yes
3.5	20,210	0.057	55.10	43.29	1.27×10^3	Yes
4.0	21,267	0.060	54.54	43.75	1.04×10^3	Yes
4.5	22,036	0.063	62.07	43.74	2.76×10^3	Yes
5.0	22,445	0.064	71.34	44.34	1.43×10^4	Yes
5.5	23,327	0.066	66.52	44.60	2.65×10^3	Yes
6.0	23,063	0.065	93.94	46.28	2.79×10^4	Yes
6.5	23,468	0.067	116.06	46.03	4.88×10^4	No
7.0	22,938	0.065	176.84	47.59	8.61×10^4	No
7.5	23,541	0.067	147.41	49.04	4.92×10^4	No
8.0	23,613	0.067	181.11	47.65	3.59×10^4	No
8.5	23,581	0.067	164.87	50.87	1.86×10^4	No
9.0	23,522	0.067	302.28	54.17	9.23×10^4	No
9.5	23,794	0.067	267.42	54.26	3.56×10^4	No
10	23,399	0.066	393.70	54.07	1.25×10^5	No

4.1.3. In Irregular Obstacle Environments

As we have verified, OABAS algorithm has good simulation results in the environment without obstacles and in the environment with regular obstacles. In this subsection we simulation with the environment with irregular obstacles.

In the first simulation, we add a large irregular obstacle to test more comprehensive performance. We use black areas to represent obstacles that are continuous in the center of the map. In order to visually observe the effect of obstacle avoidance, we set the starting point γ_1 and the target point γ_s such that the straight line between them passes through the obstacles. It can be seen from Figure 6, OABAS algorithm can accomplish the requirements of the short path, avoiding obstacles and maintaining safe separation from obstacles. Figure 6 shows that when we plan the path in an environment with an irregular obstacle, the number of iterations for convergence is about twice that of the previous simulations. In this environment, we find from Table 4 that it is different from the previous environment. At the same time, the failure rate in this environment has also increased significantly, so the convergence criteria is appropriately adjusted to $f_{\text{ave}} \leqslant 150$.



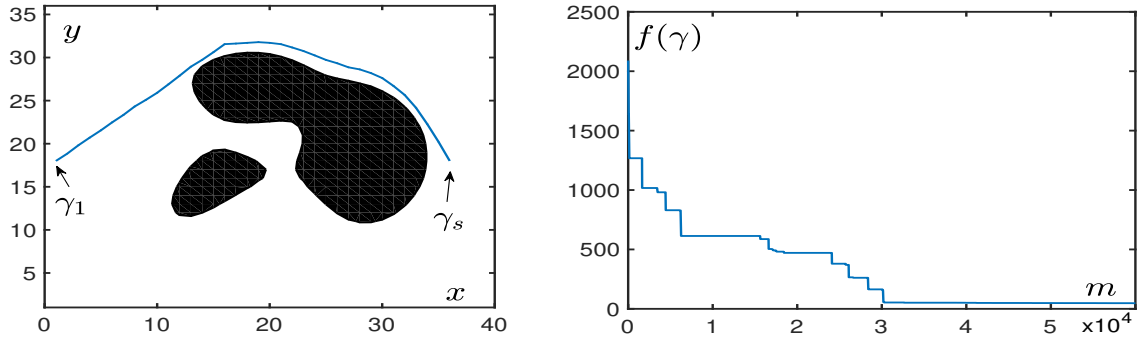
(**a**) Planned path in environment (**b**) Trend of optimal fitness value

Figure 6. The performance in an environment with an irregular obstacle.

Table 4. Relationship between step size and OABAS algorithm performance in an environment with a single irregular obstacle.

δ	m_{ave}	t (s)	f_{ave}	f_{bes}	f_{Std}	Con.
0.5	2076	0.006	839.27	440.39	8.31×10^3	No
1.0	4833	0.014	578.47	45.59	8.40×10^4	No
1.5	9304	0.026	235.70	45.79	8.58×10^4	No
2.0	15,680	0.044	185.26	45.01	6.73×10^4	No
2.5	**21454**	**0.061**	**117.10**	**45.91**	$\mathbf{3.33 \times 10^4}$	**Yes**
3.0	27,507	0.078	127.29	45.32	2.67×10^4	Yes
3.5	31,483	0.089	136.02	45.31	3.38×10^4	Yes
4.0	35,306	0.100	137.72	45.12	2.75×10^4	Yes
4.5	37,304	0.106	187.19	44.38	4.92×10^4	Yes
5.0	44,019	0.125	136.82	44.42	2.51×10^4	Yes
5.5	47,124	0.134	130.63	45.73	2.52×10^4	Yes
6.0	48,738	0.138	159.98	44.61	3.49×10^4	No
6.5	50,540	0.143	145.77	45.68	3.29×10^4	Yes
7.0	53,528	0.152	134.59	45.33	2.69×10^4	Yes
7.5	55,753	0.158	157.71	45.14	3.58×10^4	No
8.0	58,812	0.167	129.98	45.34	2.51×10^4	Yes
8.5	59,677	0.169	185.15	45.04	4.78×10^4	No
9.0	62,318	0.177	178.01	45.54	4.70×10^4	No
9.5	64,671	0.183	145.06	46.25	3.24×10^4	Yes
10	66,818	0.190	153.18	45.71	3.79×10^4	No

In the second simulation, we generated multiple irregular obstacles located in the middle and back of the map. It can be seen from Figure 7a that the algorithm can complete the task of obstacle avoidance with an environment with multiple irregular obstacles. Figure 7b shows that the time spent was roughly the same as spent in the environment with a single irregular obstacle. In this simulation, we find from Table 5 that irregular obstacles needed more iterations to converge, but had a higher success rate. We still set the convergence criterion in this environment to be $f_{\text{ave}} \leqslant 100$.



(**a**) Planned path in the environment (**b**) Trend of optimal fitness value

Figure 7. The performance in an environment with multiple irregular obstacles.

Table 5. Relationship between step size and OABAS algorithm performance in an environment with multiple irregular obstacles.

