A Modified Firefly Algorithm for UCAV Path Planning

Gaige Wang[1,2], Lihong Guo[1], Hong Duan[3,*], Luo Liu[1,2] and Heqi Wang[1]

[1]Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China
[2]Graduate School of Chinese Academy of Sciences, Beijing 100039, China
[3]School of Computer Science and Information Technology, Northeast Normal University, Changchun 130117, China
gaigewang@163.com; guolh@ciomp.ac.cn; duanh272@163.com; lllzxx@sina.com; whq200808@gail.com

Abstract

Path planning for uninhabited combat air vehicle (UCAV) is a complicated high dimension optimization problem, which mainly centralizes on optimizing the flight route considering the different kinds of constrains under complicated battle field environments. Original firefly algorithm (FA) is used to solve the UCAV path planning problem. Furthermore, a new modified firefly algorithm (MFA) is proposed to solve the UCAV path planning problem, and a modification is applied to exchange information between top fireflies during the process of the light intensity updating. Then, the UCAV can find the safe path by connecting the chosen nodes of the coordinates while avoiding the threat areas and costing minimum fuel. This new approach can accelerate the global convergence speed while preserving the strong robustness of the basic FA. The realization procedure for original FA and this improved meta-heuristic approach MFA is also presented. To prove the performance of this proposed meta-heuristic method, MFA was compared with FA and other population-based optimization methods, such as, ACO, BBO, DE, ES, GA, PBIL, PSO and SGA. The experiment shows that the proposed approach is more effective and feasible in UCAV path planning than the other model.

Keywords: *Unmanned combat air vehicle (UCAV); Path planning; Firefly algorithm (FA)*

1. Introduction

In modern warfare, uninhabited combat air vehicle (UCAV) has potential, strong advantage over manned fighter in complicated battlefield. Since the U.S. Air Force and Department of Defense developed the first UCAV X245A all over the world, UCAV has been a new means in air precision strike weapon system replacing manned aircraft to perform attack missions under the risky complicated battle field environments. Path planning and trajectory generation is one of the key technologies in coordinated UCAV combatting. The flight path planning in a large mission area is a typical large scale optimization problem, and a series of algorithms have been proposed to solve this complicated multi-constrained optimization problem, such as differential evolution [1], genetic algorithm [2] and ant colony optimization algorithm [3] and its variant [4, 5], chaotic artificial bee colony [6], and intelligent water drops optimization [7]. However, those methods can hardly solve the contradiction between the global optimization and excessive information.

The firefly algorithm (FA) is a metaheuristic algorithm, inspired by the flashing behavior of fireflies [8]. The primary purpose for a firefly's flash is to act as a signal system to appeal to other fireflies. Recent studies shows that the firefly algorithm is very efficient [9], and could outperform the other metaheuristic algorithms including particle swarm optimization [10]. A clear demonstration of superiority of firefly algorithm over particle swarm optimization has been carried out by A. Chatterjee and G.K. Mahanti [11]. In addition, firefly algorithm can efficiently solve non-convex problems with complex nonlinear constraints [12]. Further improvement on the performance is also possible with promising results [13].

However, in the field of path planning for UCAV, no application of FA algorithm exists yet. In this paper, we use an original FA and an improved FA algorithm to solve UCAV path planning problem. Here, we add information exchange between top fireflies to propose a new meta-heuristic algorithm according to the principle of FA, and then an improved FA algorithm is used to search the optimal or sub-optimal route with complicated multi-constraints. To investigate the feasibility and effectiveness of our proposed approach, it was compared with FA and other population-based optimization methods, such as, ACO, BBO, DE, ES, GA, PBIL, PSO and SGA under complicated combating environments. The simulation experiments indicate that our hybrid meta-heuristic method can generate a feasible optimal route for UCAV more effectively than other population-based optimization methods.

The rest of this paper is structured as follows: Section 2 describes the mathematical model in UCAV path planning problem. Subsequently, the principle of the basic FA is explained in Section 3, and then an improved modified FA for UCAV path planning is presented in Section 4 and the detailed implementation procedure is also described in this section. The simulation experiment is conducted in Section 5. We terminate our paper with conclusion and future work in Section 6.

2. Mathematical Model in UCAV Path Planning

Path planning for UCAV is a new low altitude penetration technology to achieve the purpose of terrain-following, terrain avoidance and flight with evading threat, which is a key component of mission planning system [14]. The goal for path planning is to calculate the optimal or sub-optimal flight route for UCAV within the appropriate time, which enables the UCAV to break through the enemy threat environments, and self-survive with the perfect completion of mission. In my work, we use the mathematical model in UCAV path planning in [1], which is described as follows.

2.1. Problem Description

Path planning for UCAV is the design of optimal flight route to meet certain performance requirements according to the special mission objective, and is modeled by the constraints of the terrain, data, threat information, fuel and time [15]. In this paper, firstly the route planning problem is transformed into a *D*-dimensional function optimization problem (Figure 1).

In Figure 1, we transform the original coordinate system into new coordinate whose horizontal axis is the connection line from starting point to target point according to transform expressions shown in Equation (1) and (2), where, the point (x, y) is coordinate in the original ground coordinate system O_{XY}; the point (x', y') is coordinate in the new rotating coordinate system $O_{X'Y'}$; θ is the rotation angle of the coordinate system.

$$\theta = \arcsin \frac{y_2 - y_1}{|\overrightarrow{AB}|} \tag{1}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \bullet \begin{pmatrix} x' \\ y' \end{pmatrix} + \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} \tag{2}$$

Then, we divide the horizontal axis X' into D equal partitions and then optimize vertical coordinate Y' on the vertical line for each node to get a group of points composed by vertical coordinate of D points. Obviously, it is easy to get the horizontal abscissas of these points. We can get a path from start point to end point through connecting these points together, so that the route planning problem is transformed into a D-dimensional function optimization problem.

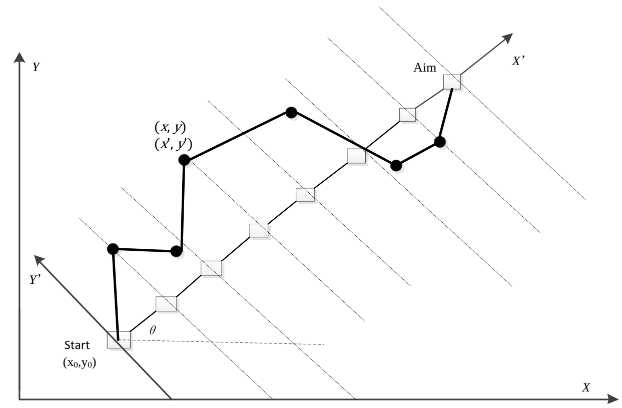


Figure 1. Coordinates Transformation Relation

2.2. Performance Indicator

A performance indicator of path planning for UCAV mainly contains the completion of the mandate of the safety performance indicator and fuel performance indicator, *i.e.*, indicators with the least threat and the least fuel.

