Grey Wolf Optimizer for Unmanned Combat Aerial Vehicle Path Planning

Sen Zhang[1] Yongquan Zhou[1,2] Zhiming Li[1] Wei Pan[1]

[1]College of Information Science and Engineering,Guangxi University for Nationalities, Nanning 530006, China

[2] Key Laboratory of Guangxi High Schools Complex System and Computational Intelligence, Nanning 530006, China

e-mail:yongquanzhou@126.com

Abstract: Unmanned combat aerial vehicle (UCAV) path planning is a fairly complicated global optimum problem, which aims to obtain an optimal or near-optimal flight route with the threats and constraints in the combat field well considered. A new meta-heuristic grey wolf optimizer (GWO) is proposed to solve the UCAV two-dimension path planning problem. Then, the UCAV can find the safe path by connecting the chosen nodes of the two-dimensional coordinates while avoiding the threats areas and costing minimum fuel. Conducted simulations show that the proposed method is more competent for the UCAV path planning scheme than other state-of-the-art evolutionary algorithms considering the quality, speed, and stability of final solutions.

Keywords: unmanned combat aerial vehicle; path planning; grey wolf optimizer

1 Introduction

The development of automation and unmanned flight technology has become an irresistible trend in many countries. In fact, unmanned combat aerial vehicle (UCAV) has been of great importance to many military organizations throughout the world owing to its potential to perform dangerous, repetitive tasks in remote and hazardous environments [1]. Path planning and trajectory generation is one of the critical technologies in coordinated UCAV combating. The path planning aims to offer an optimal path from a starting point to a desired destination with the artificial threats and a variety of constraints considered in the combat field. For the UCAV path planning scheme, the optimality of a flight path can be calculated through minimizing the gross flight route, fuel consumption, and exposure to radar or artillery, and so on [2]. With the development of ground defense weapons, the difficulty of describing artificial threats is greatly increased. So, solving such problems requires effective optimization techniques and consideration of several difficulties included: constrained, local solutions, and expensive objective function.

Swarm Intelligence (SI) [3] techniques have become a reliable alternatives compared to conventional optimization techniques in several engineering fields because of their promising performances when solving different kinds of real world optimization problems: parameter estimation, feature selection, neural network training, knapsack problem, and so on. Although SI-based algorithms include many methods, two of the most popular algorithms in this field are particle swarm optimization (PSO) [4] and ant colony optimization (ACO) [5] that are inspired by the social behavior of birds and marking paths via pheromone by ants when searching for food. Some of the recent SI-based algorithms have been developed and proposed in the literature such as grey wolf optimizer (GWO) [6], whale optimization algorithm (WOA) [7], ant lion optimizer (ALO) [8], virus colony search (VCS) [9], dolphin echolocation (DE) [10], and big bang-big crunch (BB-BC) [11]. These algorithms have been successfully applied to a wide range of practical problems as well. Although these algorithms are

popular, there is little application on UCAV path planning. The SI-based algorithms have the potential to solve various hard optimization problems. This is the current work, in which GWO will be analyzed in detail and employed to solve UCAV problems.

Up to now, series of algorithms have been proposed to solve the UCAV path planning problem. Some of the most popular in the literature are A^* search algorithm [12], evolutionary computation [13], genetic algorithm [14], particle swarm optimization [15], ant colony optimization [16], artificial bee colony [17, 18], and differential evolution [19]. The recent evolutionary algorithms are biogeography-based optimization approach [20], firefly algorithm [21], cuckoo search [22], bat algorithm [23], flower pollination algorithm [24], intelligent water drops [25], and imperialist competitive algorithm [26]. However, those methods can be easily trapped into the local best, therefore would probably end up without finding a satisfying trajectory path. Moreover, there is a theorem here called No Free Lunch (NFL) [27] that has been logically proved that there is no meta-heuristic best suited for solving all optimization problems. In other words, a particular meta-heuristic may show very promising results on a case, but the same algorithm may show poor performance on another. These reasons allow researcher to investigate of new algorithms in UCAV path planning problem.

Grey wolf optimizer (GWO) was originally presented by Mirjalili et al. [6] in 2014. It simulates hunting behavior and social leadership of grey wolves in nature. Some of the advantages of GWO are simplicity, flexibility, derivation-free mechanism, and local optima avoidance. Also, it is easy to implement; and it requires few control parameters to regulate. First, GWO is fairly simple. It is inspired by hunting behavior and social leadership of grey wolves in nature. The inspirations are related to animals' behaviors that are pretty easy to understand. Furthermore, the simplicity assists some scientists engaging in different research fields to learn the algorithm quickly and apply it to their problems. Second, flexibility refers to GWO applying in different problems without any special changes in the structure of the algorithm. GWO is easily applicable to different problems because it supposes problems as black boxes. Third, GWO has derivation-free mechanisms. In comparison with gradient-based optimization methods, GWO optimizes problems stochastically. During the process of optimization, there is no need to calculate the derivative of search spaces. This will be effectively used for real problems with expensive or unknown derivative information. Finally, local optima avoidance compared to conventional optimization techniques is high due to the stochastic nature of GWO. This leads to GWO highly suitable for solving highly nonlinear, multivariable, multimodal function optimization problems.

Recent studies have shown that GWO is able to provide competitive results compared to other well-known meta-heuristics. The GWO has been successfully applied to three classical engineering design problems and real optical engineering [6]. Sulaiman et al. [28] have used GWO for reactive power dispatch problem. Song et al. [29] have successfully applied GWO for surface wave analysis. Komaki et al. [30] have used GWO for two-stage assembly flow shop scheduling problem. Medjahed et al. [31] have used GWO for hyperspectral band selection. Emary et al. [32, 33] have applied GWO for attribute reduction and feature selection. Mirjalili [34] has surveyed the effectiveness of GWO in training multi-layer perceptions (MLP). Saremi et al. [35] proposed the use of evolutionary population dynamics (EPD) in the GWO to further enhance its performance. Song et al. [36] have successfully applied GWO for solving combined economic emission dispatch problems. Mirjalili et al. [37] proposed multi-objective grey wolf optimizer that is an essential in solving real problems.

However, in the field of two-dimensional path planning for UCAV, no application of GWO exists yet. In this paper, we use GWO to solve UCAV path planning problem. GWO is verified on three test cases

with different dimensions and compared with other population-based optimization methods, such as CS [38], FPA [39], NBA [40], BSA [41], ABC [42], and GGSA [43]. The rest of the paper is structured as follows.

Section 2 describes the mathematical model in UCAV path planning problem. Section 3 discusses the principles of GWO. The experimental results are conducted in Section 4. Finally, Section 5 concludes the study and advises some directions for future studies.