δ	$m_{\mathbf{ave}}$	t (s)	$f_{\mathbf{ave}}$	$f_{\mathbf{bes}}$	$f_{\mathbf{Std}}$	**Con.**
0.5	17,100	0.049	1131.63	1011.86	5.84×10^2	No
1.0	14,902	0.042	1029.72	667.69	9.34×10^3	No
1.5	8053	0.023	700.90	44.30	7.98×10^4	No
2.0	12,208	0.035	324.22	44.21	7.84×10^4	No
2.5	18,918	0.054	112.02	43.19	1.64×10^4	No
3.0	23,355	0.066	107.40	43.46	9.18×10^3	No
3.5	29,487	0.084	92.42	44.40	6.57×10^3	Yes
4.0	**35,404**	**0.100**	**75.76**	**43.80**	$\mathbf{3.76 \times 10^3}$	**Yes**
4.5	38,980	0.111	77.82	43.39	5.02×10^3	Yes
5.0	41,242	0.117	93.62	43.02	6.82×10^3	Yes
5.5	44,172	0.125	83.76	43.41	5.12×10^3	Yes
6.0	48,920	0.139	86.69	43.28	4.53×10^3	Yes
6.5	50,991	0.145	83.37	44.16	4.41×10^3	Yes
7.0	52,113	0.148	82.61	44.49	3.44×10^3	Yes
7.5	57,042	0.162	86.57	44.02	6.95×10^3	Yes
8.0	55,957	0.159	96.58	44.46	7.09×10^3	Yes
8.5	58,915	0.167	79.12	44.29	3.62×10^3	Yes
9.0	61,916	0.176	84.94	42.79	5.25×10^3	Yes
9.5	62,468	0.177	101.00	44.37	7.51×10^3	No
10	66,231	0.188	70.62	44.01	3.26×10^3	Yes

4.1.4. In Mixed Obstacle Environments

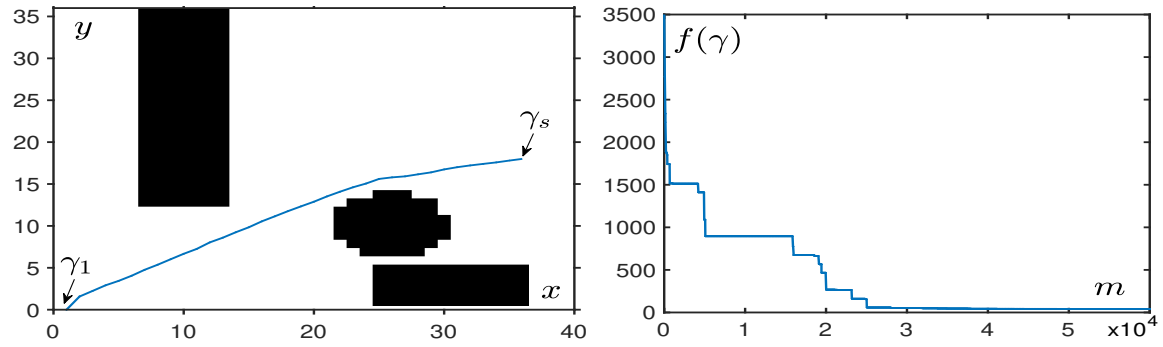
We placed both regular and irregular obstacles in the map to test the performance of OABAS algorithm. As with the previous environment, it was necessary to avoid multiple obstacles in order to reach the endpoint located in the midpoint on the right from the starting point in the lower left corner.

Figure 8a shows that OABAS successfully avoids obstacles and path length was short with short part length. Figure 8b shows that in the convergence speed of the algorithm was roughly the same as the environment with regular obstacles and obstacle-free.

Finally, we performed a more detailed performance test as shown in Table 6, which quantifies the performance impact of different step sizes on OABAS algorithm in the case of mixed obstacles. The OABAS algorithm converged very quickly in the environment with mixed obstacles, and the step size δ should be set within a reasonable range for a wide range of searches.

From the above simulations, we can observe that the performance of the OABAS algorithm changes in different types of environments. These simulations demonstrate that OABAS algorithm performs well in different types of environments. Although there are some differences in the

performance of the OABAS algorithm in different types of environments, most of the planned paths have a high success rate and can converge quickly (visible in Section 4.1.3).



(**a**) Planned path in the environment (**b**) Trend of optimal fitness value

Figure 8. The performance in an environment with mixed types of obstacles.

Table 6. Relationship between step size and OABAS algorithm performance in an environment with mixed types of obstacles.

δ	m_{ave}	t (s)	f_{ave}	f_{bes}	f_{Std}	Con.
0.5	12,488	0.035	40.53	39.97	0.11	Yes
1.0	17,521	0.050	40.22	39.91	0.08	Yes
1.5	21,086	0.060	40.40	39.99	0.22	Yes
2.0	21,694	0.062	40.95	40.10	2.12	Yes
2.5	22,186	0.063	44.53	40.42	5.73×10^2	Yes
3.0	22,336	0.063	47.37	40.49	8.52×10^2	Yes
3.5	**23,226**	**0.066**	**43.37**	**41.07**	$\mathbf{1.24 \times 10^2}$	**Yes**
4.0	23,170	0.066	49.39	41.67	8.93×10^2	Yes
4.5	23,319	0.066	49.12	42.22	1.26×10^3	Yes
5.0	23,189	0.066	58.41	42.75	2.70×10^3	Yes
5.5	23,420	0.066	73.59	43.10	6.83×10^3	Yes
6.0	23,131	0.066	67.96	42.91	3.91×10^3	Yes
6.5	23,575	0.067	86.27	42.75	3.71×10^4	Yes
7.0	23,796	0.067	100.55	44.47	2.71×10^4	No
7.5	23,672	0.067	133.62	45.15	4.51×10^4	No
8.0	23,706	0.067	135.61	46.58	2.00×10^4	No
8.5	23,769	0.067	158.27	47.54	4.26×10^4	No
9.0	23,803	0.068	205.10	50.41	5.27×10^4	No
9.5	23,882	0.068	248.31	50.15	1.30×10^5	No
10	23,799	0.068	395.42	51.71	1.39×10^5	No

4.2. Cycle Batch Tests

The content of this section explores the impact of different step size δ and step attenuation rate η on path planning success rates ψ. In order to find parameters that can generate as many successful paths as possible, we performed multiple tests with different δ and η. In this experiment, we set the number of iterations to 1000. A smaller number of iterations resulted in a more significant difference in success rates, making it easier to select the optimal set of step and attenuation rate parameter pairs. But this was not our actual planning success rate. The sample size of the simulation was 100. Success rate ψ was calculated after OABAS algorithm generated 100 paths in the simulation of multiple irregular obstacles. Finally, the optimal δ and η were obtained, and the targeted recurrence test was carried out. That is, the simulation performed several times with the same value of δ and η which generated a thousand paths and was used to represent the quality of paths visually.

The Table 7 is the result of the path generated by the combination of different parameters. The horizontal axis represents the initial δ of 1 to 9, and the vertical axis represents η of 0.99991 to

1 (a total of ten levels). The values in the table are the number of successful paths generated after 100 trials.

We visualize it as shown in Figure 9 to show its distribution. In this histogram, the effect of the two parameters δ and η on the convergence effect is more obvious. The better results are concentrated in, the more appropriate δ and η.

Table 7. Influence of step size and attenuation rate on the success rate of path planning.