Minimum of performance indicator for threat

$$\min J_t = \int_0^L w_t \mathrm{d}l \text{ , } L \text{ is the length of the path} \tag{3}$$

Minimum of performance indicator for fuel

$$\min J_f = \int_0^L w_f \mathrm{d}l \text{ , } L \text{ is the length of the path} \tag{4}$$

Then the total performance indicators for UCAV route

$$\min J = kJ_t + (1-k)J_f \tag{5}$$

Where, w_t is the threat cost for each point on the route; w_f is fuel cost for each point on the path which depends on path length (in this paper, $w_f \equiv 1$); $k \in [0,1]$, is balanced coefficient between safety performance and fuel performance, whose value is determined by the special task UCAV performing, *i.e.*, if flight safety is of highly vital importance to the task, then we choose a larger k, while if the speed is critical to the aircraft task, then we select a smaller k.

2.3. Threat Cost

When the UCAV is flying along the path L_{ij}, the total threat cost generated by N_t threats is calculated as follows:

$$w_{t,L_{ij}} = \int_0^{L_{ij}} \sum_{k=1}^{N_t} \frac{t_k}{[(x-x_k)^2+(y-y_k)^2]^2} \mathrm{d}l \tag{6}$$

To simplify the calculations (as shown in Figure 2), each edge is divided into five equal partitions, and threat cost of five points on this edge is calculated. If the distance from the threat point to the edge is within threat radius, we can calculate the responding threat cost according to Equation (**7**).

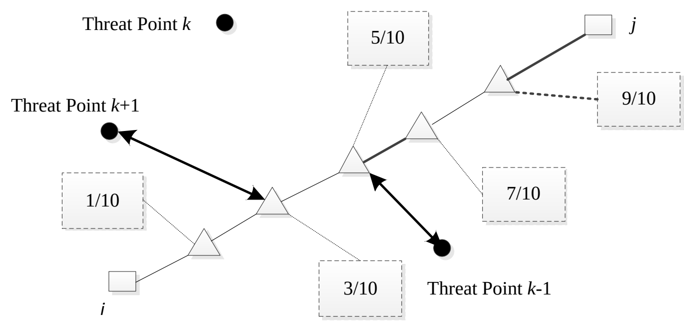


Figure 2. Calculation for Threat Cost

$$w_{t,L_{ij}} = \frac{L_{ij}^5}{5} \sum_{k=1}^{N_t} t_k \left(\frac{1}{d_{0.1,k}^4} + \frac{1}{d_{0.3,k}^4} + \frac{1}{d_{0.5,k}^4} + \frac{1}{d_{0.7,k}^4} + \frac{1}{d_{0.9,k}^4} \right) \tag{7}$$

where, L_{ij} is the length of the sub-segment connecting node i and node j; $d_{0.1,k}$ is the distance from the 1/10 point on the sub-segment L_{ij} to the kth threat; t_k is threat level of the kth threat. Moreover, it can simply consider the fuel cost w_f to L. As fuel cost related to flight length, we can consider w_f=L, for simplicity, and fuel cost of each edge can be expressed by $w_{f,L_{ij}} = L_{ij}$.

3. Firefly Algorithm (FA)

The firefly algorithm is a new swarm intelligence optimization method, in which the search algorithm is inspired by social behavior of fireflies and the phenomenon of bioluminescent communication. There are two important issues in the firefly algorithm that are the variation of light intensity and formulation of attractiveness.

3.1 Mainframe of FA

Firefly algorithm imitates the social behavior of fireflies flying in the tropical summer sky. Fireflies communicate, search for pray and find mates using bioluminescence with varied flashing patterns. By mimicking nature, various metaheuristic algorithms can be designed. Some of the flashing characteristics of fireflies were idealized so as to design a firefly-inspired algorithm. For simplicity, only three rules were followed:

(1) All fireflies are unisex so that one firefly will be attracted by other fireflies regardless of their sex.

(2) Attractiveness is proportional to firefly brightness. For any pair of flashing fireflies, the less bright one will move towards the brighter one. Attractiveness is proportional to the brightness which decreases with increasing distance between fireflies. If there are no brighter fireflies than a particular firefly, this individual will move at random in the space.

(3) The brightness of a firefly is influenced or determined by the objective function.

For a maximization problem, brightness can simply be proportional to the value of the cost function. Other forms of brightness can be defined in a similar way to the fitness function in genetic algorithms. The basic steps of the FA are summarized by the pseudo code shown in Algorithm 1 which consists of the three rules discussed above. It should be noted that there is some conceptual similarity between the firefly algorithms and the bacterial foraging algorithm (BFA) [16].

Algorithm 1 *The algorithm of firefly algorithm*

Begin

Step 1: Initialization. *Set the generation counter G = 1; Initialize the population of n fireflies P randomly and each firefly corresponding to a potential solution to the given problem; define light absorption coefficient γ; set controlling the step size α and the initial attractiveness β_0 at r = 0.*

Step 2: *Evaluate the light intensity I for each firefly in P determined by f(x)*

Step 3: While *the termination criteria is not satisfied* **or** *G < MaxGeneration* **do**

  for *i=1:n (all n fireflies)* **do**

    for *j=1:n (n fireflies)* **do**

      if *($I_j < I_i$),*

        move firefly i towards j;

      end if

    Vary attractiveness with distance r via exp[-γr^2];

    Evaluate new solutions and update light intensity;

    end for *j*

  end for *i*

  G = G+1;

Step 4: end while

Step 5: *Post-processing the results and visualization;*

End.

Algorithm 1. The Algorithm of Firefly Algorithm

The main update formula for any couple of two fireflies x_i and x_j is

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2}(x_i^t - x_j^t) + \alpha \varepsilon_i^t \tag{8}$$

where α is a parameter controlling the step size, β_0 is the attractiveness at $r = 0$, the second term is due to the attraction, while the third term is randomization with the vector of random variables ε_i being drawn from a Gaussian distribution or other distribution. The distance between any two fireflies i and j at x_i and x_j can be the Cartesian distance $r_{ij}= ||x_i-x_j||_2$ or the l_2-norm. For other applications such as scheduling, the distance can be time delay or any suitable forms, not necessarily the Cartesian distance. For most cases in our implementation, we can take $\beta_0 = 1$, $\alpha \in [0, 1]$, and $\gamma = 1$. It can be shown that the limiting case $\gamma \rightarrow 0$ corresponds to the standard Particle

Swarm Optimization (PSO). In fact, if the inner loop (for *j*) is removed and the brightness I_j is replaced by the current global best g^*, then FA essentially becomes the standard PSO.

3.2 Algorithm FA for UCAV Path Planning

In FA, the standard ordinates are inconvenient to solve UCAV path planning directly. In order to apply FA to UCAV path planning, one of the key issues is to transform the original ordinate into rotation ordinate by Equation (1) and (2).

Light intensity of firefly *i* is determined by the threat cost by Equation (5), and the smaller the threat cost, the brighter the light intensity of firefly *i*. Each firefly is encoded by *D*-dimensional deciding variables. And then, we use FA to optimize the path planning to get the best solution that is optimal flight route for UCAV. At last, the best solution is inversely converted to the original ordinates and output. The algorithm FA for UCAV path planning is shown in Algorithm 2.