2 Mathematical Model for UCAV Path Planning

As a key component of mission planning system [44], UCAV path planning is a new low altitude penetration technology to achieve the purpose of terrain following and terrain avoidance and flight with evading threat. The goal for path planning is to find an optimal or near-optimal flight path for UCAV to break through the enemy threat environments, and self-survive with the perfect completion of mission. In this paper, we use the mathematical model for UCAV path planning described as follows [17, 23].

2.1 Threat resource model in UCAV path planning

In this model, S and T are defined as the starting point and the target point (see Fig. 1), respectively. There are some installations in the combat field, for instance, radars, missiles, and artilleries. The effects of such installations are presented by circles in the combat field of different radiuses and threat weights [18]. If part of its path falls in a circle, an UCAV will be vulnerable to the threat with a certain probability proportional to the distance away from the threat center. Moreover, when the fight path is outside a circle, it will not be attacked. The UCAV flight mission is to calculate an optimal path from S to T. Meanwhile all the given threat regions in the combat field and the fuel consumption should be considered.

To make this problem more specific, we draw a segment ST connecting the starting point S and the target point T. Then, ST is divided into D equal portions and vertical coordinate Y' is optimized 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 (see the red circle in Fig. 1) together, so that the route planning problem is transformed into a D-dimensional function optimization problem.

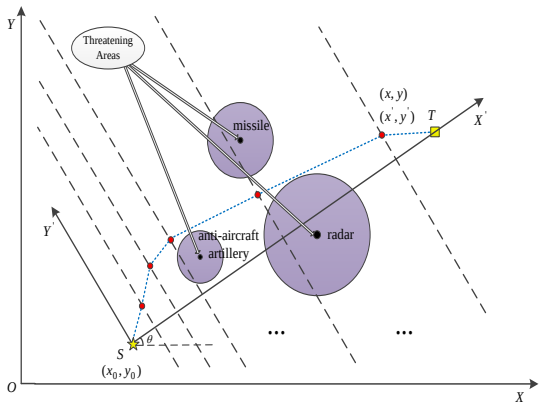

Fig. 1 UCAV battle field model

In Fig. 1, the original coordinate system is transformed into new coordinate whose horizontal axis is

the connection line from the starting point to the target point according to the transformation formula shown in Eq. (1), where the point (x, y) is coordinate in the original 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.

$$\begin{aligned} \theta &= \arcsin \frac{y_2 - y_1}{\left|\overrightarrow{AB}\right|} \\ \begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} + \begin{bmatrix} x_1 \\ y_2 \end{bmatrix} \end{aligned} \tag{1}$$

2.2 Performance evaluation

Regarding the evaluation of one candidate flight path, the threat cost J_t and the fuel cost J_f are taken into consideration as follows [17]:

$$\begin{aligned} J &= \lambda \cdot J_t + (1-\lambda) \cdot J_f \\ &= \lambda \cdot \int_0^{length} w_t dl + (1-\lambda) \cdot \int_0^{length} w_f dl \end{aligned} \tag{2}$$

where J is the weighted sum of flight cost for this flight path, w_t and w_f are variables close related with the current path point and changing along with l, which respectively present the threat cost and fuel cost of each line segment on the route, and $length$ is the total length of the generated path. $\lambda \in [0, 1]$ denotes the weighting parameter. When λ is close to 1, a shorter path is needed to be planned, and less attention is paid to the radar's exposed threat. In addition, when λ is close to 0, it requires avoiding the threat as far as possible on the cost of sacrifice the trajectory length. In this paper, λ is set to 0.5 according to [17, 18, 20-24]. The optimized path is founded only when function J reaches its minimal value.

When the UCAV is flying along the path $L_{i \to i+1}$, the total threat cost generated by N_t threats is calculated using $w_{t, L_i \to L_{i+1}} = \int_0^{length_i} \sum_{k=1}^{N_t} \frac{t_k}{[(x-x_k)^2 + (y-y_k)^2]^2} \, dl$. In order to simplify the integral operations in Eq. (2), the detection cost $w_{t, L_i \to L_{i+1}}$ from the point along L_i to the one along L_{i+1} is calculated at five sample points [18], as shown in Fig. 2.

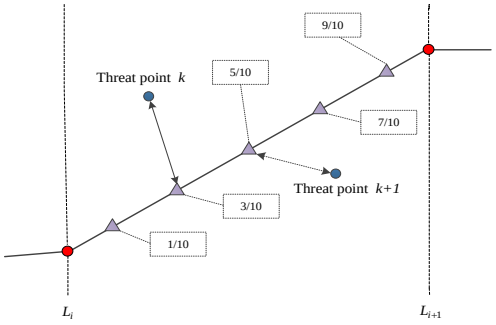

Fig. 2 Computation of threat cost

If the distance from the threat point to the end of the each sub-segment is within threat radius, the threat cost is calculated as follows [17]:

$$w_{t,\,L_i\to L_{i+1}}=\frac{length_{\,i}}{5}\cdot\sum_{k=1}^{N_t}\left[t_k\cdot(\frac{1}{d_{0.1,\,i,\,k}^4}+\frac{1}{d_{0.3,\,i,\,k}^4}+\frac{1}{d_{0.5,\,i,\,k}^4}+\frac{1}{d_{0.7,\,i,\,k}^4}+\frac{1}{d_{0.9,\,i,\,k}^4})\right] \quad (3)$$

where N_t is the number of threatening areas, $length_{\,i}$ is the i th sub-path length, $d_{0.1,\,i,\,k}^4$ is the distance from the $1/10$ point on the path and the k th threat center, and t_k is the threat level of k th threat. It is assumed that the speed of UCAV is a constant. It is common practice to assume that the fuel cost J_f is in direct proportion to $length$, which means w_f will be a constant [18]. This in no problem if we set $w_f\equiv 1$ because the polynomial $(1-\lambda)$ in Eq. (2) is always attached with w_f [17, 18].

3 Grey Wolf Optimizer (GWO)

3.1 Mainframe of GWO

Grey wolf optimizer [6] is a new meta-heuristic optimization method, which is proposed by Mirjalili et al. and based on the simulation of the hunting behavior and social leadership of grey wolves in nature. Similarly to other meta-heuristics, it initializes the optimization process by generation a set of random candidate solutions. During every iteration, the first three best wolves are considered as alpha (α), beta (β), and delta (δ), who lead other wolves toward promising zones of the search space. The rest of grey wolves are thought of as omega (ω) who update their positions encircling α , β , or δ as follows [6]:

$$\vec{D}=\left|\vec{C}\cdot\vec{X}_p(t)-\vec{X}(t)\right| \quad (4)$$

$$\vec{X}(t+1)=\vec{X}_p(t)-\vec{A}\cdot\vec{D} \quad (5)$$

Where t is the current iteration, $\vec{C}=2\cdot\vec{r}_2$, $\vec{A}=2\vec{a}\cdot\vec{r}_1-\vec{a}$, $\vec{X}_p$ is the position vector of the prey, $\vec{X}$ is the position vector of a grey wolf, $\vec{a}$ is gradually decreased from 2 to 0, and r_1 , r_2 are random numbers in $[0,1]$.