η \ δ	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0.99991	0	35	67	76	72	66	64	63	57
0.99992	0	35	72	79	77	73	61	58	68
0.99993	2	43	73	83	78	81	61	62	70
0.99994	1	51	79	78	76	76	73	63	63
0.99995	1	49	82	**90**	87	78	74	69	64
0.99996	1	55	87	89	82	76	60	56	39
0.99997	1	66	**91**	86	78	43	4	0	0
0.99998	2	76	80	46	2	0	0	0	0
0.99999	7	76	7	0	0	0	0	0	0
1.00000	10	12	0	0	0	0	0	0	0

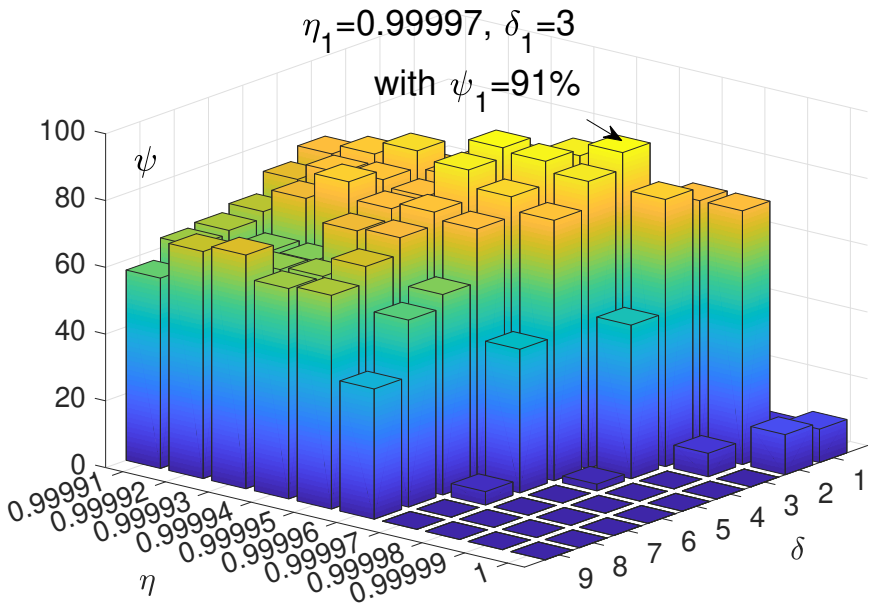
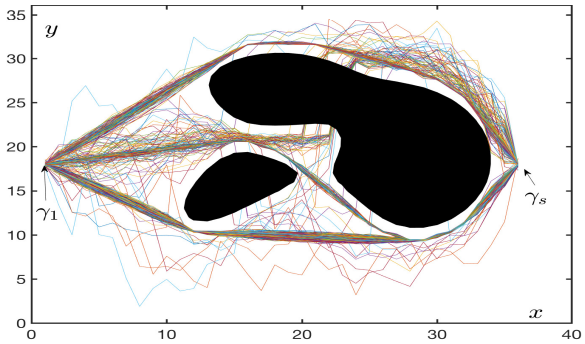
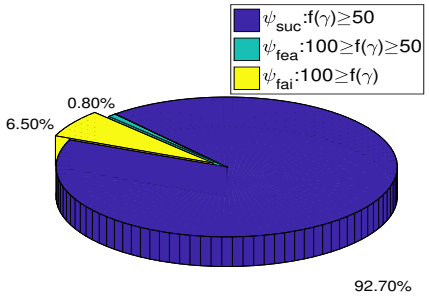


Figure 9. Effect of step and decay rate on success rate of OABAS algorithm.

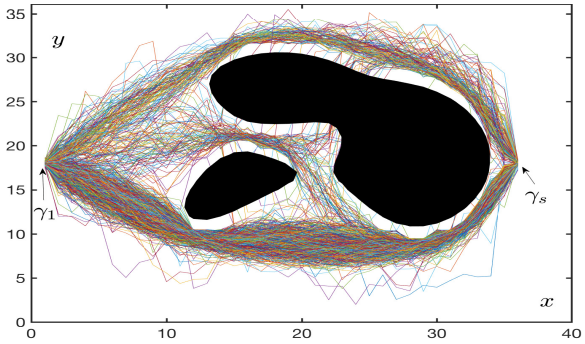
In Figure 10b,d, we used the blue part ψ_{suc} to represent the number of excellent paths generated. The green part ψ_{fea} is used to indicate the number of available paths generated which means this types of the path with a longer path length, but satisfying the maximum turning angle and avoiding the obstacles. Finally, we used the yellow part ψ_{fai} to indicate the number of failed paths generated. According to the above statistical simulation, when $\delta_1 = 4$ and $\eta_1 = 0.99995$, the path planning had a success rate $\psi_1 = 93.7\%$. In addition, when $\delta_2 = 3$ and $\eta_2 = 0.99997$, there was also a high success rate $\psi_2 = 92.7\%$. The convergence properties of the OABAS algorithm were similar to those of the BAS on the objective function, and both the appropriate δ and η were needed to achieve an optimal convergence.



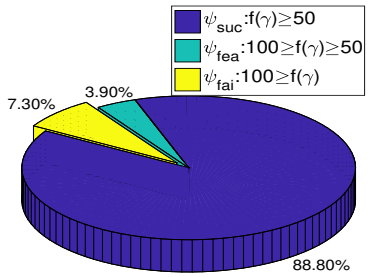
(a) Path distribution of 1000 plans with $\delta_1 = 4$ and $\eta_1 = 0.99995$.



(b) The distribution of the quality of the planned path with $\delta_1 = 4$ and $\eta_2 = 0.99995$.



(c) Path distribution of 1000 plans with $\delta_2 = 3$ and $\eta_2 = 0.99997$.



(d) The distribution of the quality of the planned path with $\delta_2 = 3$ and $\eta_2 = 0.99997$.

Figure 10. Path distribution in the case of fixed step size and decay rate.

4.3. Comparisons with PSO and ABC

In order to compare the performance between the OABAS algorithm and traditional bio-heuristic algorithms in the path planning, we need to make a more detailed evaluation. Unlike Section 4.2, after obtaining the best parameters, our OABAS uses an iteration step of 50000 to ensure the reliability of path planning.

We performed multiple simulations and took the average value to eliminate contingency and ensure that the results were indicative of real scenarios. We wanted to get the path length as short as possible. At the same time, the path planning success rate was also one of the evaluation criteria for evaluating whether a path planning algorithm has good quality or not. The path planning algorithm that does not have a reliable and stable result will lead to very serious consequences when the UAVs are flying in the real world. Therefore, the higher the success rate of path planning, the better. So we counted the following indicators of the algorithm in Tables 8 and 9, the average convergence time t of the algorithm, the average convergence fitness value f_{ave} of the algorithm and the success rate ψ of the algorithm.

