International Journal of Hybrid Information Technology
Vol.7, No.1 (2014), pp.183-200
http://dx.doi.org/10.14257/ijhit.2014.7.1.15

A Wolf Colony Search Algorithm Based on the Complex Method for Uninhabited Combat Air Vehicle Path Planning

Qiang Zhou[1], Yongquan Zhou[1,2] and Xin Chen[1]

[1]*College of Information Science and Engineering,*
Guangxi University for Nationalities, Nanning Guangxi 530006, China
[2]*Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis,*
Nanning Guangxi 530006, China

yongquanzhou@126.com

Abstract

Path planning for uninhabited combat air vehicle (UCAV) is a class of complicated high dimensional optimization problem, which mainly centralizes on path planning considering the different kinds of constrains in the complex environment of war. In order to solve this problem, it is converted to a kind of constrained function optimization problem, and a wolf colony search algorithm based on the complex method is proposed, which combines the complex method with a wolf colony search algorithm, and it solves the problem of UCAV path planning successfully. The experiment results show that our proposed algorithm is feasible and effective to solve the problem of UCAV path planning.

Keywords: *Wolf colony search algorithm; complex method; UCAV path planning; constrained optimization*

1. Introduction

Path planning is a new generation of low altitude penetration technology to achieve the purpose of terrain-following, terrain avoidance and flight with evading threat. While the path planning for UCAV is an important part in the mission planning system. The goal for path planning is to calculate the optimal or sub-optimal flight route for UCAV within the appropriate time so that the UCAV can break through the enemy threat environments and ensure the mission to conduct smoothly. UCAV path planning problem is a kind optimization problem which is related to the national defense and security, so a series of algorithms have been proposed to solve this complicated multi-constrained optimization problem, such as differential evolution (EA) [1, 13], genetic algorithm (GA) [2], ant colony optimization algorithm (ACO) [3], particle swarm optimization (PSO) [4] and artificial bee colony (ABC) [19] and so on. However, these algorithms can hardly solve the contradiction between the global optimization and excessive information.

The wolf colony algorithm (WCA) is proposed by C. G Yang *et al.*, in 2007. The algorithm is a swarm intelligence algorithm to simulate the intelligent predatory behaviors of the wolf colony. The wolf is a very intelligent animal. They are not alone when they catch and feed on food but by teams composed of several wolves. The wolf colony sends a few wolves to search quarry by smell. When the searching wolves discover the quarry, they notify the position of the quarry to the other wolves by howl.

ISSN: 1738-9968 IJHIT

The other wolves get close to the quarry and besiege it. After they get the quarry, they distribute the food according to the strength of the wolf. At last, the weak wolves will be eliminated. WCA is proposed by this predatory behaviors of the wolf colony mixed with ABC successfully and applied in the path planning for the mobile robot. WCA has a good convergence rate, but solution accuracy is not high and easy to fall into local optimization.

In this paper, on the basis of the basic wolves' algorithm, we introduce a complex method strategy, and propose a wolf colony search algorithm based on the complex method (CWCA), and apply it in UCAV path planning problem successfully. Finally comparative experiments are conducted with ACO, BBO [15, 18], DE, ES [16], GA [8-9], PBIL [14], PSO, SGA [12], FA [5] and MFA [6]. Experiment simulation results show that CWCA is more effective to solve UCAV path planning problem than other algorithms.

2. UCAV Path Planning Mathematical Model

2.1 Problem description [20]

Path planning for UCAV is formulated according to the practical situation and marks out the optimal flight route meeting certain performance requirements according to some performance indicators, and needs to consider many factors, such as the terrain, data, threat information, and time and so on. In order to solve UCAV path planning problem, this paper builds a function optimization problem, creates a mathematical model according to constraints, and finds the optimal flight routes satisfying the requirements. Shown as Figure 1, the actual problem is transformed into a D-dimensional function optimization problem by converting coordinates.

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

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

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

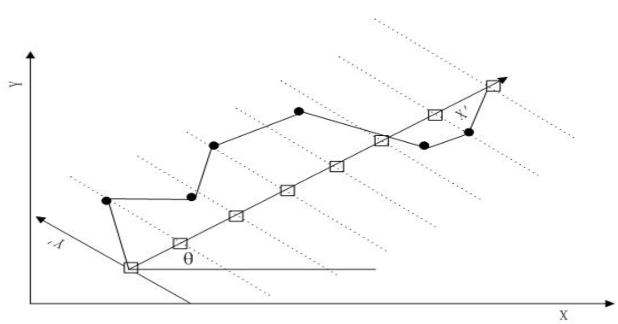

Figure 1. Coordinates Transformation Relation

After the above conversion, we divide the abscissa of new coordinate $O_{x'y'}$, into D equal partitions, and then get a vertical line and the coordinate of ordinate Y' for the corresponding node, so we can obtain a two-dimensional coordinate points set formed by D points. The abscissas of these points are divided into equal partitions, so it is easy to get their point coordinate. We can get a path from start point to end point through connecting these points together, so UCAV problem is transformed into a constrained function optimization problem to let the problem become simplistic.

2.2 Performance indicator

In the problem description, UCAV needs to consider many factors to complete the task; these factors are performance indicators of the problem including safety performance indicator and performance indicator. UCAV needs to avoid some threats and pitfalls to make UCAV's threat minimum, and so does the fuel cost. We call it threat cost.

Minimum of performance indicator for threat is calculated by Equation (3).

$$\min J_t = \int_0^L \omega_t dl \tag{3}$$

Minimum of performance indicator for fuel is calculated by Equation (4).

$$\min J_f = \int_0^L \omega_f dl \tag{4}$$

Then the total performance indicators for UCAV route are calculated by Equation (5).