It should be noted here that during optimization the ω wolves update their positions encircling α , β , and δ . So each ω wolf can reposition relative to α , β , and δ concurrently as follows [6]:

$$\vec{D}_\alpha=\left|\vec{C}_1\cdot\vec{X}_\alpha-\vec{X}\right|,\ \vec{D}_\beta=\left|\vec{C}_2\cdot\vec{X}_\beta-\vec{X}\right|,\ \vec{D}_\delta=\left|\vec{C}_3\cdot\vec{X}_\delta-\vec{X}\right| \quad (6)$$

$$\vec{X}_1=\vec{X}_\alpha-\vec{A}_1\cdot(\vec{D}_\alpha),\ \vec{X}_2=\vec{X}_\beta-\vec{A}_2\cdot(\vec{D}_\beta),\ \vec{X}_3=\vec{X}_\delta-\vec{A}_3\cdot(\vec{D}_\delta) \quad (7)$$

$$\vec{X}(t+1)=\frac{\vec{X}_1+\vec{X}_2+\vec{X}_3}{3} \quad (8)$$

Where $\vec{X}_\alpha$ denotes the position of the alpha, $\vec{X}_\beta$ denotes the position of the beta, $\vec{X}_\delta$ denotes the

position of the delta, $\vec{C}_1$, $\vec{C}_2$, $\vec{C}_3$ are random vectors and $\vec{X}$ is the position of the current solution.

According to [6], the parameters A and C oblige the GWO algorithm to explore and exploit the search space. With decreasing A, half of the iterations are devoted to exploration (when $|A|>1$) and the rest are dedicated to exploitation (when $|A|<1$). As can be seen from the above equations, the $\vec{C}$ vector contains random values in $[0, 2]$. This component provides random weights for prey in order to stochastically emphasize ($C>1$) or deemphasize ($C<1$) the effect of prey in defining the distance in Eq.(4). This assists GWO to show a more random behavior across optimization, favoring exploration, and local optima avoidance. More detailed information about the GWO algorithm can be found in [6].

Based on these approximations and idealization, the basic steps of grey wolf optimizer can be made a summing up in Algorithm 1.

Algorithm 1. GWO algorithm

Begin
Step 1: Initialize the grey wolf population $X_i (i=1,2,\ldots,n)$
Initialize a, A, and C
Step 2: For all X_i **do**
Calculate fitness $F(X_i)$
End for
Get the first three best wolves as X_α, X_β, and X_δ
Step 3: While($t <$ *Max number of iterations*)
For each search agent
Update the position of the current search agent by Eq.(8)
End for
Update a, A, and C
For all X_i **do**
Calculate fitness $F(X_i)$
End for
Update X_α, X_β, and X_δ
$t = t+1$
End while
Return X_α
End

3.2 Algorithm GWO for UCAV path planning

In GWO, the standard ordinates are inconvenient to solve UCAV path planning directly. For the sake of applying GWO to UCAV path planning, one of the critical issues is to transform the original ordinate into rotation ordinate by Eq. (1).

Fitness of grey wolf i at position $\vec{X}_i$ is determined by the threat cost by Eq. (2), and the smaller the threat cost, the smaller the fitness of grey wolf i at position $\vec{X}_i$. Each grey wolf is encoded by D-dimensional deciding variables. And then, we use GWO to optimize flight path for UCAV. At last,

the best solution is inversely converted to the original ordinates and output. The algorithm GWO for UCAV path planning is shown in Algorithm 2.

Algorithm 2. GWO algorithm for UCAV path planning

Begin

Step 1: Initialize the grey wolf population $X_i (i = 1,2,\ldots,n)$

Initialize a , A ,and C

Step 2: Transform the original coordinate system into new rotation coordinate whose horizontal axis is the connection line from starting point to target point by Eq. (1);

Convert battlefield threat information to the rotation coordinate system and divide the axis ST into D equal portions. Each feasible solution, denoted by $X_i = \{x_i, i = 1,2,\ldots,D\}$, is an array indicated by the composition of D coordinates.

Step 3: For all X_i **do**

Evaluate the weighted sum of flight cost J by Eq. (2)

End for

Get the first three best wolves as X_α , X_β ,and X_δ

Step 4: While($t < Max\ number\ of\ iterations$)

For each search agent

Update the position of the current search agent by Eq. (8)

End for

Update a , A ,and C

For all X_i **do**

Evaluate the weighted sum of flight cost J by Eq. (2)

End for

Update X_α , X_β ,and X_δ

$t = t + 1$

End while

Return X_α

Inversely transform the coordinates in final optimal path into the original coordinate, and output

End

4 Simulation Experiments

4.1 Simulation platform

All the algorithms are implemented in Matlab R2012 (a). The test environment is set up on a computer with Intel Core i5-4590 CPU, 3.30GHz, 4GB RAM, running on Windows 7.

4.2 The value of λ setting

To illustrate how the value of λ in the model impacts on the total cost, we let our algorithm run 30 times in the same battlefield using case 1 in Table 2 as an example. Table 1 shows the mean total cost in the same battlefield when the value of λ is from 0.1 to 0.9. It is obvious that the total cost value decreases with the λ increases in the same battlefield. In this study, λ is set to 0.5 that means we suppose the importance of safety and fuel is equal. Such experimental set has been carefully selected to assure compatibility between similar works reported in the literature [17, 18, 20-24].

Table 1 Mean optimization results on different λ

λ	D					
	10	15	20	25	50	100
0.1	90.7714	90.7522	90.7442	90.8360	92.3594	95.3121
0.3	70.7503	70.6272	70.6564	70.6966	71.9049	74.8847
0.5	50.7532	50.4957	50.5146	50.5516	50.5676	50.6260
0.7	30.7697	30.3743	30.3207	30.3471	31.1022	32.2139
0.9	10.7500	10.2501	10.1721	10.1341	10.4228	10.8327

4.3 Experimental setup

In this section GWO is benchmarked using three simulation cases [17, 21]. In the first case, the starting point is set to (10, 10) and the terminal point is set to (55, 100). In the second case, the starting point is (11, 11) and the terminal point is (75, 75). In the third case, the starting point is (0, 0) and the terminal point is (100, 100). The results are also compared with CS [38], FPA [39], NBA [40], BSA [41], ABC [42], and GGSA [43] for verification. The threat locations and threat grades for the three cases are listed in Table 1. The parameters of these algorithms are shown in Table 2. Such values represent the best parameter sets for these algorithms according to the original paper of GWO [6], CS [38], FPA [39], NBA [40], BSA [41], ABC [42], and GGSA [43]. In all experiments, the swarm population is set to 30 and the maximum iteration number is set to 200.