Algorithm 2 *Algorithm of FA for UCAV path planning*

Begin

Step 1: **Initialization.** *Set the generation counter G = 1; Initialize the population of fireflies P randomly and each firefly corresponding to a potential solution to the given problem; define light absorption coefficient γ; set controlling the step size α and the initial attractiveness β_0 at r = 0.*

Step 2: Generating rotation coordinate system. *Transform the original coordinate system into new rotation coordinate whose horizontal axis is the connection line from starting point to target point according to* **Equation (1)** *and* **(2)**; *convert battlefield threat information to the rotation coordinate system and divide the axis X' into D equal partitions. Each feasible solution, denoted by $\boldsymbol{P}=\{p_1, p_2, \cdots, p_D\}$, is an array indicated by the composition of D coordinates which are the floating-point numbers*

Step 3: *Evaluate the threat cost/light intensity J for each firefly in P by* **Equation (5)**

Step 4: while *The halting criteria is not satisfied or G < MaxGeneration* **do**

```
        for i=1:n (all n fireflies) do
            for j=1:n (n fireflies) do
                if (J_j < I_i),
                  Move firefly i towards j;
                end if
            Vary attractiveness with distance r via exp[-γr^2];
            Evaluate new solutions and update threat cost/light intensity;
            end for j
        end for i
        G = G+1
```

Step 5: end while

Step 6: *Inversely transform the coordinates in final optimal path into the original coordinate, and output*

End.

Algorithm 2. Algorithm of FA for UCAV Path Planning

4. Modified Firefly Algorithm (MFA)

Due to different fireflies working almost independently, it may lack the exchange information between top fireflies. Therefore, in this paper, we add information exchange between top fireflies to the FA during the process of the light intensity updating. And then, the MFA algorithm is proposed based on exchange information between top fireflies to solve the UCAV path planning.

4.1 Mainframe of MFA

For firefly algorithm, as the search relies entirely on random walks, a fast convergence cannot be guaranteed. Described here for the first time, a main modification of adding the handling of top fireflies is made to the FA, including two minor modifications, which are made with the aim of speeding up convergence, thus making the method more practical for a wider range of applications but without losing the attractive features of the original method.

```
Algorithm 3   The algorithm of exchanging information among top fireflies
Begin
    Set max Lévy flight step size A and golden ratio φ.
    for i=1:NoTop (all top fireflies) do
        Current firefly at position x_i
        Pick another firefly from the top fireflies at random x_j
        if x_i = x_j then
            Calculate Lévy flight step size α ← A/G^2
            Perform Lévy flight from x_i to generate new firefly x_k
            Evaluate the light intensity I_k for firefly x_k by I_k = f(x_k)
            Choose a random firefly l from all fireflies
            if (I_k > I_l)
                Move firefly k towards l;
            end if
        else
            dx = |x_i - x_j| / φ
            Move distance dx from the worst firefly to the best firefly to find x_k
            Evaluate the light intensity I_k for firefly x_k by I_k = f(x_k)
            Choose a random firefly l from all fireflies
            if (I_k > I_l) then
                Move firefly k towards l;
            end if
        end if
    end for i
End.
```

Algorithm 3. The Algorithm of Exchanging Information among Top Fireflies

The first modification is adding Lévy flight to the FA with the step size α. Moreover, in the MFA, the value of α decreases as the number of generations increases. This is done for the same reasons that the inertia constant is reduced in the PSO [17], *i.e.*, to encourage more localized searching as the individuals, or the fireflies, get closer to the solution. The second modification is to add information exchange between the fireflies in an attempt to accelerate the convergence speed to optima. In the FA, there is no information exchange between individuals and the searches are performed independently in essence, *i.e.*, different fireflies work almost independently [18]. In the MFA, a portion of the fireflies with the brightest light intensity are made up of a group of top fireflies. For each of the top fireflies, a second firefly in this group is picked at random and a new firefly is then generated on the line connecting these two top fireflies. The distance along this line at which the new firefly is located is computed, using the inverse of the golden ratio $\varphi = (1+\sqrt{5})/2$, such that it is closer to the firefly with the brightest light intensity. In the case that both fireflies have the same light intensity, the new firefly is produced at the middle point. There is a possibility that, in this step, the same firefly is picked twice. In this case, a local Lévy flight search is performed from the randomly picked firefly with step size $\alpha = A/G^2$. The steps for exchanging information between top fireflies involved in the modified firefly algorithm are shown in detail in Algorithm 3, and the basic framework of modified firefly algorithm can be simply described in Algorithm 4. There are two parameters, the parameter of light absorption coefficient and the fraction of fireflies to make up the top fireflies, which need to be adjusted in the MFA. Through testing on path planning for UCAV, it was found that setting the parameter of light absorption coefficient to 1.0 and the fraction of fireflies placed in the top fireflies group to 0.25 produced the best results through a series of simulation experiments.

Algorithm 4 *The algorithm of modified firefly algorithm*

Begin

Step 1: Initialization. *Set the generation counter G = 1; Initialize the population of n fireflies P randomly and each firefly corresponding to a potential solution to the given problem; define light absorption coefficient γ; set controlling the step size α and the initial attractiveness β_0 at r = 0; set max step size A, golden ratio φ and discovery rate p_a.*

Step 2: *Evaluate the light intensity I for each firefly in P determined by f(x).*

Step 3: While *the termination criteria is not satisfied* **or** *G < MaxGeneration* **do**

Sort the population of fireflies P from best to worst by order of light intensity I for each firefly;

Exchange information between top fireflies by **Algorithm 3**

for *i=1:n (all n fireflies)* **do**

  for *j=1:n (n fireflies)* **do**

    if *($I_j < I_i$),*

      Move firefly i towards j;

    end if

    Vary attractiveness with distance r via exp[$-\gamma r^2$];

    Evaluate new solutions and update light intensity;

  end for *j*

end for *i*

Evaluate the light intensity I for each firefly in P determined by f(x).

Sort the population of fireflies P from best to worst by order of light intensity I for each firefly;

G = G+1;

Step 4: end while

Step 5: *Post-processing the results and visualization;*

Algorithm 4. The Algorithm of Modified Firefly Algorithm

4.2 Algorithm MFA for UCAV Path Planning

Improved MFA can adapt to the needs of UCAV path planning, while optimization algorithms can improve the FA fast search capabilities, accelerate convergence and increase the search to the global possible optimum solution. Light intensity of firefly *i* is represented by the objective function shown as Equation (5) in UCAV path planning model, the smaller the threat cost, then the brighter the light intensity of firefly *i*.

Based on the above analysis, the pseudo code of MFA for UCAV path planning is described as follows (Algorithm 5).

Algorithm 5 *Algorithm of MFA for UCAV path planning*

Begin

Step 1: Initialization. *Set the generation counter G = 1; Initialize the population of n fireflies P randomly and each firefly corresponding to a potential solution to the given problem; define light absorption coefficient* γ*; set controlling the step size* α *and the initial attractiveness* β_0 *at r = 0.*

Step 2: Generating rotation coordinate system. *Transform the original coordinate system into new rotation coordinate whose horizontal axis is the connection line from starting point to target point according to* **Equation (1)** *and* **(2)***; convert battlefield threat information to the rotation coordinate system and divide the axis* X' *into D equal partitions. Each feasible solution, denoted by* $\boldsymbol{P}=\{p_1, p_2, \cdots, p_D\}$, *is an array indicated by the composition of D coordinates which are the floating-point numbers*

Step 3: *Evaluate the threat cost/light intensity J for each firefly in P by* **Equation (5)**

Step 4: while *The halting criteria is not satisfied* ***or*** *G < MaxGeneration* ***do***

Sort the population of n fireflies P from best to worst by order of threat cost/light intensity J for each firefly;

Exchange information between top fireflies by **Algorithm 3**

for *i=1:n (all n fireflies)* ***do***

for *j=1:n (n fireflies)* ***do***

if $(J_j < J_i)$,

Move firefly i towards j;

end if

Vary attractiveness with distance r via $\exp[-\gamma r^2]$;

Evaluate new solutions and update threat cost/light intensity;

end for *j*

end for *i*

Evaluate the threat cost/light intensity for each firefly in P by **Equation (8)**.