On the map of size k, we set the number of path points $s_1 = k/2$ and $s_2 = k$. First, we set the path point to s_1. In this case, a show from the Table 8 that the OABAS algorithm leads the time indicator. However, after the visual observation of the paths, we found that the UAVs will collide with obstacles. After analysis, we believe that too few path point settings will lead to rough path planning so that UAV will collide with obstacles.

After checking the path, fewer nodes were found to cause the flight path to avoid obstacles, so we needed to set up more nodes for path planning. More nodes will lead to an increase in planning time. We set the number of path points to s_2 and tested again in the same situation. In the visualization results, the path was successfully planned, in this case did not collide with the obstacle, which proves that the number of path points we selected was reasonable. In Table 9, the OABAS algorithm was faster

than other algorithms. The OABAS algorithm achieved a very high success rate in each case while the PSO was challenging to converge due to too many path points. Under the premise of guaranteeing the success rate, the efficiency of the OABAS algorithm was much better, and the time complexity of the ABC algorithm was high.

Table 8. Performance comparisons of different algorithms in low dimensional situations.

Type of Obstacles	Algorithms	t (s)	f_{ave}	ψ (%)
Single regular	OABAS	**0.059**	17.63	**100**
	PSO	0.606	19.54	100
	ABC	0.356	17.29	100
Multiple regular	OABAS	**0.083**	17.37	97
	PSO	2.254	19.30	97
	ABC	0.279	17.32	100
Single irregular	OABAS	**0.102**	21.16	**100**
	PSO	7.670	22.68	87
	ABC	0.450	19.29	100
Multiple irregular	OABAS	**0.065**	19.69	**100**
	PSO	17.285	19.52	63
	ABC	0.571	19.90	100
Mixed	OABAS	**0.088**	**19.56**	**100**
	PSO	0.536	21.54	98
	ABC	0.325	21.64	100

Table 9. Performance comparisons of different algorithms in high dimensional situations.

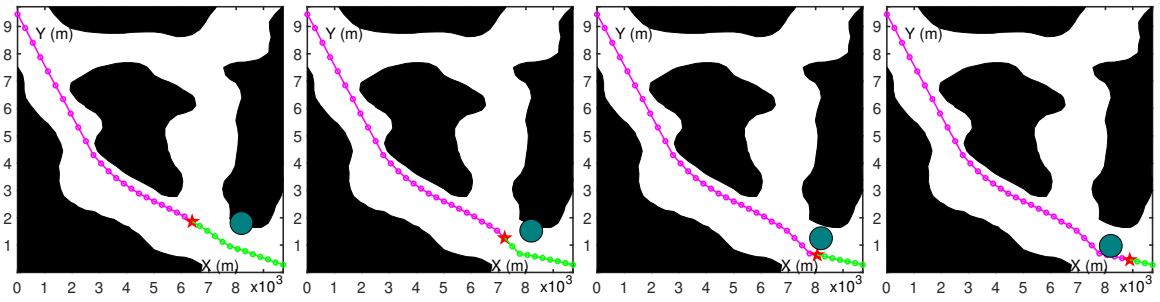
Type of Obstacles	Algorithms	t (s)	f_{ave}	ψ (%)
Single regular	OABAS	**0.301**	**36.94**	**100**
	PSO	NA	57.04	52
	ABC	49.729	40.47	96
Multiple regular	OABAS	**0.371**	**41.77**	99
	PSO	NA	51.18	38
	ABC	32.382	48.48	100
Single irregular	OABAS	**0.406**	46.01	99
	PSO	184.593	44.44	39
	ABC	35.434	40.91	100
Multiple irregular	OABAS	**0.428**	41.71	**95**
	PSO	251.506	40.88	12
	ABC	56.813	41.36	48
Mixed	OABAS	**0.345**	**39.91**	**100**
	PSO	NA	57.04	92
	ABC	46.956	48.93	100

4.4. Tested Results on High-Resolution Maps with Various Types of Obstacles

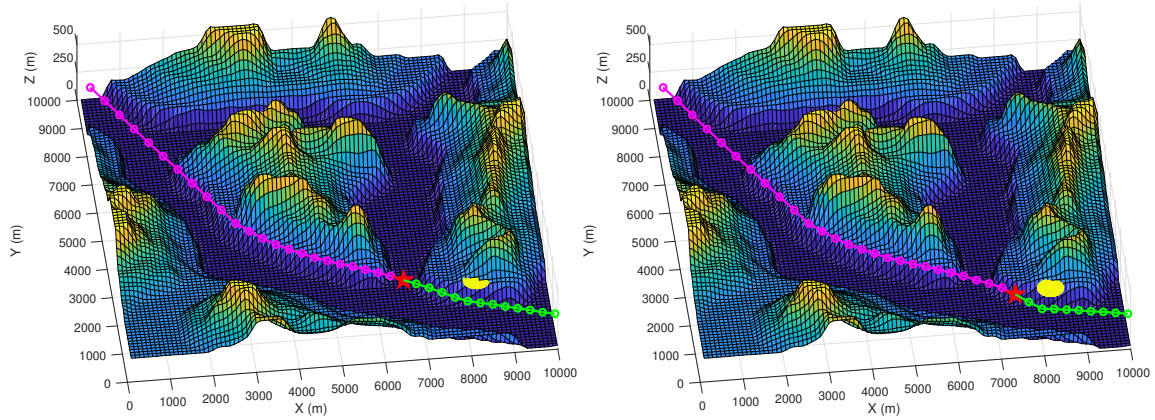
We used more high-resolution test maps from the GeoBase repository [57] and ran experiments in a dynamic environment. By pre-planning the unpassed path, we implement a simple dynamic path planning function. We assumed that there was a dynamic obstacle in the environment, and used the dark green circle in the picture to represents it. We set the UAV's sensing range 800 m in the real environment, and obstacles within 800 m of distance from the UAV were be discovered. In the experiment, the size of the UAV was generally small, so we did not consider its specific shape and model it according to its specific size. We took the larger of the fuselage length and the length of the wing as diameter to draw hollow circles to represent the UAV. For example, the Global Hawk has a length of 13.5 m and a wing length of 35 m, so we chose 35 meters as the model's parameters. In the

experiment, we used 50 m as the diameter to draw hollow circles for modeling in Figures 11 and 12, and used 100 m as the diameter of the UAV in Figure 13.