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

where J_t, J_f, J are the performance indicator of threat, fuel and the total performance indicators for UCAV route respectively; ω_t is the threat cost of each point on the route; ω_f is fuel cost of each point on the path; L is the path length; $k, k \in [0,1]$ is balanced coefficient between safety performance and fuel performance, whose value is determined by the actual situation of the task UCAV performing, and depending on whether the aircraft is the emphasis on safety or on the time of task completion. The part is paid more attention; the share of the coefficient values will be greater.

2.3 Threat cost

When the UCAV is performing tasks, flying along the path $L_{i,j}$, the total threat cost generated by N_t threats is calculated by Equation (6).

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

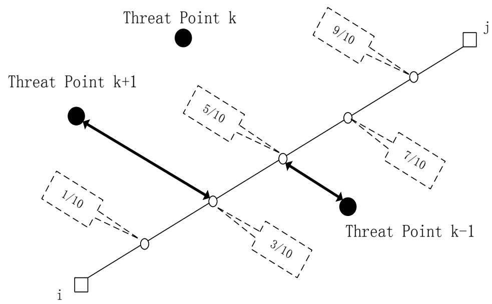

Figure 2. Calculation for Threat Cost

To simplify the calculations, we fetch a number of points of each segment to calculate threat cost. Shown in Figure 2, each edge is divided into five equal partitions, and the threat cost on this edge is calculated by the five points. If the distance from the threat point to the edge is less than the threat radius, we can calculate the responding threat cost according to Equation (7).

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

where $L_{i,j}$ 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 $L_{i,j}$ to the k-th threat; t_k is threat level of the k-th threat.

2.4 Fuel cost

In the practical problem of UCAV path planning, fuel cost depends on path length. And the fuel cost of each point in path ω_f identically equal to 1, so $\omega_f = L$, for simplicity, and fuel cost of each edge can be expressed by $\omega_{f,L_{ij}} = L_{i,j}$.

3. CWCA

In 1965,Box used simplex method [11] for solving unconstrained optimization problems to solve the constrained optimization problems, and the formed complex method for solving unconstrained optimization problems. In order to increase the optimization capability of WCA, the most strong wolves were selected to build a complex, the complex centroid was made use of to produce a new point by reflecting, extension and shrinking, and the new point replaced bad points to continuously close to the optimal point. This algorithm is called CWCA.

3.1 Initialize the wolf colony

At this phase, the purpose is to let every wolf uniformly distribute in the domain of definition of the objective function. The scale of the wolf colony is N, the search space dimension is D, the position of the i-th wolf is

$$X_i = (x_{i1},...,x_{id},...,x_{iD}), \quad (1 \le i \le N, 1 \le d \le D)$$
$$x_{id} = x_{\min} + rand * (x_{\max} - x_{\min}) \tag{8}$$

where $rand$ is a random number uniformly distributed in the range $[0,1]$, $x_{\max}$ and $x_{\min}$ are upper and lower limits of search space.

3.2 Search the quarry

In order to search the quarry, the optimal (that the fitness value is best) q wolves are selected as the searching wolf, q wolves are searching in h directions around themselves, the position of wolf is $P_i(p_{i1},..., p_{id},..., p_{iD})$. If the current position of the searching wolf is P_0, P_1 is produced around the current position P_0.If P_1 is better than P_0, then P_1 is set as the current position, and continue to search. When the searching number is larger than $\max dh$ or the current position is better than the searching position, the searching behavior ends.

In the h positions produced around the searching wolf, $y_{jd}(1 \le j \le h)$ is the position of j-th point of the d-th dimension, it can be calculated as follows:

$$y_{jd} = xx_{id} + rand * stepa \tag{9}$$

Where, $rand$ is a random number uniformly distributed in the region $[-1,1]$, xx_{id} is the position of the i-th searching wolf of the d-th dimension. $stepa$ is the searching step.

3.3 Besiege the quarry

Suppose that the position of the quarry is the position of the searching wolf. When the searching wolves discover the quarry, they notify the position of the quarry to the other wolves by how. The other wolves get close to the quarry and besiege it. Then the updated position z_{id} of the i-th wolf of the d-th dimension is:

$$z_{id} = x_{id} + rand * stepb * (x_{ld} - x_{id}) \tag{10}$$

Where x_{id} is the current position of the i-th wolf of the d-th dimension, $rand$ is a random number uniformly distributed in the region $[0,1]$, $stepb$ is movement step, x_{id} is the position of the d-th dimension of the quarry.

3.4 Lead the wolf colony to search the quarry based on complex method

In the wolf colony, we select u strong wolves to build a complex, suppose the position of the u-th wolf is sorted as $(Z_1, Z_2,..., Z_u)$ by fitness values. The complex steps are as follows**:**

Step1: calculate the complex centroid Z_c

$$Z_c = \frac{1}{u-1}\sum_{i=1}^{u-1} Z_i \tag{11}$$

Step2: calculate the reflection point Z_r

$$Z_r = Z_c + \alpha(Z_c - Z_u) \tag{12}$$

where α is reflection coefficient, generally taken $\alpha = 1.3$. If Z_r is better than Z_u, then Z_r will replace Z_u; otherwise, turn to **Step4.**

Step3: extended operation

$$Z_e = Z_r + \beta(Z_r - Z_c) \tag{13}$$

Where β is extension coefficient, taken $\beta = 0.5 \sim 0.8$. If Z_e is better than Z_u, then Z_e will replace Z_u; otherwise, turn to **Step4.**

Step4: shrinked operation

$$Z_s = Z_u + \chi(Z_c - Z_u) \tag{14}$$

Where χ is shrinkage coefficient, taken $\chi = 0.7$. If Z_s is better than Z_u, then Z_s will replace Z_u; otherwise, turn to **Step1.**

A complex is constructed by selecting the strong wolves and conducts the wolf colony to search quarry so that the optimization capability of the algorithm is improved and uneasy into local optimization.