Table 2 Information of threatening areas in combat field

Case number	Threat center location	Threat radius	Threat grade
1	[45, 52]	10	2
	[12, 40]	10	10
	[32, 68]	8	1
	[36, 26]	12	2
	[55, 80]	9	3
2	[52, 52]	10	2
	[32, 40]	10	10
	[12, 48]	8	1
	[36, 26]	12	2
	[80, 60]	9	3
	[63, 56]	7	5
	[50, 42]	10	2
	[30, 70]	10	4
3	[30, 20]	10	5
	[50, 15]	21	5
	[65, 55]	10	5
	[0, 24]	17	5
	[50, 80]	12	5
	[75, 90]	10	5
	[100, 70]	20	5
	[50, 36]	11	5

Table 3 The initial parameters of algorithms

Algorithm	Parameter	Value
GWO	$\vec{a}$	linearly decreased from 2 to 0
	Accelerating coefficient $(c_1^{'})$	$(-2t^3/T^3)+2$
GGSA	Accelerating coefficient $(c_2^{'})$	$(2t^3/T^3)$
	Gravitational constant (G_0)	1
	Coefficient of decrease (α)	20
ABC	Limit	D
	Cognitive coefficient (c_1)	1.5
	Social coefficient (c_2)	1.5
BSA	Indirect behavior (a_1)	1
	Direct behavior (a_2)	1
	Frequency of flight behavior (FQ)	10
	Constants α and γ	0.9
	Loudness (A_0)	$[0, 2]$
	Pulse rate (r_0)	$[0, 1]$
NBA	Loudness and pulse emission rate (G)	10
	Probability of habitat selection (P)	$[0.5, 0.9]$
	Contraction-expansion coefficient (θ)	$[0.5, 1]$
	Compensation rate (C)	$[0.1, 0.9]$
	Inertia weight (w)	$[0.4, 0.9]$
FPA	Probability switch (p)	0.8
CS	Discovery rate pa	0.25

where D denotes dimension of the problem, T indicates the maximum number of iterations, and t is the current iteration.

The experimental results are listed in Table 4, Table 6, and Table 8. The results are averaged over 30 independent runs. In these tables D denotes dimension of the problem, and the Best, Worst, Mean and Std represent the optimal fitness value, worst fitness value, mean fitness value and standard deviation respectively. The best results are denoted in bold type. Please note here that we can get the source code of GWO from http://www.alimirjalili.com/Projects.html.

To improve the performance evaluation of evolutionary algorithms, statistical tests should be conducted [45]. In order to determine whether the results of GWO differ from the best results of the other algorithms in a statistical method, a nonparametric test which is known as Wilcoxon's rank-sum test [46, 47] is performed at 5% significance level. Tables 5, 7 and 9 report the p values produced by Wilcoxon's test for the pairwise comparison of the best value of six groups. Such groups are formed by GWO versus CS, GWO versus FPA, GWO versus NBA, GWO versus BSA, GWO versus ABC, and GWO versus GGSA. Usually, p values < 0.05 can be considered as sufficient evidence against the null hypothesis.

4.3.1 The first test case

Table 4 Experimental results for the first test case

D	Result	CS	FPA	NBA	BSA	ABC	GGSA	GWO
	Best	50.8889	51.1650	50.7252	50.9205	50.8995	50.7628	**50.7176**
10	Worst	51.7694	52.0079	52.3014	58.0239	51.6249	59.7502	**51.0834**
	Mean	51.3070	51.5225	51.2087	53.3472	51.1399	53.4249	**50.7532**
	Std	0.2462	0.2245	0.5024	2.0145	0.1616	2.4407	**0.0883**
	Best	50.5331	50.8413	50.4542	50.8503	50.5069	50.6680	**50.4538**
15	Worst	50.7309	51.2892	51.5363	51.8120	50.9526	52.3699	**50.5671**
	Mean	50.6058	51.0777	50.8757	51.4240	50.6510	51.2269	**50.4957**
	Std	0.0454	0.1226	0.3694	0.2905	0.1135	0.4674	**0.0357**
	Best	51.3910	50.6162	50.4286	50.6617	50.5104	50.5982	**50.4172**
20	Worst	52.8854	51.2855	54.6933	50.9645	50.9386	51.0978	**50.7627**
	Mean	52.0905	51.0035	50.8844	50.7626	50.7143	50.8211	**50.5146**
	Std	0.3977	0.1658	0.9715	**0.0657**	0.1134	0.1560	0.1025
	Best	50.6762	50.4995	50.4661	50.4751	50.5864	50.5310	**50.4619**
25	Worst	50.9080	51.1415	**50.6375**	50.6477	51.2481	50.8422	50.6539
	Mean	50.7621	50.8453	**50.4933**	50.5236	50.8069	50.6613	50.5516
	Std	0.0549	0.1527	**0.0376**	0.0456	0.1564	0.0880	0.0440
	Best	51.5724	50.4992	50.5301	50.9173	52.7645	50.4450	**50.4621**
50	Worst	52.5589	50.8866	52.9879	51.4757	55.6890	50.9818	**50.6938**
	Mean	52.0340	50.6085	51.3273	51.2227	54.2649	50.5895	**50.5676**
	Std	0.2807	0.0920	0.7892	0.1456	0.8111	0.1136	**0.0487**
	Best	52.6308	50.9812	52.1106	50.8022	53.6325	52.2564	**50.4152**
100	Worst	54.0194	51.8001	61.2633	51.5266	57.8689	54.4118	**50.8796**
	Mean	53.2536	51.5029	54.1672	51.1201	55.1569	53.3526	**50.6260**
	Std	0.2730	0.1580	1.7283	0.1729	0.9626	0.5307	**0.1197**

Table 5 p values of the Wilcoxon rank-sum test via comparing GWO vs CS, FPA, NBA, BSA, ABC, and GGSA in the first case ($p \geq 0.05$ have been underlined)

GWO vs	CS	FPA	NBA	BSA	ABC	GGSA
D = 10	1.0937E-10	3.0199E-11	6.5277E-08	3.6897E-11	1.4110E-09	5.4941E-11
D = 15	1.2057E-10	3.0199E-11	3.6459E-08	3.0199E-11	1.4110E-09	3.0199E-11
D = 20	3.0199E-11	6.6955E-11	2.7726E-05	6.7220E-10	2.0283E-07	4.1825E-09
D = 25	3.0199E-11	2.0338E-09	3.0103E-07	4.2259E-03	1.9568E-10	7.5991E-07
D = 50	3.0199E-11	0.1537	3.1821E-04	3.0199E-11	3.0199E-11	0.9234
D = 100	3.0199E-11	3.0199E-11	3.0199E-11	4.5043E-11	2.9155E-11	3.0199E-11