Sort the population of fireflies P from best to worst by order of threat cost/light intensity J for each firefly;

G = G+1;

Step 5: end while

Step 6: *Inversely transform the coordinates in final optimal path into the original coordinate, and output*

End.

Algorithm 5. Algorithm of MFA for UCAV Path Planning

5. Simulation Experiments

In this section, we look at the performance of MFA as compared with other population-based optimization methods, such as, ACO, BBO, DE, ES, GA, PBIL, PSO and SGA. Firstly, we compare performances between MFA and other population-based optimization methods on the different parameters the maximum generation *Maxgen* and the dimension of converted optimization function D, and then we compare performances between MFA and FA on the different parameters discovery rate p_a and light absorption coefficient γ.

To allow a fair comparison of running times, all the experiments were implemented on a PC with an AMD Athlon(tm) 64 X2 Dual Core Processor 4200+ running at 2.20 GHz, 1024 MB of RAM and a hard drive of 160 Gbytes. Our implementation was compiled using MATLAB R2011b (7.13) running under Windows XP SP3. No commercial BBO tools or other population-based optimization tools were used in the following experiments.

5.1 General Performance of MFA

In this subsection, firstly we will present the supposed problem we use to test the performance of MFA. We use the battle field environment parameter described as [1]. Supposed that there has the following map information, UCAV flight from start point (10, 10) to end point (55, 100). In the flight course, there are five threat areas. Their coordinates and corresponding threat radii are shown in Table 1. Also, we set balanced coefficient between safety performance and fuel performance $k = 0.5$.

Table 1. Information about Known Threats

No.	Location(km)	Threat radius(km)	Threat grade
1	(45,50)	10	2
2	(12,40)	10	10
3	(32,68)	8	1
4	(36,26)	12	2
5	(55,80)	9	3

In order to explore the benefits of MFA, in this subsection we compared its performance on UCAV path planning problem with FA and eight other population-based optimization methods, which are ACO, BBO, DE, ES, GA, PBIL, PSO and SGA. ACO (ant colony optimization) [19, 20] is a swarm intelligence algorithm for solving computational problems which is based on the pheromone deposition of ants. Biogeography-based optimization (BBO) is a new evolutionary algorithm (EA) developed for global optimization which is a generalization of biogeography to EA [21]. DE (differential evolution) [22, 23] is a simple but excellent optimization method that uses the difference between two solutions to probabilistically adapt a third solution. An ES (evolutionary strategy) [24, 25] is an algorithm that generally distributes equal importance to mutation and recombination, and that allows two or more parents to reproduce an offspring. A GA (genetic algorithm) [26] is a search heuristic that mimics the process of natural evolution. PBIL (probability-based incremental learning) [27] is a type of genetic algorithm where the genotype of an entire population (probability vector) is evolved rather than individual members. PSO (particle swarm optimization) [28] is also a swarm intelligence algorithm which is based on the swarm behavior of fish, and bird schooling in nature. A stud genetic algorithm (SGA) [29] is a GA that uses the best individual at each generation for crossover.

We did some rough tuning on each of the optimization algorithms to achieve relatively reasonable performance, but we did not make any special efforts to fine-tune the algorithms. Except an ad hoc explain, in the following experiments, we use the same MATLAB code and parameters settings for other population-based optimization methods as in [21, 30].

To compare the different effects among the parameters *Maxgen* and *D*, we ran 100 Monte Carlo simulations of each algorithm on the above UCAV path planning problem to get representative performances. For simplicity, we subtract 50 from the actual value, *i.e.*, if a value is 0.4419 in the following table, then its corresponding value 50.4419 is its true value. We must point out that we mark the best value with italic and bold font for each algorithm in the following tables.

5.1.1 Effect of Maximum Generation *Maxgen*: The choice of the best maximum generation of meta-heuristic algorithm is always critical for specific problems. Increasing the maximum generation will increase the possibility of reaching optimal solution, promoting the exploitation of the search space. Moreover, the probability to find the correct search direction increases considerably. The influence of maximum generation is investigated in this sub-subsection. For all the population-based optimization methods, all the parameter settings are the same as above mentioned, only except for maximum generation *Maxgen* =50, *Maxgen* = 100, *Maxgen* = 150, *Maxgen* =200 and *Maxgen* =250. The results are recorded in Tables 2, 3, 4 and 5 after 100 Monte Carlo runs. Table 2 shows the best minima found by each algorithm over 100 Monte Carlo runs. Table 3 shows the worst minima found by each algorithm over 100 Monte Carlo runs. Table 4 shows the average minima found by each algorithm, averaged over 100 Monte Carlo runs. Table 5 shows the average CPU time consumed by each algorithm, averaged over 100 Monte Carlo runs. In other words, Table 2, 3 and 4 shows the best, worst, and average performance of each algorithm respectively, while Table 5 shows the average CPU time consumed by each algorithm.

From Table 2, we see that MFA performed the best on all the groups, while FA and DE performed the second best on the 5 groups. Table 3 shows that PBIL was the worst at finding objective function minima on all the five groups when multiple runs are made, while the SGA was the best when *Maxgen* =200 and MFA was the best on the other groups in the worst values. Table 4 shows that MFA was the most effective at finding objective function minima when multiple runs are made, while FA, DE and SGA performed the second best on the 5 groups. Table 5 shows that PBIL was the most effective at finding objective function minima when multiple runs are made, performing the best on all the 5 groups, while FA performed the worst on the 5 groups. By carefully looking at the results in Tables 2, 3 and 4, we can recognize that the values for each algorithm are obviously decreasing with the increasing *Maxgen*, while the performance of MFA increases little with the *Maxgen* increasing from 50 to 250, so, we set *Maxgen* = 200 in other experiments. In sum, from Table 2, 3, 4 and 5 we can draw the conclusion that the more generations, the smaller objective function value we can reach, while the CPU time consumes more. Moreover, MFA performs better than other population-based optimization methods for the UCAV path planning problem with different maximum generation.

Table 2. Best normalized optimization results on UCAV path planning problem on different *Maxgen*. The numbers shown are the best results found after 100 Monte Carlo simulations of each algorithm.