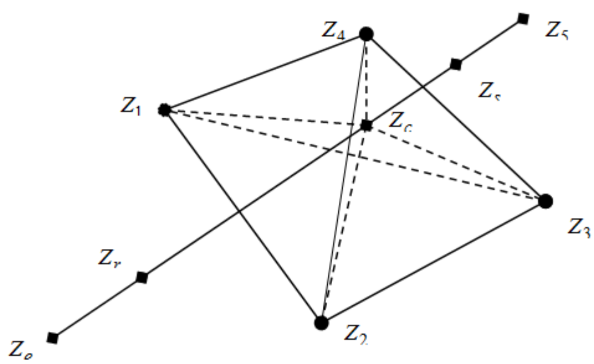

Figure 3. Complex Method

Figure 3 is an actual model of complex, the model is built by Z_1, Z_2, Z_3, Z_4, and the centroid Z_c is found by the four points, and then Z_c maps out reflection point Z_r and shrinkage point Z_s of Z_5. And the Z_r that meets the conditions can get its extension point Z_e. This paper applies the strategy to WCA to search the optimal solution. Every point is the position of the wolf in the strategy.

3.5 Update the wolf colony

The assignment rule of the wolf colony is to assign the food to the strong wolf at first and then to the weak one. The rule makes that the strong wolf gets enough food while the weak one gets little so that the weak wolf will starve to death. But the rule can ensure that the strong wolves prey next time, so the adapt ability of the wolf colony can be enhanced. By simulating the principle of survival of the fittest, the paper removes the worst m artificial wolves in the colony and generates m wolves randomly. Therefore, the algorithm can avoid the local optimum and the wolf colony becomes various.

3.6 The pseudo code of CWCA

Five iterations are processed to build complex in the experiment, in order to explain the process of CWCA, the pseudo code of CWCA is given as follows:

BEGIN
Initialize: initialize the wolf colony randomly;
While (termination condition is not satisfied)
Select q searching wolves by calculating fitness value $f(x)$;
The position of the strongest wolf is as the position of the quarry, other wolves besiege it;
Select u better wolves after the colony sorted;
$\mathrm{Number}=0$;
While ($\mathrm{Number}<5$)
The complex is built by the position of u wolves, the centroid of complex are calculated;
$x_c=\frac{1}{u-1}\sum_{i=1}^{u-1}x_i$;
Calculate reflection point, $x_r=x_c+\alpha(x_c-x_u)$;
If $f(x_r)$ is better than $f(x_u)$
$x_u=x_r$;
Extended operation, $x_e=x_r+\beta(x_r-x_c)$;
If $f(x_e)$ is better than $f(x_u)$
$x_u=x_e$
Else
Shrinked operation, $x_s=x_u+\chi(x_c-x_u)$
If $f(x_s)$ is better than $f(x_u)$
$x_u=x_s$
End If
End If
Else
Shrinked operation, $x_s=x_u+\chi(x_c-x_u)$;
If $f(x_s)$ is better than $f(x_u)$
$x_u=x_s$
End If
End If
End While
Assign the food and update the wolf colony, remove the worst m wolves and generate m wolves randomly;
End While
Output the optimal individuals and fitness value of the populations;
End

4. CWCA for Solving UCAV Path Planning

CWCA can adapt to the demand for UCAV path planning, and has a good search capability and improves the accuracy of the solution. Then we introduce the process of CWCA to apply in UCAV path planning.

4.1 The steps of UCAV path planning based on CWCA

Step1: Initialization. Initial the wolves' population size n, maximum number of iterations $\max t$, searching step $stepa$, besieging step $stepb$ and other parameters.

Step2: Transform coordinate system. A new coordinate is built in the original coordinate system. The abscissa is divided into equal D partitions, and every aliquot ordinate is become the position of every dimension of wolves so the position of the i-th wolf is $(x_{i1,}x_{i2},...,x_{iD})$.

Step3: Calculate the fitness value. Threat cost of every wolf is calculated by equation (5).

Step4: Searching the quarry. q wolves are selected, namely the threat cost is optimal, searching the quarry according to equation (9).

Step5: Besiege the quarry, the position of the wolf colony is updated by equation (10).

Step6: Build a complex. Select u wolves which have better threat cost to build a complex by 3.4.

S**tep7:** Update the wolf colony. The paper removes the worst m wolves in the colony and generates m wolves randomly.

Step8: After completing an iteration, the algorithm enters the next iteration, and determines whether it meets the conditions or not. If it meets the conditions, it exits the loop, and records the position of wolves and indexes; otherwise go to **Step2**.

Step9: Inversely transform the coordinates. Transform the new coordinates into the original coordinates, and record results.