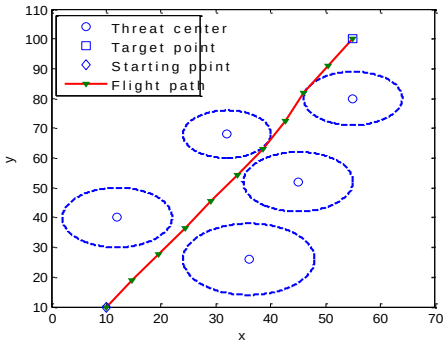

Fig. 3 Path planning result optimized by GWO algorithm in Case 1 (D = 10)

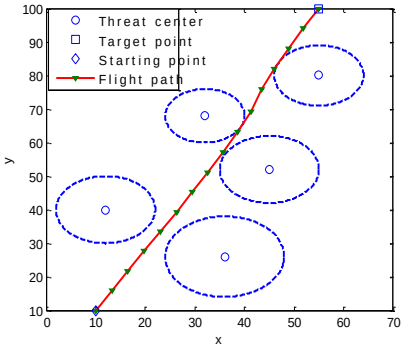

Fig. 4 Path planning result optimized by GWO algorithm in Case 1 (D = 15)

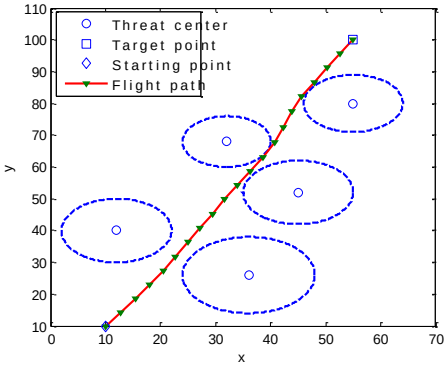

Fig. 5 Path planning result optimized by GWO algorithm in Case 1 (D = 20)

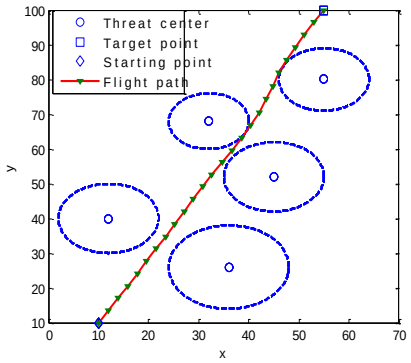

Fig. 6 Path planning result optimized by GWO algorithm in Case 1 (D = 25)

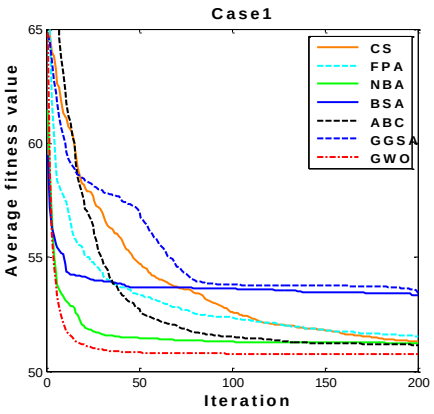

Fig. 7 Comparative convergence curves of algorithms in Case 1 (D = 10)

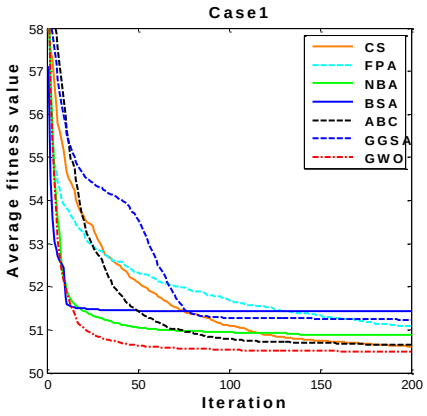

Fig. 8 Comparative convergence curves of algorithms in Case 1 (D = 15)

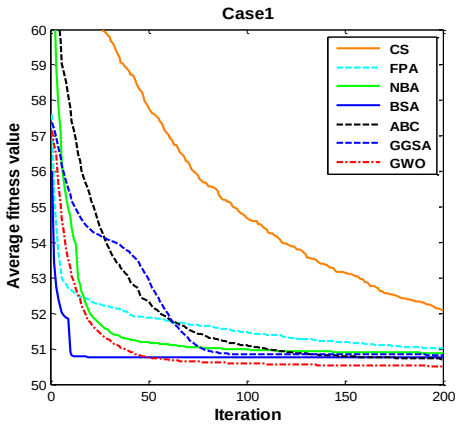

Fig. 9 Comparative convergence curves of algorithms in Case 1 (D = 20)

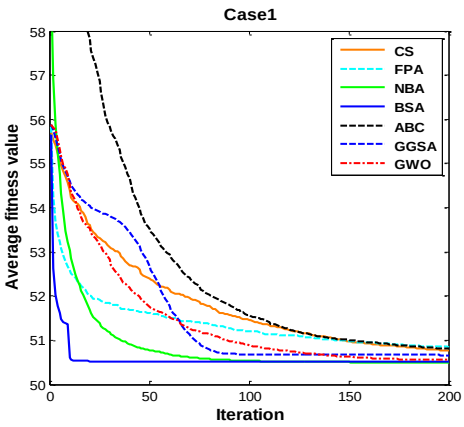

Fig. 10 Comparative convergence curves of algorithms in Case 1 (D = 25)

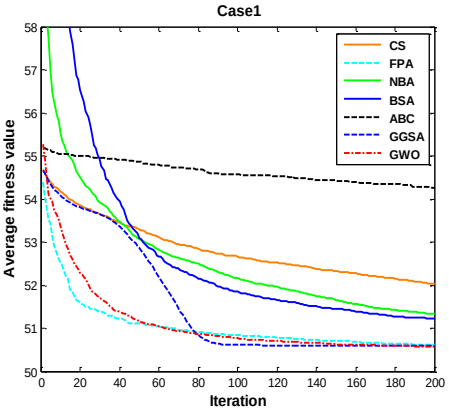

Fig. 11 Comparative convergence curves of algorithms in Case 1 (D = 50)

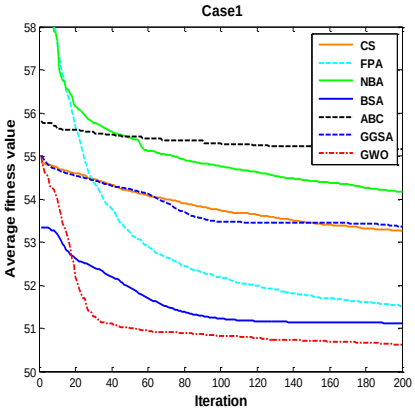

Fig. 12 Comparative convergence curves of algorithms in Case 1 (D = 100)