Parameter			Algorithm									
Popsize	Maxgen	D	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA
30	**50**	20	9.6276	4.8658	3.6814	7.7983	1.4713	1.6448	**0.7030**	66.0274	3.6012	1.7041
30	**100**	20	11.5242	4.2616	0.9439	10.2329	0.6577	1.5294	**0.5382**	48.9269	2.3357	1.3520
30	**150**	20	5.6381	5.6105	0.7015	9.8027	0.5459	1.2042	**0.4857**	47.4630	2.6165	0.9498
30	**200**	20	11.2445	2.9424	0.5188	10.7960	0.4931	1.0695	**0.4661**	18.6980	2.3469	0.8392
30	**250**	20	9.7607	3.5209	0.4829	10.2540	0.4753	0.8781	**0.4508**	20.8802	2.9229	0.7839

Table 3. Worst normalized optimization results on UCAV path planning problem on different *Maxgen*. The numbers shown are the worst results found after 100 Monte Carlo simulations of each algorithm.

Parameter			Algorithm									
Popsize	Maxgen	D	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA
30	**50**	20	18.7099	28.6806	31.3392	33.1927	28.0425	10.9773	**4.6726**	**312.9370**	33.1539	17.6318
30	**100**	20	17.7404	28.2427	19.4058	33.1979	29.3022	11.1678	**4.5749**	**373.8334**	27.8806	11.6446
30	**150**	20	17.4223	40.1797	14.5560	35.7277	27.8480	17.5637	**4.9631**	**210.6140**	28.3542	15.2145
30	**200**	20	17.0679	32.1981	16.6736	52.4090	26.5768	11.9124	9.1502	**183.9630**	28.2524	**6.7380**
30	**250**	20	17.0679	27.6544	8.5122	46.0828	26.3005	7.4338	**3.6783**	**169.1446**	29.6341	16.2672

Table 4. Mean normalized optimization results on UCAV path planning problem on different *Maxgen*. The numbers shown are the minimum objective function values found by each algorithm, averaged over 100 Monte Carlo simulations.

Parameter			Algorithm									
Popsize	Maxgen	D	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA
30	**50**	20	16.2648	14.2076	13.2645	20.3152	6.2034	4.1142	**1.9576**	151.9844	9.9671	5.2389
30	**100**	20	16.3500	13.4074	7.3195	20.4387	4.3526	3.7731	**1.3048**	113.6434	8.9057	3.7475
30	**150**	20	16.1722	12.6978	3.5255	19.9485	4.1809	3.4671	**0.9933**	90.8722	8.5509	3.1680
30	**200**	20	16.2154	11.8556	2.3975	20.7501	2.2791	2.9711	**0.8984**	74.1964	8.9892	2.3792
30	**250**	20	16.0444	11.9654	2.4849	20.1739	2.2064	2.6605	**0.7025**	65.4942	9.2143	2.5929

Table 5. Average CPU time on UCAV path planning problem on different *Maxgen*. The numbers shown are the minimum average CPU time consumed by each algorithm.

Parameter			Algorithm									
Popsize	Maxgen	D	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA
30	**50**	20	1.3743	0.8196	1.1511	1.2938	2.8404	1.1813	2.4197	**0.6141**	1.0999	1.1371
30	**100**	20	2.7429	1.5988	2.2950	2.5811	5.6219	2.3296	4.7686	**1.1798**	2.1749	2.2839
30	**150**	20	4.1120	2.3730	3.4833	3.8397	8.3564	3.4659	7.2298	**1.7408**	3.2768	3.4243
30	**200**	20	5.4725	3.1650	4.6924	5.1260	11.2589	4.6889	9.6421	**2.3304**	4.3882	4.5655
30	**250**	20	6.8355	3.9372	5.6820	6.4169	13.8879	5.8137	12.1014	**2.8600**	5.4287	5.6978

5.1.2 Effect of Dimensionality *D*: In order to investigate the influence of the dimension on the performance of MFA, we carry out a scalability study comparing with other population-based optimization methods for the UCAV path planning problem with the dimensionality D=5, D=10, D=15, D=20, D=25, D=30, D=35, D=40. The results are recorded in Tables 6, 7, 8 and 9 after 100 Monte Carlo runs. Table 6 shows the best minima found by each algorithm over 100 Monte Carlo runs. Table 7 shows the worst minima found by each algorithm over 100 Monte Carlo runs. Table 8 shows the average minima found by each algorithm, averaged over 100 Monte Carlo runs. Table 9 shows the average CPU time consumed by each algorithm, averaged over 100 Monte Carlo runs. In other words, Table 6, 7 and 8 shows the best, worst, and average performance of each algorithm respectively, while Table 9 shows the average CPU time consumed by each algorithm. We must point out we use *PS* and *MG* to represent *Popsize* and *Maxgen* for short respectively.

From Table 6, we see that DE performed the best when D = 5, 10 and 15, while MFA performed the best on the other groups when multiple runs are made. Table 7 shows that PBIL was the worst at finding objective function minima on all the eight groups when multiple runs are made, while the SGA was the best when D = 5 and MFA was the best on the other groups in the worst values. Table 8 shows that SGA was the most effective when D = 5 and MFA was the best on the other groups at finding objective function minima when multiple runs are made. Table 9 shows that PBIL was the most effective at finding objective function minima on all the groups. So, from the experimental results of this sub-subsection, we can conclude that the exchange information between top fireflies during the process of the light intensity updating has the ability to accelerate FA in general; especially the improvements are more significant at higher dimensionality. The higher dimension, the better result we get, but it consumes more time; furthermore, the result is good enough when D = 20, while increasing D from 20 to 40, the performance of MFA improves little with consuming more time. In sum, we should get the best D = 20 under the comprehensive consideration. Therefore, we make D = 20 in other experiments.

Table 6. Best normalized optimization results on UCAV path planning problem on different *D*. The numbers shown are the best results found after 100 Monte Carlo simulations of each algorithm.

Parameter			Algorithm									
PS	MG	D	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA
30	200	**5**	11.3724	10.3302	**4.3568**	9.5895	4.3585	5.2471	4.3573	9.7627	5.1667	5.6538
30	200	**10**	10.2281	2.9472	**1.3950**	7.4272	1.3990	1.6068	1.3966	33.1123	2.2073	1.5489
30	200	**15**	8.5298	2.5569	**0.6114**	8.2547	0.6172	0.8711	0.6115	57.2225	2.0969	0.8071
30	200	**20**	10.4451	4.7230	0.5102	10.2329	0.4626	0.8252	**0.4552**	80.1521	2.4643	0.8460
30	200	**25**	11.5490	5.5286	0.5512	13.3685	0.4908	1.2421	**0.4571**	109.7418	3.7378	1.2394
30	200	**30**	13.2299	6.6071	0.8987	15.7251	0.6828	1.9218	**0.5160**	180.1498	3.2993	1.6165
30	200	**35**	16.9599	13.0206	2.5372	16.7445	1.0829	2.3109	**0.4709**	220.3331	5.5025	1.6326
30	200	**40**	19.7946	13.5504	4.5490	18.2314	1.5225	2.2084	**0.4506**	340.6174	5.7367	2.6180

Table 7. Worst normalized optimization results on UCAV path planning problem on different *D*. The numbers shown are the worst results found after 100 Monte Carlo simulations of each algorithm.