4.2 The flow chart of UCAV path planning based on CWCA

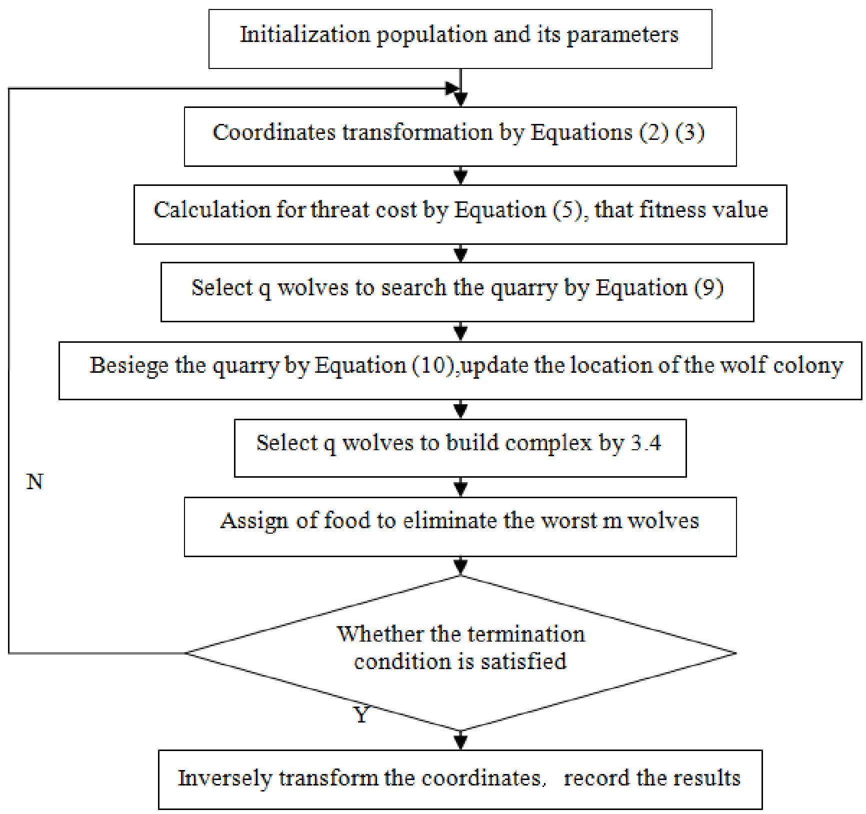

Figure 4. The Flow Chart of CWCA Solving UCAV Path Planning Problem

5. Experiment Simulations

This work use the emulator which is written by MATLAB 2010b and it is performed on the PC with Windows 7 OS, AMD athlon 640 Quad-core processors and 3GB memory.

In [10] an example is simulated. UCAV flight starts at the point $(10,10)$ and end at point $(55,100)$. There are five threats in this example, and they have their own threat grade and threat radius. Among, we set balanced coefficient between safety performance and fuel performance $k = 0.5$. Information about known threats is shown Table 1.

Table 1. Information about known threats

Location	[45,50]	[12,40]	[32,68]	[36,26]	[58,80]
Threat radius	10	10	8	12	9
Threat grade	2	10	1	2	3

5.1 Influence of control parameters

The choice of the control parameters has a great impact on different problems, the choice of besieging step $stepb$, extension coefficient β, and searching step $stepa$ have different influences on route planning.

Order $D = 20,$, iterations $\max t = 200,$, population size $n = 30$, the number of searching wolves $q = 5$, searching direction $h = 4$, the maximum number of searching $\max dh = 15$, searching step $stepa = 0.5$, the number of worst wolves $m = 5$. Each algorithm runs 100 times independently, best and means normalized optimization results on UCAV path planning problem are shown in Table 2. In order to represent the total performance value conveniently in the table, the threat cost all subtract 50 from the actual value, i.e., if a value is 0.5762 in the following table, then its corresponding value 50.5762 is its true value.

From the Table 2, when besieging step is $stepb = 0.1$. Regardless of extension coefficient β changing, the result exist only a little difference, here set $\beta = 0.8$, because its mean value is the smallest.

Table 2. Best and mean normalized optimization results on UCAV path planning problem on different $stepb$ and β

$Stepb$	β	Optimal	Mean
0.1	0.5	0.4191	0.5136
0.3	0.5	0.4594	0.6241
0.5	0.5	0.4467	0.7402
0.1	0.65	**0.4151**	0.5044
0.3	0.65	0.4558	0.6297
0.5	0.65	0.4666	0.8785
0.1	0.8	0.4173	**0.5038**
0.3	0.8	0.4240	0.5788
0.5	0.8	0.4367	0.6328

Before besieging the quarry, the wolf colony searches the quarry, so the convergence speed of the algorithm is improved. But if the convergence speed is too fast, the algorithm is easy to fall into local optimization. So we simulate with different searching step $stepa$ to find out the appropriate search step by comparing the total performance. From Table 2, we can know that the appropriate value for CWCA is $stepb = 0.1, \beta = 0.8$. Best and mean normalized optimization results on UCAV path planning problem on different searching step are shown in Table 3.

Table 3. Best and mean normalized optimization results on UCAV path planning problem on different $stepa$

Stepa	Optimal	Mean
0.05	0.3895	**0.4344**
0.08	**0.3901**	**0.4344**
0.1	0.3926	0.4468
0.3	0.4005	0.4550
0.5	0.4173	0.5038
1.0	0.4276	0.6208
1.5	0.4517	0.6052

From Table 3, when the searching step is $stepa = 0.05,$ the value is optimal, but the running time is too long, and convergence speed is too slow. While the searching step meets an equation $stepa = 0.1$, the result is worse than $stepa = 0.05,$, so set $stepa = 0.08,$ as the most appropriate, regardless of the convergence speed, the running time, and the accuracy of the solution are appropriate.

5.2 Effect of Dimensionality D

In order to investigate the influence of the dimension on the performance of CWCA, this paper sets $d = 5,10,15,20,25,30,35,40.$ Eight different dimensions run 100 times independently, and compare with other tens of algorithm. Table 4, Table 5 and Table 6 show the best minima, the worst minima and the mean minima found by each algorithm respectively. In the simulations, we use the same population size and the maximum number of iterations ($n = 30, \max t = 200)$.

From Table 4 we can see that CWCA is better than other algorithms no matter how large D is. For example, $D = 10$, the result got by CWCA is 0.6518, but the solution of other algorithms is more than 1, the accuracy of CWCA is higher than several other algorithms.