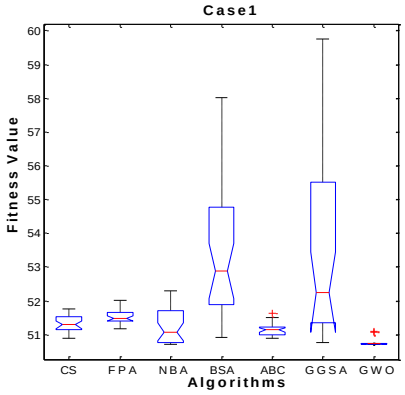

Fig. 13 Comparative ANOVA tests of algorithms in Case 1 (D = 10)

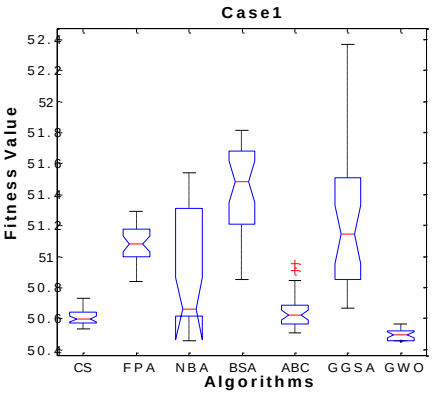

Fig. 14 Comparative ANOVA tests of algorithms in Case 1 (D = 15)

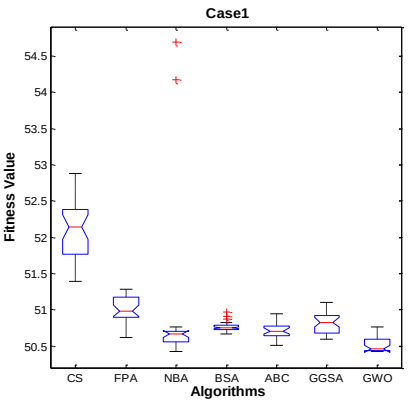

Fig. 15 Comparative ANOVA tests of algorithms in Case 1 (D = 20)

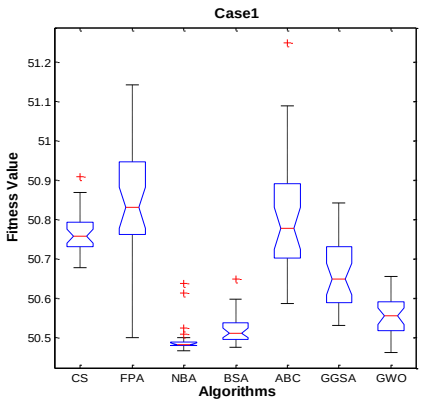

Fig. 16 Comparative ANOVA tests of algorithms in Case 1 (D = 25)

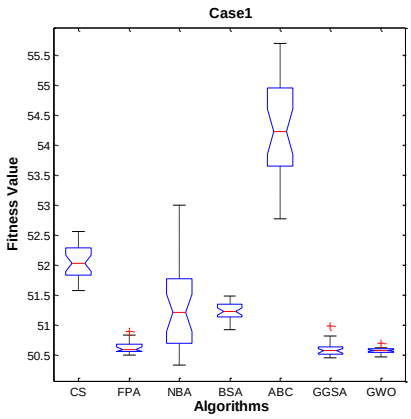

Fig. 17 Comparative ANOVA tests of algorithms in Case 1 (D = 50)

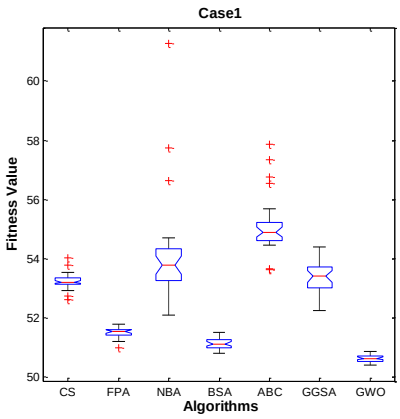

Fig. 18 Comparative ANOVA tests of algorithms in Case 1 (D = 100)

Table 4 shows the results for the first test case. As can be seen in this table, the GWO outperforms the other algorithms on 10-*D*, 15-*D*, 50-*D*, and 100-*D* in terms of the Best, Worst, Mean, and Std values of the results. The results of these statistical variables prove that GWO has the best ability to avoid local minima. For the first case on 20-*D*, the Best, Worst, and Mean values of the results for GWO are much smaller than the other algorithms, while the Std value of BSA is slightly better than GWO. For the first case on 25-*D*, NBA has the best results for Worst, Mean, and Std values. However, as the result of Best value shows, GWO performs better than the other algorithms. According to the *p* values of the first case with different dimensions in Table 5, GWO achieves significant improvement on 10-*D*, 15-*D*, 20-*D*, 25-*D*, and 100-*D* compared to other algorithms. For 50-*D*, when comparing GWO and other algorithms, we can conclude that GWO is significantly performing better in four out of six groups compared algorithms. Figs. 3~6 show the UCAV flight paths obtained by GWO testing the first case on different dimensions. We can see that the flight path is divided into D equal portions. The GWO algorithm can find the flight path that avoids the threat areas with the smallest threat cost. Figs. 7~12 illustrate the convergence cures of all algorithms based on averages with different

dimensions over 30 independent runs. Note that all the convergence curves in the following subsections are also averaged curves. As can be seen from these curves, the convergence rate of GWO in Figs. 7~9, Fig. 11, and Fig. 12 is better than the other algorithms, but BSA has the fastest convergence curve in Fig. 10. Figs. 13~16 show the graphical analysis results of the ANOVA tests dealing with the first case. From Fig. 13, Fig. 14, Fig. 17, and Fig. 18, these figures show that the boxplot of GWO is significantly lower and narrower than other algorithms. Fig. 15 shows that the boxplot of BSA is super narrow; while GWO is under the minima of other algorithms. This means that GWO tends to find the global minimum and significantly outperforms other algorithms. Fig. 16 shows that the best performance belongs to NBA.