Parameter			Algorithm									
PS	MG	D	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA
30	200	**5**	13.3199	121.5724	12.2083	62.2665	15.7395	11.6013	12.4186	**22.2463**	16.0713	**11.2006**
30	200	**10**	18.1911	26.8270	6.7358	73.4605	6.7095	10.1096	**3.7858**	**69.2468**	18.6221	6.1652
30	200	**15**	11.0084	40.3705	12.5808	53.8683	44.2763	7.4472	**3.8319**	**139.2557**	37.3201	11.7962
30	200	**20**	17.1887	28.2063	14.5783	31.4587	28.9142	9.1795	**2.0279**	**287.3709**	28.1596	18.9518
30	200	**25**	12.0733	30.3315	19.6664	33.9148	16.4518	10.3977	**3.7043**	**649.6845**	28.1399	15.6967
30	200	**30**	14.7139	28.5885	24.1279	41.3024	15.9757	12.7183	**8.3364**	**2364.08**	43.6950	14.7140
30	200	**35**	18.7271	43.8512	34.4447	38.7646	33.8871	24.4790	**5.8830**	**6312.96**	32.8328	17.6058
30	200	**40**	27.0641	40.7087	43.2604	46.4224	36.6626	22.0688	**7.7236**	**7053.50**	34.7302	17.8669

Table 8. Mean normalized optimization results on UCAV path planning problem on different *D*. The numbers shown are the minimum objective function values found by each algorithm, averaged over 100 Monte Carlo simulations.

Parameter			Algorithm									
PS	MG	D	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA
30	200	**5**	11.5151	22.7318	8.5962	30.7228	**8.7499**	10.4747	9.1673	16.1391	9.9061	10.5013
30	200	**10**	11.9485	7.9650	3.1045	26.2868	2.1801	2.5422	**1.5740**	51.4355	7.0411	2.2790
30	200	**15**	10.2554	9.5257	2.2783	21.8618	2.8217	2.1880	**0.8967**	78.2477	8.3395	1.8910
30	200	**20**	16.2205	11.8761	2.7221	20.1892	3.7327	3.0900	**0.7004**	135.4365	8.2483	3.1670
30	200	**25**	11.5674	14.7800	4.4081	22.7794	3.9039	3.7814	**0.9987**	207.7272	10.2627	4.1567
30	200	**30**	13.9593	17.8746	9.9884	24.7757	4.9621	5.0079	**1.3568**	345.5447	12.3847	4.5211
30	200	**35**	18.3108	21.5615	17.9027	26.5217	5.9955	5.9599	**1.6009**	634.6550	14.1354	5.8260
30	200	**40**	24.5754	24.8531	27.6201	30.2595	7.8558	7.4927	**2.1978**	1119.72	14.8845	7.1100

Table 9. Average CPU time on UCAV path planning problem on different *D*. The numbers shown are the minimum average CPU time consumed by each algorithm.

Parameter			Algorithm									
PS	MG	D	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA
30	200	**5**	2.33	1.48	2.43	2.59	8.32	2.62	5.06	**1.42**	2.77	2.57
30	200	**10**	3.36	2.07	3.43	3.30	9.33	3.36	6.61	**1.74**	3.32	3.27
30	200	**15**	4.50	2.81	3.84	4.27	10.31	4.03	8.20	**2.10**	3.87	3.93
30	200	**20**	5.39	3.02	4.54	5.04	11.08	4.52	9.59	**2.26**	4.26	4.33
30	200	**25**	6.72	3.57	5.30	5.88	12.01	5.23	11.29	**2.65**	4.84	5.05
30	200	**30**	7.75	4.22	5.91	6.83	12.94	5.73	12.79	**2.92**	5.30	5.56
30	200	**35**	8.75	4.72	6.48	7.65	13.80	6.27	14.44	**3.26**	5.81	6.04
30	200	**40**	9.90	5.21	6.94	8.47	14.62	6.56	16.05	**3.56**	6.22	6.49

5.2 Influence of Control Parameter

In [18], X.S. Yang concluded that if we adjust the parameter $\boldsymbol{\gamma}$ properly so that FA can outperform both the random search and PSO. The choice of the control parameters $\boldsymbol{\gamma}$ is of vital importance for different problems. To compare the different effects among the parameters $\boldsymbol{\gamma}$ and p_a (for MFA only), we ran 100 Monte Carlo simulations of FA and MFA algorithm on the above problem to get representative performances.

5.2.1 Light Absorption Coefficient $\boldsymbol{\gamma}$: To investigate the influence of the light absorption coefficient on the performance of MFA, we carry out this experiment comparing with FA for the UCAV path planning problem with the Light absorption coefficient $\boldsymbol{\gamma}$ =0, 0.2, 0.4, ···, 1.8, 2.0 and fixed discovery p_a = 0.25. All other parameter settings are kept unchanged. The results are recorded in Tables 10, 11, 12 and 13 after 100 Monte Carlo runs. Table 10 shows the best minima found by FA and MFA algorithm over 100 Monte Carlo runs. Table 11 shows the worst minima found by FA and MFA algorithm over 100 Monte Carlo runs. Table 12 shows the average minima found by FA and MFA algorithm, averaged over 100 Monte Carlo runs. Table 13 shows the average CPU time consumed by FA and MFA algorithm, averaged over 100 Monte Carlo runs. In other words, Table 10, 11 and 12 shows the best, worst, and average performance of FA and MFA algorithm respectively, while Table 13 shows the average CPU time consumed by FA and MFA algorithm.

From Table 10, we see that MFA performed the best (on average) on all the groups. Table 11 shows that FA was the worst at finding objective function minima when multiple runs are made. Table 12 shows that MFA performed the best on average on all the groups and MFA reaches minima when $\boldsymbol{\gamma}$ = 1.0. Table 13 shows that MFA was more effective at finding objective function minima when multiple runs are made, performing the best on all the groups. By carefully looking at the results in Table 10, 11 and 12, we can recognize that the value for FA and MFA is varies little with the increasing $\boldsymbol{\gamma}$, and MFA reaches mean minimum 0.6818 on $\boldsymbol{\gamma}$ = 1.0. So, we set $\boldsymbol{\gamma}$ = 1.0 in other experiments. In sum, from Table 10, 11, 12 and 13, we can conclude that the exchange information between top fireflies during the process of the light intensity updating has the ability to accelerate FA in general.

Table 10. Best normalized optimization results on UCAV path planning problem on different γ. The numbers shown are the best results found after 100 Monte Carlo simulations of FA and MFA algorithm.

Parameter		Algorithm	
γ	p_a	FA	MFA
0	0.2	0.6817	**0.4744**
0.2	0.2	0.5233	**0.4588**
0.4	0.2	0.4838	**0.4648**
0.6	0.2	0.4756	**0.4645**
0.8	0.2	0.4826	**0.4701**
1.0	0.2	0.4727	**0.4517**
1.2	0.2	0.4614	**0.4601**
1.4	0.2	0.4525	**0.4493**
1.6	0.2	0.4715	**0.4552**
1.8	0.2	0.4737	**0.4589**
2.0	0.2	0.4771	**0.4539**
	0.2500		

Table 11. Worst normalized optimization results on UCAV path planning problem on different γ. The numbers shown are the worst results found after 100 Monte Carlo simulations of FA and MFA algorithm.