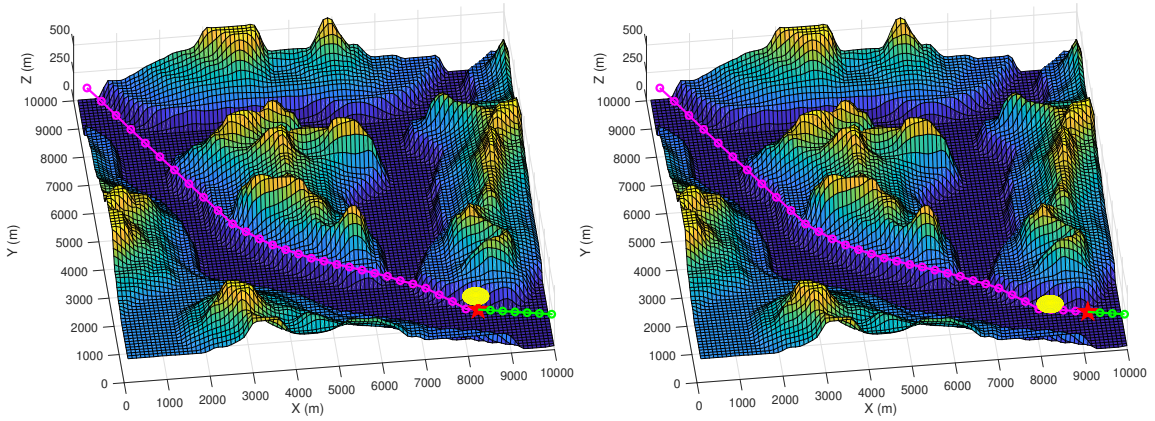
A red star indicates where the UAV is currently located. Green circles and line segments represent pre-planned paths. The dark green dot represents an obstacle in the 2D environment that exists in the minimum threat surface of the UAV and poses a threat to the path. In the iterative process, dynamic path planning can try to plan a better pre-planning path while avoiding obstacles.



(a) Step 24 of UAV trajectory in a 2D environment **(b)** Step 27 of UAV trajectory in a 2D environment **(c)** Step 30 of UAV trajectory in a 2D environment **(d)** Step 23 of UAV trajectory in a 2D environment

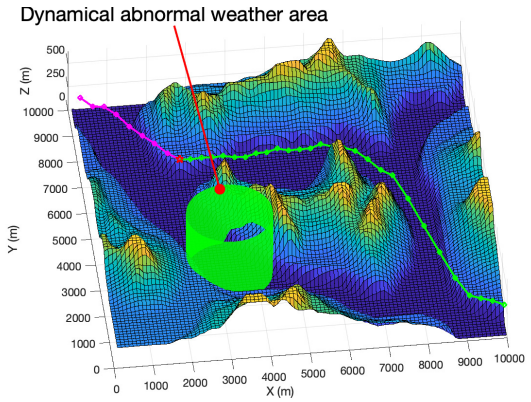


(e) Step 24 of UAV trajectory in a 3D environment **(f)** Step 27 of UAV trajectory in a 3D environment

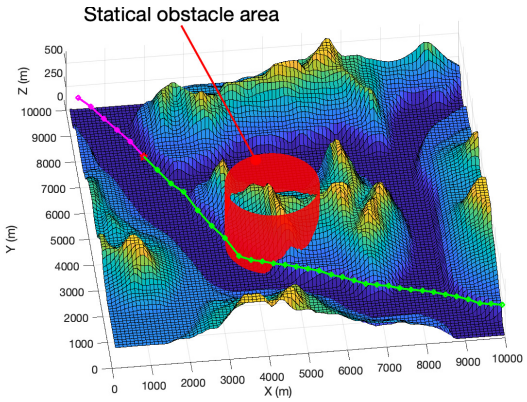


(g) Step 30 of UAV trajectory in a 3D environment **(h)** Step 33 of UAV trajectory in a 3D environment

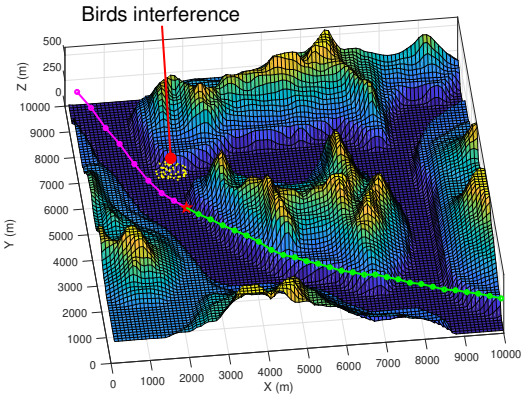
Figure 11. 2D and 3D visualization of dynamic obstacle avoidance processes in the virtual map.



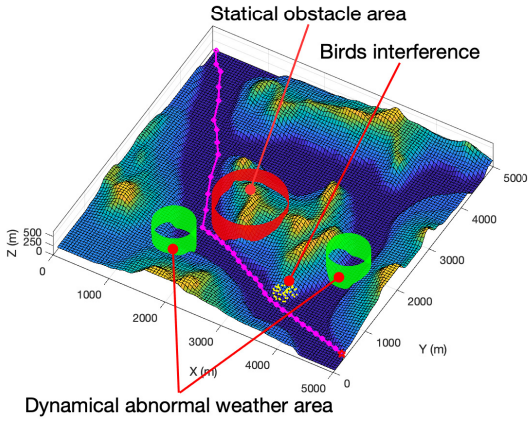
(**a**) Planned path in an environment with dynamical abnormal weather area



(**b**) Planned path in an environment with statical obstacle area



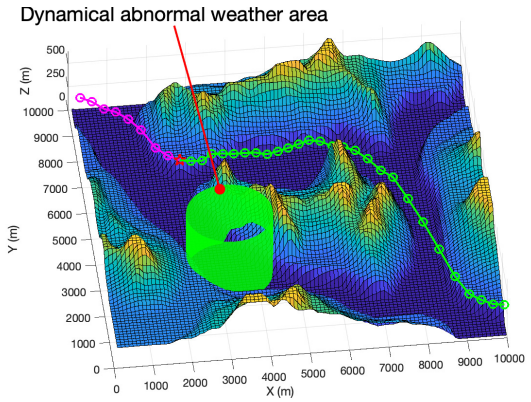
(**c**) Planned path in an environment with the interference of birds



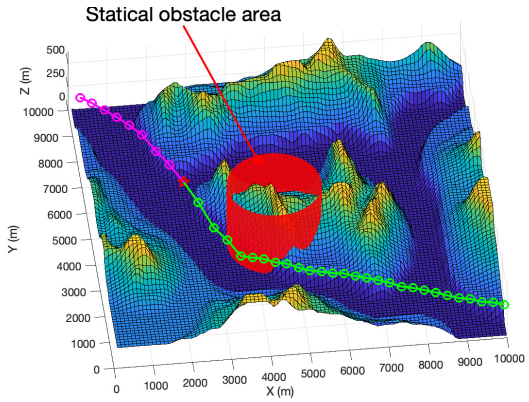
(**d**) Planned path in an environment with mixed obstacles

Figure 12. 3D visualization of small size (with diameter being 35 m) UAV with various types of obstacle avoidance in the virtual map.