4.3.2 The second test case

Table 6 Experimental results for the second test case

D	Result	CS	FPA	NBA	BSA	ABC	GGSA	GWO
	Best	55.3087	54.4759	58.2017	53.5735	58.1706	63.9935	**48.5129**
15	Worst	59.3489	62.8929	79.0485	71.0403	**58.8793**	101.8329	59.8476
	Mean	58.3312	59.8460	66.7618	61.0665	58.4610	72.6290	**50.4146**
	Std	0.7557	1.7991	5.6650	3.5112	**0.1545**	8.5253	3.3922
	Best	50.8408	51.4405	50.4467	51.0586	50.4323	52.0376	**50.3884**
20	Worst	52.0700	53.5758	67.7871	60.3952	51.6527	65.4804	**50.8559**
	Mean	51.2154	52.1940	56.2739	53.4634	50.9465	56.2815	**50.5808**
	Std	0.2734	0.6337	6.5182	2.0783	0.2763	3.7927	**0.1304**
	Best	51.6474	48.8181	48.0096	48.5438	48.2650	48.8063	**47.8883**
25	Worst	55.2726	53.9716	61.1633	53.9576	**51.0348**	54.3383	51.5245
	Mean	53.0967	51.5040	48.9893	50.2722	49.3217	51.1714	**48.2013**
	Std	0.8872	1.3029	2.4296	1.2461	0.6496	1.5172	**0.6388**
	Best	48.0248	47.7000	47.0098	47.3450	47.7713	47.4924	**46.9251**
30	Worst	49.5400	49.6887	48.6054	50.2379	51.0973	50.8869	**48.5753**
	Mean	48.6106	48.7535	47.4335	48.8904	49.6097	48.6565	**47.0906**
	Std	0.3764	0.5348	0.4622	0.6612	0.8095	0.8635	**0.2923**
	Best	47.5135	47.1762	46.4560	46.9963	48.3625	47.0314	**46.4326**
35	Worst	49.1512	49.3337	49.1263	50.0104	52.0304	48.7548	**46.8573**
	Mean	48.2835	48.2604	47.0469	48.1808	50.2311	47.7819	**46.5448**
	Std	0.4144	0.5313	0.5295	0.7175	1.0151	0.4801	**0.1054**
	Best	48.2535	47.1704	46.3635	46.1622	49.6705	50.0655	**46.3189**
50	Worst	49.6909	48.7243	49.1103	50.3159	56.3889	50.4728	**47.5570**
	Mean	48.9430	47.8305	47.3053	47.7252	53.3091	50.2371	**46.7002**
	Std	0.4109	0.4687	0.8564	1.0615	1.4425	**0.0965**	0.2787
	Best	50.6742	49.8418	49.8089	**49.1109**	53.8203	49.8681	49.6494
100	Worst	53.4929	**50.9730**	57.9059	51.3424	68.8826	51.4648	51.3160
	Mean	51.5772	**50.4974**	51.8049	50.5582	57.2075	50.6816	50.5455
	Std	0.5947	**0.2670**	1.9491	0.5709	3.0647	0.3904	0.3937

Table 7 *p* values of the Wilcoxon rank-sum test via comparing GWO vs CS, FPA, NBA, BSA, ABC, and GGSA in the second case ($p \geq 0.05$ have been underlined)

GWO vs	CS	FPA	NBA	BSA	ABC	GGSA
D = 15	9.8329E-08	3.4742E-10	8.1527E-11	3.8202E-10	1.0666E-07	3.0199E-11
D = 20	4.5043E-11	3.0199E-11	2.8790E-06	3.0199E-11	3.0103E-07	3.0199E-11
D = 25	3.0199E-11	1.3289E-10	9.2113E-05	3.4742E-10	7.3803E-10	1.9568E-10
D = 30	1.2057E-10	8.9934E-11	5.5999E-07	8.1527E-11	4.0772E-11	1.3289E-10
D = 35	3.0199E-11	3.0199E-11	4.6856E-08	3.0199E-11	3.0199E-11	3.0199E-11
D = 50	3.0199E-11	9.9186E-11	6.3772E-03	4.4205E-06	3.0199E-11	3.0199E-11
D = 100	2.6695E-09	0.4464	3.1573E-05	0.5895	2.8646E-11	0.2519

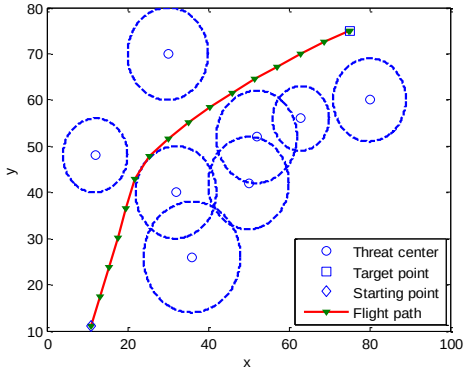

Fig. 19 Path planning result optimized by GWO algorithm in Case 2 (D = 15)

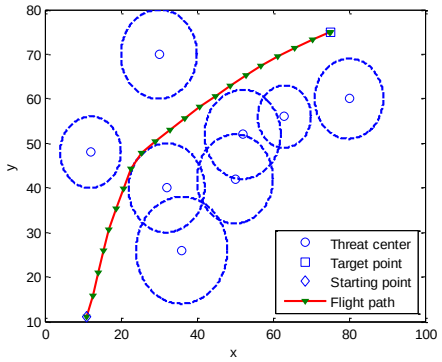

Fig. 20 Path planning result optimized by GWO algorithm in Case 2 (D = 20)

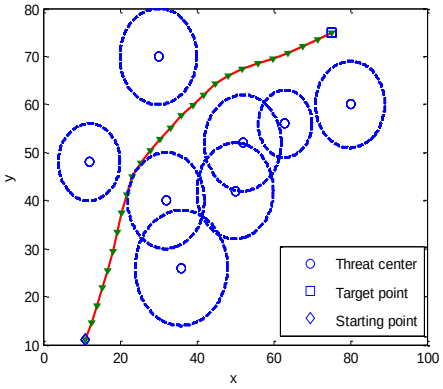

Fig. 21 Path planning result optimized by GWO algorithm in Case 2 (D = 25)

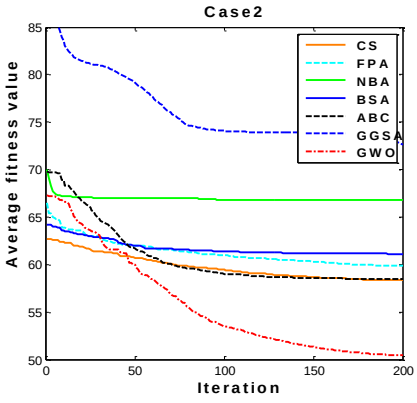

Fig. 22 Comparative convergence curves of algorithms in Case 2 (D = 15)

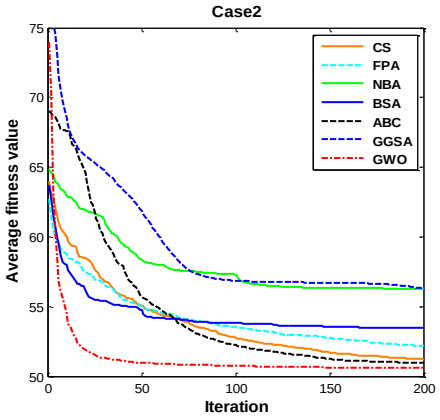

Fig. 23 Comparative convergence curves of algorithms in Case 2 (D = 20)