Table 4. Best normalized optimization results on UCAV path planning problem on different D

D	Algorithm										
	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA	CWCA
5	11.3724	10.3302	4.3568	9.5895	4.3585	5.2471	4.3573	9.7627	5.1667	5.6538	**3.3791**
10	10.2281	2.9472	1.3950	7.4272	1.3990	1.6068	1.3966	33.1123	2.2073	1.5489	**0.6518**
15	8.5298	2.5569	0.6114	8.2547	0.6172	0.8711	0.6115	57.2225	2.0969	0.8071	**0.4230**
20	10.4451	4.7230	0.5102	10.2329	0.4626	0.8252	0.4552	80.1521	2.4643	0.8460	**0.3901**
25	11.5490	5.5286	0.5512	13.3685	0.4908	1.2421	0.4571	109.7418	3.7378	1.2394	**0.3881**
30	13.2299	6.6071	0.8987	15.7251	0.6828	1.9218	0.5160	180.1498	3.2993	1.6165	**0.3913**
35	16.9599	13.0206	2.5372	16.7445	1.0829	2.3109	0.4709	220.3331	5.5025	1.6326	**0.3793**
40	19.7946	13.5504	4.5490	18.2314	1.5225	2.2084	0.4506	340.6174	5.7367	2.6180	**0.3638**

Table 5 shows the worst normalized optimization results on different D . From it, we can see that PBIL gets the worst results compared with other algorithms, its results increase with D rising. When D increases from 25 to 30, the results increase in a nearly four-fold. While the result obtained by CWCA decreases by 0.3. With the rise of the dimension, results obtained by most of algorithms are in a rising trend, but some parts of algorithms decline when D increases from 20 to35. The worst result of CWCA is better than several other algorithms.

Table 5. Worst normalized optimization results on UCAV path planning problem on different D

D	Algorithm										
	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA	CWCA
5	13.3199	121.5724	12.2083	62.2665	15.7395	11.6013	12.4186	22.2463	16.0713	11.2006	**3.5307**
10	18.1911	26.8270	6.7358	73.4605	6.7095	10.1096	3.7858	69.2468	18.6221	6.1652	**0.7092**
15	11.0084	40.3705	12.5808	53.8683	44.2763	7.4472	3.8319	139.2557	37.3201	11.7962	**0.4479**
20	17.1887	28.2063	14.5783	31.4587	28.9142	9.1795	2.0279	287.3709	28.1596	18.9518	**0.5511**
25	12.0733	30.3315	19.6664	33.9148	16.4518	10.3977	3.7043	649.6845	28.1399	15.6967	**0.5221**
30	14.7139	28.5885	24.1279	41.3024	15.9757	12.7183	8.3364	2364.08	43.6950	14.7140	**0.5400**
35	18.7271	43.8512	34.4447	38.7646	33.8871	24.4790	5.8830	6312.96	32.8328	17.6058	**0.5540**
40	27.0641	40.7087	43.2604	46.4224	36.6626	22.0688	7.7236	7053.50	34.7302	17.8669	**0.6821**

Table 6 is the mean normalized optimization results on different D, when CWCA sets different D, the result of CWCA is better than other algorithms. For instance, when D meets the equation $D = 40$, the mean of CWCA is 0.5099, while the mean of ACO is 24.5754. It can be seen clearly that CWCA is better than ACO, and the results of other algorithms are worse than CWCA.

Table 6. Mean normalized optimization results on UCAV path planning problem on different D

D	Algorithm										
	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA	CWCA
5	11.5151	22.7318	8.5962	30.7228	8.7499	10.4747	9.1673	16.1391	9.9061	10.5013	**3.4072**
10	11.9485	7.9650	3.1045	26.2868	2.1801	2.5422	1.5740	51.4355	7.0411	2.2790	**0.6819**
15	10.2554	9.5257	2.2783	21.8618	2.8217	2.1880	0.8967	78.2477	8.3395	1.8910	**0.4344**
20	16.2205	11.8761	2.7221	20.1892	3.7327	3.0900	0.7004	135.4365	8.2483	3.1670	**0.4344**
25	11.5674	14.7800	4.4081	22.7794	3.9039	3.7814	0.9987	207.7272	10.2627	4.1567	**0.4458**
30	13.9593	17.8746	9.9884	24.7757	4.9621	5.0079	1.3568	345.5447	12.3847	4.5211	**0.4339**
35	18.3108	21.5615	17.9027	26.5217	5.9955	5.9599	1.6009	634.6550	14.1354	5.8260	**0.4379**
40	24.5754	24.8531	27.6201	30.2595	7.8558	7.4927	2.1978	1119.72	14.8845	7.1100	**0.5099**

From Figure 4, Figure 5, and Figure 6, it can be seen that the optimal value, the worst value and the average value obtained by CWCA are better than other algorithms. Though with the increase of dimension, the results of CWCA also increase, yet in some cases, it is in a downward trend and the accuracy of the solution is improved indeed. With the increasing of dimensions, the complexity and running time of the algorithm grow, and the accuracy is not improved too much. In summary from these tables above, setting $D = 20$ *or 25* is more appropriate, the accuracy of the two solutions have some difference. Considering the complexity of the algorithm, if $D = 20$, its complexity is $o(n)$, that when we set $D = 25$, its complexity is $1.25O(n)$, so we set $D = 20$.

Figures 5~12 show that When D is set different values; we can get the position of fight at every moment. In Figure 5, D =5, it can be seen that the fight route is consist of seven points, and after removing the start point and the end point, the route is just matched up to D =5. We can see the flight route and the threat center form the figure. We can also see how the airplane escapes from the threat center to make the threat cost become smallest. UCAV path planning is to find an appropriate path which the threat is smallest.