Parameter		Algorithm	
γ	p_a	FA	MFA
0	0.2	**24.2596**	5.4853
0.2	0.2	**26.3834**	4.4896
0.4	0.2	**25.9909**	4.5284
0.6	0.2	**16.9588**	4.2748
0.8	0.2	**22.6322**	3.6463
1.0	0.2	**26.6847**	2.9929
1.2	0.2	**26.3925**	3.7586
1.4	0.2	**26.2005**	6.2209
1.6	0.2	**27.3431**	3.4639
1.8	0.2	**26.6228**	4.8045
2.0	0.2	**26.6203**	3.5995

Table 12. Mean normalized optimization results on UCAV path planning problem on different γ. The numbers shown are the minimum objective function values found by FA and MFA algorithm, averaged over 100 Monte Carlo simulations.

Parameter		Algorithm	
γ	p_a	FA	MFA
0	0.2	4.4532	**1.0837**
0.2	0.2	3.6396	**0.8336**
0.4	0.2	2.2297	**0.7467**
0.6	0.2	2.3610	**0.7675**
0.8	0.2	3.2530	**0.7176**
1.0	0.2	3.6474	**0.6818**
1.2	0.2	2.6510	**0.6987**
1.4	0.2	2.5297	**0.7612**
1.6	0.2	3.0948	**0.6967**
1.8	0.2	2.4524	**0.8540**
2.0	0.2	3.0974	**0.7265**

Table 13. Average CPU time on UCAV path planning problem on different γ. The numbers shown are the minimum average CPU time consumed by FA and MFA algorithm.

Parameter		Algorithm	
γ	p_a	FA	MFA
0	0.2	11.2259	**9.5890**
0.2	0.2	11.0729	**9.5361**
0.4	0.2	11.1534	**9.5805**
0.6	0.2	11.1742	**9.5684**
0.8	0.2	11.1190	**9.5618**
1.0	0.2	11.1051	**9.5732**
1.2	0.2	11.0948	**9.5903**
1.4	0.2	11.1523	**9.5752**
1.6	0.2	11.2221	**9.5672**
1.8	0.2	11.1196	**9.5191**
2.0	0.2	11.0665	**9.5496**

5.2.2 Discovery Rate p_a: For the sake of investigating the influence of the discovery rate on the performance of MFA, we carry out this experiment for the UCAV path planning problem with the crossover constant p_a =0, 0.1, 0.2, ···, 1.0 and fixed light absorption coefficient: $\boldsymbol{\gamma}$ = 1.0. All other parameter settings are kept unchanged. The results are recorded in Tables 14 after 100 Monte Carlo runs. Column 1, 2 and 3 in Table 14 shows the best, worst, and average performance of MFA algorithm respectively, while Column 4 in Table 18 shows the average CPU time consumed by MFA algorithm.

From Table 14, we can recognize that the values for MFA varies little with the increasing p_a, and MFA reaches best minimum on p_a = 0.2. So, we set p_a = 0.2 in other experiments. From Tables 14 we can draw the conclusion that MFA is insensitive to the discovery rate p_a, so we do not have to fine-tune the parameter p_a to get the best performance for different problems.

Table 14. Best normalized optimization results and average CPU time on UCAV path planning problem on different p_a. The numbers shown are the best results found after 100 Monte Carlo simulations of MFA algorithm.

Parameter		Algorithm			
		MFA			
γ	p_a	Best	Worst	Mean	CPU Time
1.0	**0**	0.7787	**3.9986**	**1.7971**	2.6011
1.0	**0.1**	0.7905	5.5656	1.9514	2.4983
1.0	**0.2**	**0.6833**	**11.2584**	2.1452	2.4210
1.0	**0.3**	0.8212	5.6645	2.3072	2.3579
1.0	**0.4**	0.6473	15.4946	2.4372	2.2730
1.0	**0.5**	0.8431	7.4277	2.5482	2.1992
1.0	**0.6**	0.7353	8.4473	2.5198	2.1332
1.0	**0.7**	0.8071	17.5002	3.4084	2.0382
1.0	**0.8**	0.7630	13.3585	3.1679	1.9717
1.0	**0.9**	0.8367	13.2816	3.9205	1.8724
1.0	**1.0**	**0.8943**	17.2353	**6.3522**	1.7538

The simulation experiment implemented in Subsection 6.1 and Subsection 6.2 shows that the algorithm MFA we proposed performed the best but worst effectively when solving the UCAV path planning problem. From deep investigation, we can see that MFA cam reach minima when maximum generation *Maxgen* =50 and population size *Popsize* = 30, while other population-based optimization methods cannot achieve satisfactory result under this condition, *i.e.*, MFA needs fewer maximum generation, less population size, less time than other population-based optimization methods when arriving to the same performance. In sum, the simulation implemented in Section 6 shows that the algorithm MFA we proposed performed the best and most absolutely effectively, and it can solve the UCAV path planning problem perfectly. Furthermore, comparing to other population-based optimization methods, the algorithm MFA is insensitive to the parameter light absorption coefficient $\boldsymbol{\gamma}$ and discovery rate p_a, so we do not have to fine-tune the parameter $\boldsymbol{\gamma}$ and p_a to get the best performance for different problems.

5.3 Discussions

The FA algorithm is a simple, fast, and robust global optimization algorithm developed by X. S Yang in 2008. However, it may lack the exchange information between top fireflies, because different fireflies work almost independently. Therefore, in this work, we add exchange information between top fireflies to the FA during the process of the light intensity updating. And then, the MFA algorithm is proposed based on exchange information between top fireflies to solve the UCAV path planning. From the experimental results we can sum up that

• Our proposed MFA approach is effective and efficient. It can solve the UCAV path planning problem effectively.

• The overall performance of MFA is superior to or highly competitive with FA and other compared state-of-the-art population-based optimization methods.

• MFA and other population-based optimization methods were compared for different maximum generations and the dimension. Under majority conditions, MFA is significantly substantial better than other population-based optimization methods.

• MFA and FA were compared for different absorption coefficient $\boldsymbol{\gamma}$ and discovery rate p_a. Under almost all the conditions, MFA is far better than FA.

• The algorithm MFA is insensitive to the parameter light absorption coefficient $\boldsymbol{\gamma}$ and discovery rate p_a, so we do not have to fine-tune the parameter $\boldsymbol{\gamma}$ and p_a to get the best performance for different problems.

6. Conclusion and Future Work

This paper presented a modified firefly algorithm for UCAV path planning in complicated combat field environments. A novel type of FA model has been described for single UCAV path planning, and a modification is applied to exchange information between top fireflies during the process of the light intensity updating. Then, the UCAV can find the safe path by connecting the chosen nodes while avoiding the threat areas and costing minimum fuel. This new approach can accelerate the global convergence speed while maintaining the strong robustness of the basic FA. The detailed implementation procedure for this improved meta-heuristic approach is also described. Compared with other population-based optimization methods, the simulation experiments show that this improved method is a feasible and effective way in UCAV path planning. It is also flexible, in complicated dynamic battle field environments and pop-up threats are easily incorporated.