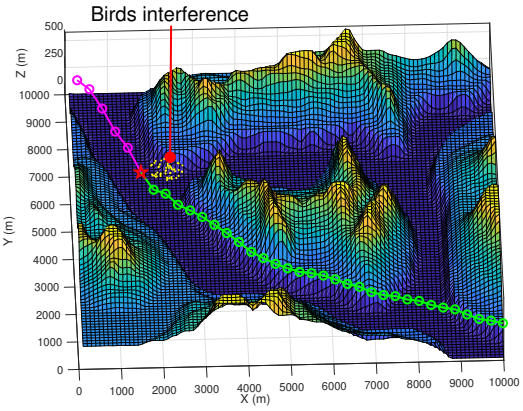
In Figure 11, the magenta circles and line segments indicate where the UAV has passed, and they will not change. Figure 11a–d show the planning effect of the algorithm in 2D space, and experimental results show that the planned path can avoid obstacles and has a short total path length. We mapped the planned paths in the 2D environment to the 3D environment. Figure 11e–h show our experimental results for dynamic obstacles avoidance in high-resolution 3D maps. The golden sphere represents an obstacle in the 3D environment that exists in the minimum threat surface of the UAV.



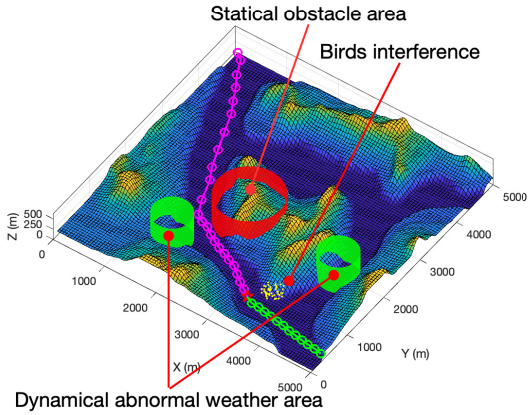
(a) Planned path in an environment with dynamical abnormal weather area



(b) Planned path in an environment with statical obstacle area



(c) Planned path in an environment with the interference of birds



(d) Planned path in an environment with mixed obstacles

Figure 13. 3D visualization of a large (with diameter being 100 m) UAV with various types of obstacle avoidance in the virtual map.

From the experiment, we know that OABAS can perform dynamic obstacle avoidance and path planning tasks in high-resolution maps. In the experiment, the time required for each re-planning took less than one second. For example, the MiG-25 is one of the fastest flying fighters in the world [58]. It has a maximum flying speed of up to 1200 km/h at the sea level distance and can fly 333 m in one second. The OABAS algorithm can quickly generate a longer track during one second, so there is plenty of time to calculate the trajectory during the flight.

In general, surface-to-air missiles are mostly guided by radar or radar and infrared. Since the missile poses a significant threat to the aircraft, we assume that the radar's detection range is a static cylindrical area whose radius is the radar's detection range. In addition to this, birds often pose a potential threat to UAV in the air. When the airborne radar can detect nearby birds, we establish a minimum circle according to its distribution [59] and consider it as a single obstacle. Building a regular model for an irregular obstacle simplifies the calculation process while avoiding the algorithm spending too much time building the model. The flying speed of birds is between 40 and 80 km/h, and our experiment set it to 80 km/h. In addition to ground threats and birds, there are also unusual weather areas in space. Due to the complexity of the weather, we use a green cylindrical area to indicate that there is unusual weather in the area. Referring to the general typhoon moving speed, we set the moving speed to 36 km/h.

In Figure 12a, we use the green cylinder area to represent the abnormal weather area. We plan an area of abnormal weather to move at a speed of 36 km/h, and the moving direction was random.

From the graphical results, our path planning algorithm can effectively plan the path to avoid the abnormal weather area. In Figure 12b, we use a red cylinder to represent a static obstacle, such as surface-to-air missile threats and radar detection from the ground. As can be seen from the Figure 12b, our planned path can avoid dangerous areas and plan a good path. In Figure 12c, we use yellow scatter to represent the flock of birds. The speed at which the birds move was set to 80 km/h, and their direction of movement is random. From the experimental results, we can see that our algorithm can effectively avoid the interference of birds. In Figure 12d, we mixed a variety of obstacles for testing. Specifically, it includes the ground threat represented by the red cylinder, the abnormal weather area indicated by the green cylinder, and the bird group represented by the yellow scatter. From the experimental results, we can still get a safe and short path.

In Figure 13, we used a larger size of the aircraft for obstacle avoidance experiments. Hollow circles with a diameter being 100 m was established in the experiment to represent the aircraft model. Even though the model we use was much larger than the real model of an ordinary drone, we can still observe in the experiment that the path planned by the OABAS algorithm can maintain a safe distance from the obstacles.

5. Conclusions

This paper has proposed a novel path planning algorithm called OABAS algorithm. Based on the BAS strategy, a new UAVs path planner has been proposed. The constraints used in this algorithm have taken into account the requirements of shorter path lengths, maximum turning angles, and obstacle avoidance. This paper has combined the MTS to plan for efficient and secure requirements. Besides, we have applied this intelligent path planning algorithm to UAVs simulations. Moreover, this paper has verified the universal applicability of OABAS algorithm obstacle avoidance and compared it with two other bio-heuristic algorithms to prove the effectiveness of the algorithm. As a final remark of this paper, to the best of authors' knowledge, this is the first work in the field of intelligent optimization that can elegantly design an effective and fast path planner by leveraging the proposed OABAS algorithm.

Author Contributions: Conceptualization, W.Q. and X.S.; methodology, W.Q. and X.S.; software, W.Q. and D.C.; validation, W.Q., Y.J. and Z.C.; formal analysis, W.Q. and X.S.; investigation, W.Q. and A.H.K.; resources, W.Q. and X.S.; data curation, W.Q. and S.L.; writing—original draft preparation, W.Q.; writing—review and editing, W.Q. and X.S.; visualization, W.Q. and X.S.; supervision, W.Q. and X.S.; project administration, W.Q. and X.S.; funding acquisition, X.S. and D.C.

Funding: This work is supported by the National Natural Science Foundation of China (with numbers 61401385 and 61702146), by Hong Kong Research Grants Council Early Career Scheme (with number 25214015), by Departmental General Research Fund of Hong Kong Polytechnic University (with number G.61.37.UA7L), and also by PolyU Central Research Grant (with number G-YBMU).