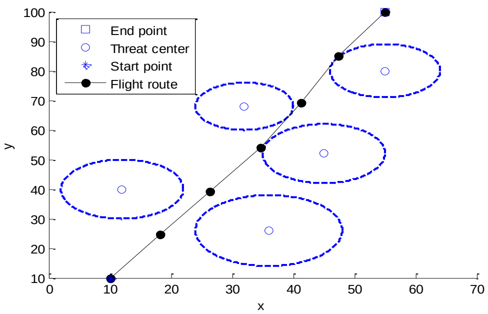

Figure 5. D=5 the result of Path Planning

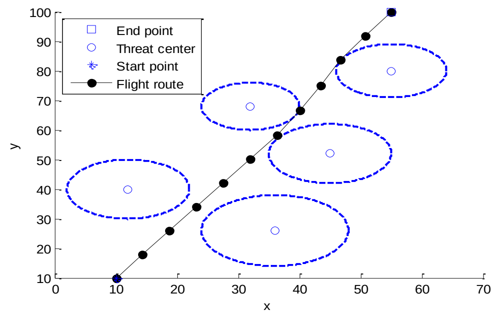

Figure 6. D=10 the result of Path Planning

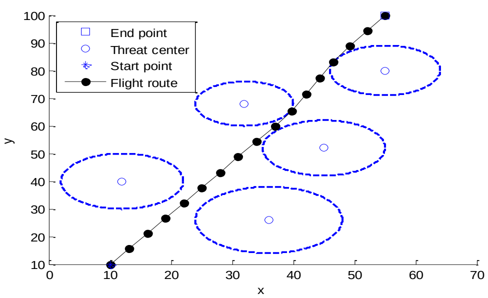

Figure 7. D=15 the result of Path Planning

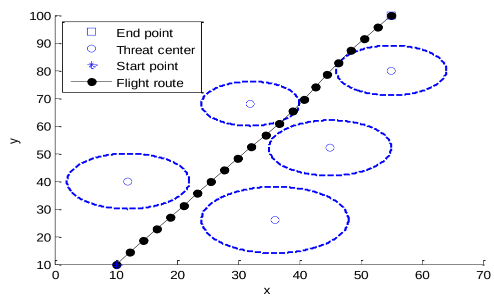

Figure 8. D=20 the result of Path Planning

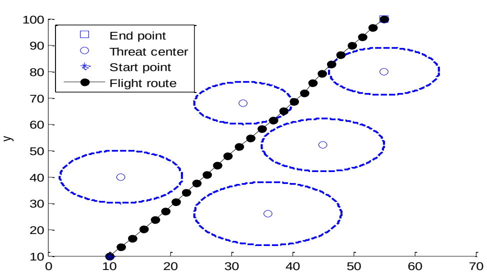

Figure 9. D=25 the result of Path Planning

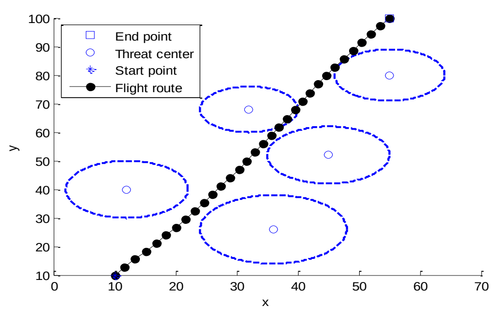

Figure 10. D=30 the result of Path Planning

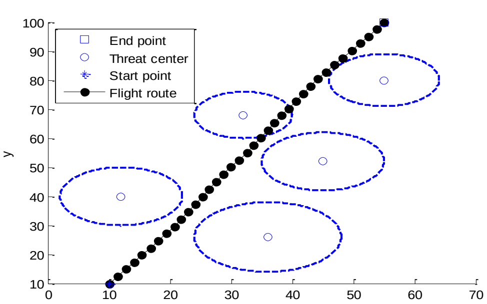

Figure 11. D=35 the result of Path Planning

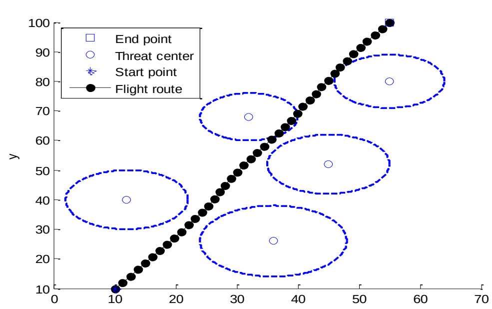

Figure 12. D=40 the result of Path Planning

5.3 Effect of maximum iterations $\max t$

The choice of the maximum number of iterations is of vital importance to solve the algorithm. The choice of the maximum number of iterations has a directly effect on the solution accuracy of different problems, so we set the maximum number of iterations as 50, 100, 150, 200, 250 respectively, and each algorithm runs 100 times independently ($N = 30, d = 20$). Table 7, Table 8 and Table 9 are the best, worst and mean results respectively on different maximum number of iterations for every algorithm.

Table 7. Best normalized optimization results on UCAV path planning problem on different *maxt*

maxt	Algorithm										
	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA	CWCA
50	9.6276	4.8658	3.6814	7.7983	1.4713	1.6118	0.7030	66.0274	3.6012	1.7041	**0.4232**
100	11.5242	4.2616	0.9439	40.2329	0.6577	4.5294	0.5382	48.9269	2.3357	1.3520	**0.3986**
150	5.6381	5.6105	0.7015	9.8027	0.5459	1.2042	0.4857	47.4630	2.6165	0.9498	**0.3956**
200	11.2445	2.9424	0.5188	10.7960	0.4931	1.0695	0.4661	18.6980	2.3469	0.8392	**0.3901**
250	9.7607	3.5209	0.4829	10.2540	0.4753	0.8781	0.4508	20.8802	2.9229	0.7839	**0.3878**