In the algorithm of UCAV path planning, there are many issues worthy of further study, and efficient route planning method should be developed depending on the analysis of specific combat field environments. Currently, the hot issue contains self-adaptive route planning for a single UCAV and collaborative route planning for a fleet of UCAVs. As the important ways of improving aircraft survivability, adaptive route planning should analyze real-time data under the uncertain and dynamic threat condition, even it can re-modify pre-planned flight path to improve the success rate of completing mission. The difficulty of the collaborative route planning for a fleet of UCAVs exists in coordination between the various UCAVs, including the fleet formation, target distribution, arrival time constraint and avoidance conflict, each of which is a complicated question worthy of further study. Our future work will focus on the two hot issues and develop new methods to solve problem in UCAV path planning and re-planning.

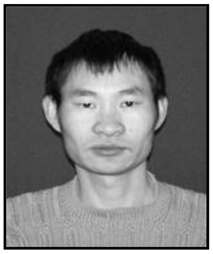
Acknowledgements

This work was supported by State Key Laboratory of Laser Interaction with Material Research Fund under Grant No. SKLLIM0902-01 and Key Research Technology of Electric-discharge Non-chain Pulsed DF Laser under Grant No. LXJJ-11-Q80. The authors are grateful to Prof. Zhao Zhang, and the anonymous reviews for their valuable comments on the submitted version of this manuscript.

References

[1] H. B. Duan, X. Y. Zhang and C. F. Xu, "Bio-Inspired Computing", Science Press, Beijing, **(2011)**.
[2] Y. V. Pehlivanoglu, "A new vibrational genetic algorithm enhanced with a Voronoi diagram for path planning of autonomous UAV", Aerosp. Sci. Technol., vol. 16, **(2012)**, pp. 47–55.
[3] W. Ye, D. W. Ma and H. D. Fan, "Algorithm for low altitude penetration aircraft path planning with improved ant colony algorithm", Chinese J. Aeronaut., vol. 18, **(2005)**, pp. 304–309.
[4] H. B. Duan, Y. X. Yu, X. Y. Zhang and S. Shao, "Three-dimension path planning for UCAV using hybrid meta-heuristic ACO-DE algorithm", Simul. Model. Pract. Th., vol. 18, **(2010)**, pp. 1104–1115.
[5] H. B. Duan, X. Y. Zhang, J. Wu and G. J. Ma, "Max-min adaptive ant colony optimization approach to multi-UAVs coordinated trajectory replanning in dynamic and uncertain environments", J. Bionic. Eng., vol. 6, **(2009)**, pp. 161–173.
[6] C. F. Xu, H. B. Duan and F. Liu, "Chaotic artificial bee colony approach to uninhabited combat air vehicle (UCAV) path planning", Aerosp. Sci. Technol., vol. 14, **(2010)**, pp. 535–541.
[7] H. B. Duan, S. Q. Liu and J. Wu, "Novel intelligent water drops optimization approach to single UCAV smooth trajectory planning", Aerosp. Sci. Technol., vol. 13, **(2009)**, pp. 442–449.
[8] X. S. Yang, "Nature-Inspired Metaheuristic Algorithms", Luniver Press, Frome, **(2008)**.
[9] X. S. Yang, "Firefly algorithms for multimodal optimization, stochastic algorithms: foundations and applications", SAGA 2009, LNCS 5792, **(2009)**, pp. 169–178.
[10] S. Lukasik and S. Zak, "Firefly algorithm for continuous constrained optimization task", ICCCI 2009, LNAI 5796, **(2009)**, pp. 97-100.
[11] A. Chatterjee and G. K. Mahanti, "Design of a fully digital controlled reconfigurable switched beam conconcentric ring array antenna using firefly and particle swarm optimization algorithm", Prog. Electromagnet. Res. B, vol. 36, **(2012)**, pp. 113-131.
[12] X. S. Yang, S. S. Hosseini and A. H. Gandomi, "Firefly algorithm for solving non-convex economic dispatch problems with valve loading effect", Appl. Soft Comput., vol. 12, **(2012)**, pp. 1180-1186.
[13] S. M. Farahani, A. A. Abshouri, B. Nasiri and M. R. Meybodi, "Some hybrid models to improve firefly algorithm performance", Int. J. Artif. In., vol. 8, **(2012)**, pp. 97-117.
[14] W. Ye and H. D. Fan, "Research on mission planning system key techniques of UCAV", J. Nav. Aeronaut. Eng. I., vol. 22, **(2007)**, pp. 201-207.
[15] J. L. Yan, "Evolutionary algorithm based route planer for Unmanned Air Vehicles", A Thesis in Control Science and Engineering, Nanjing University of Aeronautics and Astronautics, **(2008)**.
[16] K. M. Passino, "Biomimicry of Bacterial Foraging for Distributed Optimization", Princeton University Press, Princeton, **(2001)**.
[17] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization", in: Proceedings of the 2007 IEEE Swarm Intelligence Symposium, Honolulu, HI, **(2007)**, pp. 120-127.
[18] X. S. Yang, "Nature-Inspired Metaheuristic Algorithms", second ed., Luniver Press, Frome, **(2010)**.
[19] M. Dorigo, "Optimization, learning and natural algorithms", PhD thesis, Politecnico di Milano, Italy, **(1992)**.
[20] M. Dorigo and T. Stutzle, "Ant Colony Optimization", MIT Press, Cambridge, **(2004)**.
[21] D. Simon, "Biogeography-based optimization", IEEE T. Evolut. Comput., vol. 12, **(2008)**, pp. 702–713.
[22] K. Price and R. Storn, "Differential evolution", Dr. Dobb's J., vol. 22, **(1997)**, pp. 18-20.
[23] X. T. Li and M. H. Yin, "Design of a reconfigurable antenna array with discrete phase shifters using differential evolution algorithm", Prog. Electrom. Res. B, vol. 31, **(2011)**, pp. 29-43.
[24] T. Back, "Evolutionary Algorithms in Theory and Practice", Oxford University Press, Oxford, **(1996)**.
[25] H. Beyer, "The Theory of Evolution Strategies", Springer, New York, **(2001)**.
[26] D. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning, Reading", Addison-Wesley, MA, **(1989)**.
[27] B. Shumeet, "Population-based incremental learning: a method for integrating genetic search based function optimization and competitive learning", Technical Report, Pittsburgh, PA, Carnegie Mellon University, **(1994)**.
[28] J. Kennedy and R. C. Eberhart, "Particle swarm optimization", in: Proceeding of the IEEE International Conference on Neural Networks, Perth, Australia, **(1995)**, pp. 1942-1948.
[29] W. Khatib and P. Fleming, "The stud GA: a mini revolution?", in: A. Eiben, T. Back, M. Schoenauer, H. Schwefel(Eds.), Parallel Problem Solving from Nature, Springer, New York, **(1998)**, pp. 683-691.
[30] D. Simon, "The Matlab code of biogeography-based optimization", http://academic.csuohio.edu/simond/bbo/, **(2012)** February 14.

Authors



Gaige Wang obtained his bachelor degree in computer science and technology from Yili Normal University, Yining, Xinjiang, China, in 2007. His masters was in the field of "Intelligent planning and planning recognition" at Northeast Normal University, Changchun, China. In 2010 he began working on his Ph.D. for developing target threat evaluation by employing computational intelligence techniques at Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun, Jilin, China.

Dr. Lihong Guo received her PhD from Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, China. Her areas of interest include computer vision, and computer application. She is recognized Supervisor to guide the PhD scholars in Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, China.