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

References

1. Cai, G.; Dias, J.; Seneviratne, L. A Survey of Small-Scale Unmanned Aerial Vehicles: Recent Advances and Future Development Trends. *Unmanned Syst.* **2014**, *02*, 175–199. [CrossRef]
2. Ahmad, K.Y.; AlMajali, A.; Ghalyon, S.A.; Dweik, W.; Mohd, B.J. Analyzing Cyber-Physical Threats on Robotic Platforms. *Sensors* **2018**, *18* 1643.
3. Gupta, S.G.; Ghonge, M.M.; Jawandhiya, P. Review of unmanned aircraft system (UAS). *Int. J. Adv. Res. Comput. Eng. Technol.* **2013**, *2*, 1646–1658.
4. Guerrero-Castellanos, J.; Madrigal-Sastre, H.; Durand, S.; Torres, L.; Muñoz-Hernández, G. A robust nonlinear observer for real-time attitude estimation using low-cost MEMS inertial sensors. *Sensors* **2013**, *13*, 15138–15158. [CrossRef] [PubMed]
5. Yang, G.; Kapila, V. Optimal path planning for unmanned air vehicles with kinematic and tactical constraints. In Proceedings of the 41st IEEE Conference on Decision and Control, Las Vegas, NV, USA, 10–13 December 2002; pp. 1301–1306.

6. De La Iglesia, I.; Hernandez-Jayo, U.; Osaba, E.; Carballedo, R. Smart Bandwidth Assignation in an Underlay Cellular Network for Internet of Vehicles. *Sensors* **2017**, *17*, 2217. [CrossRef]
7. Jin, L.; Li, S. Distributed task allocation of multiple robots: A control perspective. *IEEE Trans. Syst. Man Cybern. Syst.* **2018**, *48*, 693–701. [CrossRef]
8. Yan, F.; Liu, Y.S.; Xiao, J.Z. Path Planning in Complex 3D Environments Using a Probabilistic Roadmap Method. *Int. J. Autom. Comput.* **2013**, *10*, 525–533. [CrossRef]
9. Cho, Y.; Kim, D.; Kim, D.S. Topology representation for the Voronoi diagram of 3D spheres. *Int. J. CAD/CAM* **2009**, *5*, 59–68,
10. Geraerts, R. Planning short paths with clearance using explicit corridors. In Proceedings of the 2010 IEEE International Conference on Communication, Cape Town, South Africa, 23–27 May 2010; pp. 1997–2004.
11. Vadakkepat, P.; Tan, K.C.; Ming-Liang, W. Evolutionary artificial potential fields and their application in real time robot path planning. In Proceedings of the 2000 Congress on Evolutionary Computation, La Jolla, CA, USA, 16–19 July 2000; pp. 256–263.
12. Musliman, I.A.; Rahman, A.A.; Coors, V. Implementing 3D network analysis in 3D GIS. *Int. Arch. ISPRS* **2008**, *37*, 913–918.
13. Carsten, J.; Ferguson, D.; Stentz, A. 3d field d: Improved path planning and replanning in three dimensions. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China, 9–15 October 2006; pp. 3381–3386.
14. Geem, Z.W.; Kim, J.H.; Loganathan, G.V. A new heuristic optimization algorithm: Harmony search. *Simulation* **2001**, *76*, 60–68. [CrossRef]
15. De Filippis, L.; Guglieri, G.; Quagliotti, F. Path planning strategies for UAVS in 3D environments. *J. Intell. Robot. Syst.* **2012**, *65*, 247–264. [CrossRef]
16. Valente, J.; Del Cerro, J.; Barrientos, A.; Sanz, D. Aerial coverage optimization in precision agriculture management: A musical harmony inspired approach. *Comput. Electron. Agric.* **2013**, *99*, 153–159. [CrossRef]
17. Miller, B.; Stepanyan, K.; Miller, A.; Andreev, M. 3D path planning in a threat environment. In Proceedings of the IEEE Conference on Decision and Control and European Control Conference, Orlando, FL, USA, 12–15 December 2011; pp. 6864–6869.
18. Chamseddine, A.; Zhang, Y.; Rabbath, C.A.; Join, C.; Theilliol, D. Flatness-based trajectory planning/replanning for a quadrotor unmanned aerial vehicle. *IEEE Trans. Aerosp. Electron. Syst.* **2012**, *48*, 2832–2848. [CrossRef]
19. Chen, D.; Li, S.; Wu, Q. Rejecting Chaotic Disturbances Using a Super-Exponential-Zeroing Neurodynamic Approach for Synchronization of Chaotic Sensor Systems. *Sensors* **2019**, *19*, 74. [CrossRef] [PubMed]
20. Volkan Pehlivanoglu, Y.; Baysal, O.; Hacioglu, A. Path planning for autonomous UAV via vibrational genetic algorithm. *Aircr. Eng. Aerosp. Technol.* **2007**, *79*, 352–359. [CrossRef]
21. Jin, L.; Zhang, Y.; Li, S.; Zhang, Y. Modified ZNN for time-varying quadratic programming with inherent tolerance to noises and its application to kinematic redundancy resolution of robot manipulators. *IEEE Trans. Ind. Electron.* **2016**, *63*, 6978–6988. [CrossRef]
22. Okdem, S.; Karaboga, D. Routing in wireless sensor networks using an ant colony optimization (ACO) router chip. *Sensors* **2009**, *9*, 909–921. [CrossRef] [PubMed]
23. Hassanzadeh, I.; Madani, K.; Badamchizadeh, M.A. Mobile robot path planning based on shuffled frog leaping optimization algorithm. In Proceedings of the 2010 IEEE International Conference on Automation Science and Engineering, Toronto, ON, Canada, 21–24 August 2010; pp. 680–685.
24. Liu, L.; Zhang, S. Voronoi diagram and GIS-based 3D path planning. In Proceedings of the 2009 17th International Conference on Geoinformatics, Washington, DC, USA, 12–14 August 2009; pp. 1–5.
25. Schøler, F.; la Cour-Harbo, A.; Bisgaard, M. Generating approximative minimum length paths in 3D for UAVs. In Proceedings of the 2012 IEEE Intelligent Vehicles Symposium IEEE, Alcala de Henares, Spain, 3–7 June 2012; pp. 229–233.
26. Chen, D.; Zhang, Y.; Li, S. Zeroing neural-dynamics approach and its robust and rapid solution for parallel robot manipulators against superposition of multiple disturbances. *Neurocomputing* **2018**, *275*, 845–858. [CrossRef]
27. Kroumov, V.; Yu, J.; Shibayama, K. 3D path planning for mobile robots using simulated annealing neural network. *Int. J. Innov. Comput. Inf. Control* **2010**, *6*, 2885–2899.
28. Passino, K.M. Bacterial foraging optimization. *Int. J. Swarm Intell. Res. (IJSIR)* **2010**, *1*, 1–16. [CrossRef]