Table 8. Worst normalized optimization results on UCAV path planning problem on different *maxt*

maxt	Algorithm										
	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA	CWCA
50	18.7099	28.6806	31.3392	33.1927	28.0425	10.9773	4.6726	312.9370	33.1539	17.6318	**0.6800**
100	17.7404	28.2427	19.4058	33.1979	29.3022	11.1678	4.5749	373.8334	27.8806	11.6446	**0.5710**
150	17.4223	40.1797	14.5560	35.7277	27.8480	17.5637	4.9631	210.6140	28.3542	15.2145	**0.5691**
200	17.0679	32.1981	16.6736	52.4090	26.5768	11.9124	9.1502	183.9630	28.2524	6.7380	**0.5511**
250	17.0679	27.6544	8.5122	46.0828	26.3005	7.4338	3.6783	169.1446	29.6341	16.2672	**0.5714**

Table 9. Mean normalized optimization results on UCAV path planning problem on different *maxt*

maxt	Algorithm										
	ACO	BBO	DE	ES	FA	GA	MFA	PBIL	PSO	SGA	CWCA
50	16.2648	14.2076	13.2645	20.3152	6.2034	4.1142	1.9576	151.9844	9.9671	5.2389	**0.5407**
100	16.3500	13.4074	7.3195	20.4387	4.3526	3.7731	1.3048	113.6434	8.9057	3.7475	**0.4713**
150	16.1722	12.6978	3.5255	19.9485	4.1809	3.4671	0.9933	90.8722	8.5509	3.1680	**0.4563**
200	16.2154	11.8556	2.3975	20.7501	2.2791	2.9711	0.8984	74.1964	8.9892	2.3792	**0.4344**
250	16.0444	11.9654	2.4849	20.1739	2.2064	2.6605	0.7025	65.4942	9.2143	2.5929	**0.4347**

From Table 7, Table 8, and Table 9, it can be seen that the CWCA can get better results than other algorithms. With the rising of the number of iterations, the accuracy of the solution of every algorithm grows high. But the complexity of the algorithm is getting higher as the number of iterations increasing, and the accuracy of the algorithm is not improved greatly. Such as in Table 7, when the maximum number of iterations is 250, the solution of MFA increases about 0.01 compared in 200 iterations, while the algorithm iterates more for 50 times. But the accuracy of the solution of CWCA has little change, the maximum number of iterations is set 200, and CWCA is better than other algorithms when solving the UCAV path planning problem.

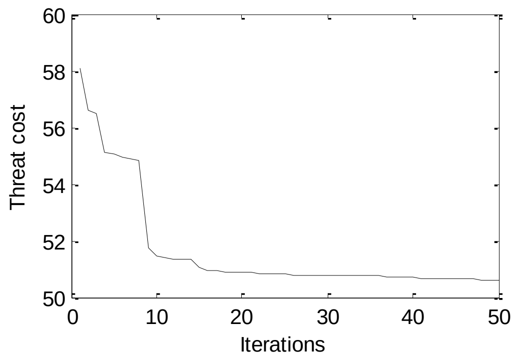

Figure. 13 Convergence curve of maxt=50

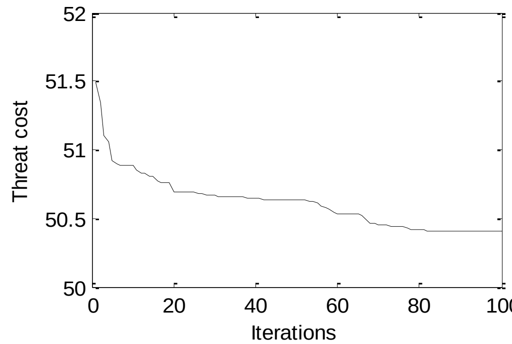

Figure. 14 Convergence curve of maxt=100

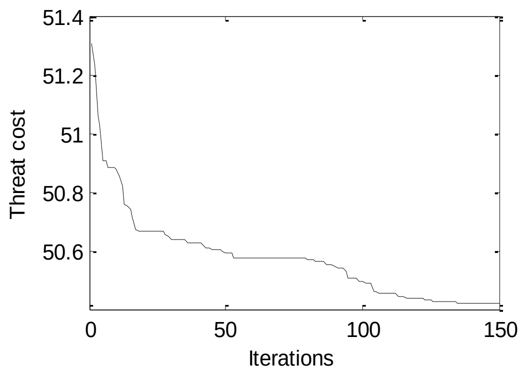

Figure. 15 Convergence curve of maxt=150

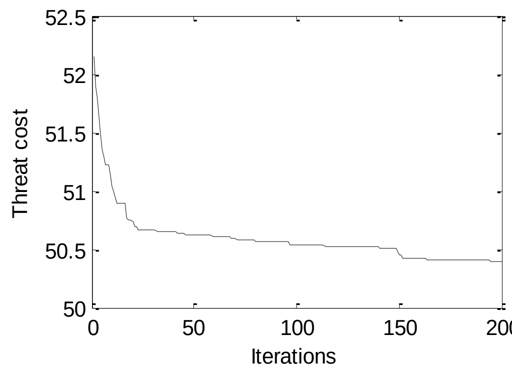

Figure. 16 Convergence curve of maxt=200

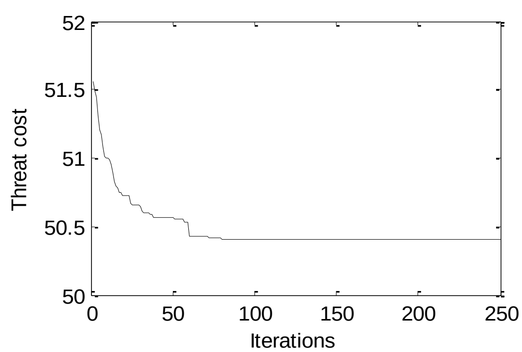

Figure. 17 Convergence curve of maxt=250

Figure 13 to Figure 17 is convergence curve figures of CWCA on different maximum number of iterations. From the figures, we can see that the convergence speed of CWCA is fast, and it is not easy to fall into local optimization. Such as Figure 15, it is converged from 20-th iteration to 80-th iteration; the algorithm jump out of local optimization at last, the accuracy of the solution is improved. It is uneasy to precocious and is effective for solving UCAV path planning problem.

6. Conclusions

For UCAV path planning problem, this paper proposes a wolf colony search algorithm based on the complex method, which let WCA mix with the complex method and the complex method as the wolf colony's leading strategy. The UCAV can find the safe path by connecting the chosen nodes while avoiding the threat areas and costing minimum fuel. Compared with other algorithms, experiment results show that CWCA is a more feasible and effective way in UCAV path planning.

Acknowledgements

This work is supported by National Science Foundation of China under Grant No. 61165015. Key Project of Guangxi Science Foundation under Grant No. 2012GXNSFDA053028, Key Project of Guangxi High School Science Foundation under Grant No. 20121ZD008, and the Innovation Project of Guangxi Graduate Education under Grant No. YCSZ2012063.

References

[1] R. Storn and K. Price, "Differential Evolution-A Simple and Efficient Heuristic for Global Optimization over Continuous Space", Journal of Global Optimization, vol. 11, **(1997)**, pp. 341-359.
[2] J. H. Holland, "Adaptation in Natural and Artificial Systems", University of Michigan Press, Ann Arbor, Mich, USA, **(1975).**
[3] W. Ye, D. W. Ma and H. D. Fan, "Algorithm for Low Altitude Penetration Aircraft Path Planning with Improved Ant Colony Algorithm", Chinese Journal of Aeronautics, vol. 18, no. 4, **(2005)**, pp. 304-309.
[4] J. Kennedy and R. Eberhart, "Particle Swarm Optimization", Neural Network, Proceedings, IEEE International Conference on IEEE, vol. 4, **(1995)**, pp. 1942-1948.
[5] X. S. Yang, "Nature-Inspired Metaheuristic Algorithms", Luniver Press, Frome, **(2008).**
[6] G. Wang, L. Guo, H. Duan, L. Liu and H. Wang, "A Modified Firefly Algorithm for UCAV Path Planning", International Journal of Hybird information Technology, vol. 5, no. 3, **(2012)**, pp. 123-143.
[7] C. Yang, X. Tu and J. Chen, "Algorithm of Marriage in Honey Bees Optimization Based on the Wolf Pack Search", International Conference on Intelligent Pervasive Computing, **(2007)**, pp. 462-467.
[8] Y. V. Pehlivanoglu, "A new vibrational genetic algorithm enhanced with a Voronoi Diagram for Path Planning of Autonomous UAV", Aerospace Science and Technology, vol. 16, no. 1, **(2012)**, pp.47-55.
[9] 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 Press, **(2008).**
[10] H. B. Duan, X. Y. Zhang and C. F. Xu, "Bio-Inspired Computing", Science Press, Beijing, China, **(2011).**
[11] M. J. Box, "A New Method of Constrained Optimization and A Comparison with Other Method", Computer Journal, vol. 8, no. 1, **(1965)**, pp. 42-52.
[12] W. Khatib and P. J. Fleming, "The stud GA: A Mini Revolution? Parallel Problem Solving from Nature-PPSN V", Springer Berlin Heidelberg, **(1998)**, pp. 683-691.
[13] Y. Zhou, Q. Lou, H. Chen and J. Wu, "Using Differential Evolution Invasive Weed Optimization for Solving Circle Packing Problem", Advanced Science Letters, vol. 19, no. 6, **(2013),** pp. 1807-1810.
[14] S. Baluja, "Population-based Incremental Learning", A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning, Carnegie-Mellon Univ Pittsburgh Pa Dept. of Computer Science, **(1994).**
[15] D. Simon, "Biogeography-based Optimization", Evolutionary Computation, IEEE Transactions on, vol. 12, no. 6, **(2008)**, pp. 702-713.
[16] E. Mayr, "Behavior Programs and Evolutionary Strategies", Natural Selection Sometimes Favors a Genetically "closed" Behavior Program, Sometimes An "open" one, American Scientist, vol. 62, no. 6, **(1974)**, pp. 650-659.
[17] C. Liu, X. Yan, C. Liu and H. Wu, "The Wolf Colony Algorithm and Its Application", Chinese Journal of Electronics, vol. 20, no. 2, **(2011)**, pp. 212-216.
[18] D. Simon, "The Matlab Code of Biogeography-based Optimization", http://academic.csuohio.edu/simond/bbo/, **(2012)**.
[19] C. Xu, H. Duan and F. Liu, "Chaotic Artificial Bee Colony Approach to Uninhabited Combat Air Vehicle (UCAV) Path Planning", Aerospace Science and Technology, vol. 14, no. 8, **(2010)**, pp. 535-541.
[20] H. Duan, X. Zhang and C. Xu, "Bio-inspired Computing", Science Press, **(2011)**, pp. 121-126.

Authors

Qiang Zhou M.S. He is currently research interest is in computation intelligence, swarm intelligence algorithm.

Yongquan Zhou, Ph.D & Prof. He received the MS degree in computer science from Lanzhou University, Lanzhou, China, in 1993 and the Ph.D degree in computation intelligence from the Xiandian University, Xi'an, China, in 2006. He is currently a professor in Guangxi University for Nationalities. His research interests include computation intelligence, neural networks, and intelligence information processing et al. He has published 1 book, and more than 150 research papers in journals.

Xin Chen M.S. He is currently research interest is in computation intelligence, swarm intelligence algorithm.