29. Yang, X.S. A new metaheuristic bat-inspired algorithm. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*; Springer: Berlin, Germany, 2010; pp. 65–74.
30. Li, S.; He, J.; Li, Y.; Rafique, M.U. Distributed recurrent neural networks for cooperative control of manipulators: A game-theoretic perspective. *IEEE Trans. Neural Networks Learn. Syst.* **2017**, *28*, 415–426. [CrossRef] [PubMed]
31. Gawel, A.; Dubé, R.; Surmann, H.; Nieto, J.; Siegwart, R.; Cadena, C. 3D registration of aerial and ground robots for disaster response: An evaluation of features, descriptors, and transformation estimation. In Proceedings of the SSRR 2017-IEEE International Symposium on Safety, Security and Rescue Robotics, Shanghai, China, 11–13 October 2017; pp. 27–34.
32. Xu, C.; Duan, H.; Liu, F. Chaotic artificial bee colony approach to Uninhabited Combat Air Vehicle (UCAV) path planning. *Aerosp. Sci. Technol.* **2010**, *14*, 535–541. [CrossRef]
33. Duan, H.; Yu, Y.; Zhang, X.; Shao, S. Three-dimension path planning for UCAV using hybrid meta-heuristic ACO-DE algorithm. *Simul. Model. Pract. Theory* **2010**, *18*, 1104–1115. [CrossRef]
34. 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]
35. Mittal, S.; Deb, K. Three-dimensional offline path planning for UAVs using multiobjective evolutionary algorithms. In Proceedings of the IEEE Congress on Evolutionary Computation, Singapore, 25–28 September 2007; pp. 3195–3202.
36. Roberge, V.; Tarbouchi, M.; Labonté, 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]
37. Hoffman, K.L.; Padberg, M. Traveling Salesman Problem (TSP) Traveling Salesman Problem. In *Encyclopedia of Operations Research and Management Science*; Springer: Berlin, Germany, 2001; pp. 849–853.
38. Eiselt, H.A.; Gendreau, M.; Laporte, G. Arc routing problems, part I: The Chinese postman problem. *Oper. Res.* **1995**, *43*, 231–242. [CrossRef]
39. Chen, D.; Zhang, Y. A hybrid multi-objective scheme applied to redundant robot manipulators. *IEEE Trans. Autom. Sci. Eng.* **2017**, *14*, 1337–1350. [CrossRef]
40. Li, S.; Zhang, Y.; Jin, L. Kinematic control of redundant manipulators using neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, *28*, 2243–2254. [CrossRef] [PubMed]
41. Guo, D.; Xu, F.; Yan, L. New Pseudoinverse-Based Path-Planning Scheme With PID Characteristic for Redundant Robot Manipulators in the Presence of Noise. *IEEE Trans. Control Syst. Technol.* **2017**, *26*, 2008–2019. [CrossRef]
42. Chen, D.; Zhang, Y. Robust Zeroing Neural-Dynamics and Its Time-Varying Disturbances Suppression Model Applied to Mobile Robot Manipulators. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 4385–4397.
43. Chen, D.; Zhang, Y.; Li, S. Tracking Control of Robot Manipulators with Unknown Models: A Jacobian-Matrix-Adaption Method. *IEEE Trans. Ind. Infor.* **2018**, *14*, 3044–3053. [CrossRef]
44. Kirkpatrick, S.; Gelatt, C.D.; Vecchi, M.P. Optimization by simulated annealing. *Science* **1983**, *220*, 671–680. [CrossRef] [PubMed]
45. Seet, B.C.; Liu, G.; Lee, B.S.; Foh, C.H.; Wong, K.J.; Lee, K.K. A-STAR: A mobile ad hoc routing strategy for metropolis vehicular communications. *Lect. Notes Comput. Sci.* **2004**, *3042*, 989–999.
46. Nikolos, I.K.; Zografos, E.S.; Brintaki, A.N. UAV path planning using evolutionary algorithms. In *Innovations in Intelligent Machines-1*; Springer: Berlin, Germany, 2007; pp. 77–111.
47. Menon, P.K.A.; Cheng, V.L.; Kim, E. Optimal trajectory synthesis for terrain-following flight. *J. Guid. Control. Dyn.* **1991**, *14*, 807–813. [CrossRef]
48. Choset, H.M.; Hutchinson, S.; Lynch, K.M.; Kantor, G.; Burgard, W.; Kavraki, L.E.; Thrun, S. *Principles of Robot Motion: Theory, Algorithms, and Implementation*; MIT Press: Cambridge, MA, USA, 2005.
49. Schøler, F. 3d path Planning for autonomous Aerial Vehicles in Constrained Spaces. Ph.D. Thesis, Section of Automation & Control, Department of Electronic Systems, Aalborg University, Aalborg, Denmark, 2012.
50. Karaman, S.; Frazzoli, E. Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.* **2011**, *30*, 846–894. [CrossRef]
51. Yang, L.; Qi, J.; Xiao, J.; Yong, X. A literature review of UAV 3D path planning. In Proceedings of the WCICA 2014-The 11th World Congress on Intelligent Control and Automation, Shenyang, China, 29 June–4 July 2014; pp. 2376–2381.

52. Asseo, S.J. Terrain following/terrain avoidance path optimization using the method of steepest descent. In Proceedings of the IEEE 1988 National Aerospace and Electronics Conference, NAECON 1988, Dayton, OH, USA, 23–27 May 1988; pp. 1128–1136.
53. Wendl, M.; Katt, D.; Young, G. Advanced automatic terrain following/terrain avoidance control concepts study. In Proceedings of the IEEE 1982 National Aerospace and Electronics Conference, NAECON 1982, Dayton, OH, USA, 18–20 May 1982; pp. 18–20.
54. Harrington, W. TF/TA System Design Evaluation Using Pilot-in-the-Loop Simulations: The Cockpit Design Challenge. In Proceedings of the 1984 SAE Aerospace Congress & Exposition, Long Beach, CA, USA, 15–18 October 1984.
55. Avellar, G.; Pereira, G.; Pimenta, L.; Iscold, P. Multi-UAV routing for area coverage and remote sensing with minimum time. *Sensors* **2015**, *15*, 27783–27803. [CrossRef]
56. Jiang, X.; Li, S. BAS: Beetle Antennae Search Algorithm for Optimization Problems. *Int. J. Robot. Control* **2018**, *1*, 1–2. [CrossRef]
57. Jenson, S.K.; Domingue, J.O. Extracting topographic structure from digital elevation data for geographic information system analysis. *Photogramm. Eng. Remote Sens.* **1988**, *54*, 1593–1600.
58. Anderson, J.D.; Hunter, L.P. Introduction to flight. *Phys. Today* **1987**, *40*, 125. [CrossRef]
59. Har-Peled, S.; Mazumdar, S. Fast algorithms for computing the smallest k-enclosing circle. *Algorithmica* **2005**, *41*, 147–157. [CrossRef]