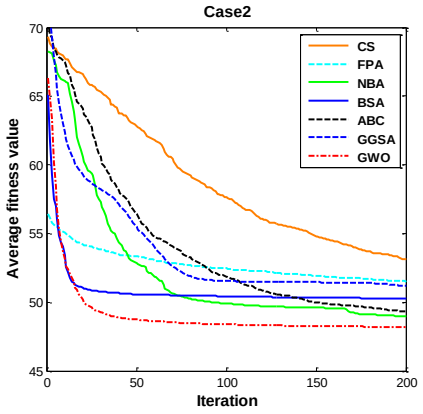

Fig. 24 Comparative convergence curves of algorithms in Case 2 (D = 25)

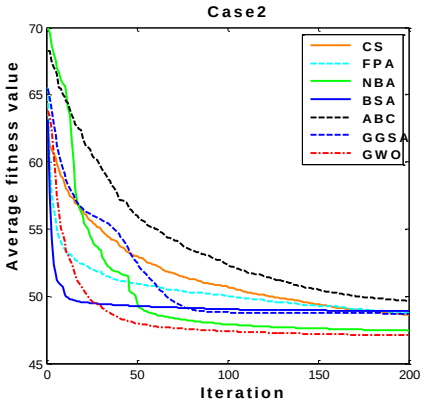

Fig. 25 Comparative convergence curves of algorithms in Case 2 (D = 30)

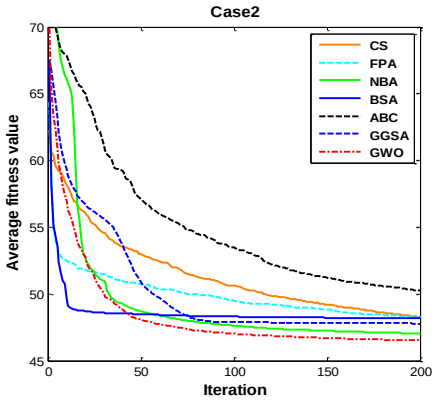

Fig. 26 Comparative convergence curves of algorithms in Case 2 (D = 35)

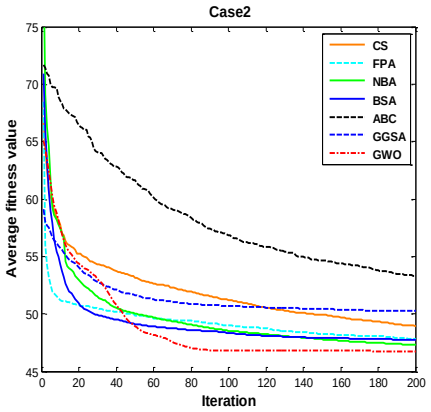

Fig. 27 Comparative convergence curves of algorithms in Case 2 (D = 50)

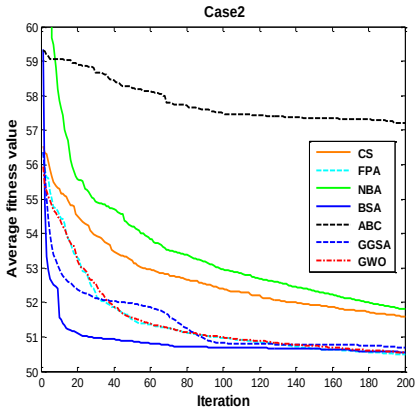

Fig. 28 Comparative convergence curves of algorithms in Case 2 (D = 100)

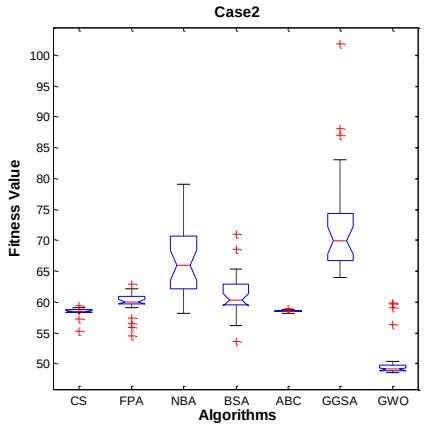

Fig. 29 Comparative ANOVA tests of algorithms in Case 2 (D = 15)

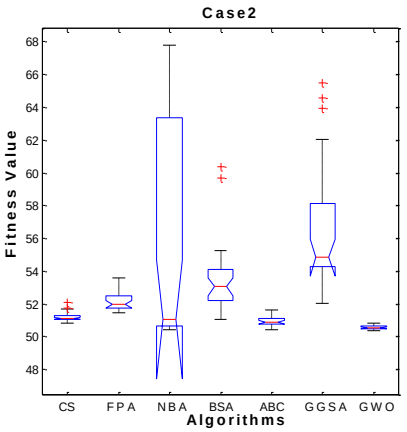

Fig. 30 Comparative ANOVA tests of algorithms in Case 2 (D = 20)

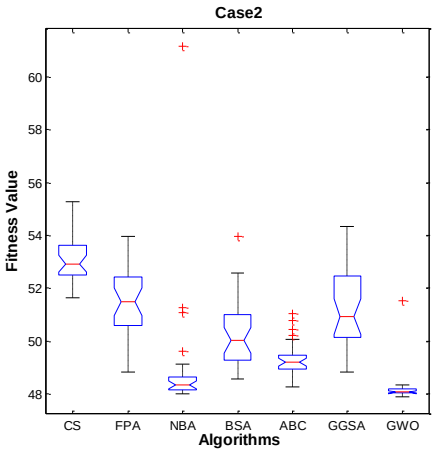

Fig. 31 Comparative ANOVA tests of algorithms in Case 2 (D = 25)

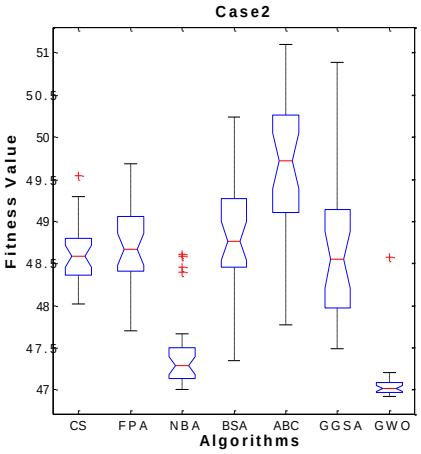

Fig. 32 Comparative ANOVA tests of algorithms in Case 2 (D = 30)

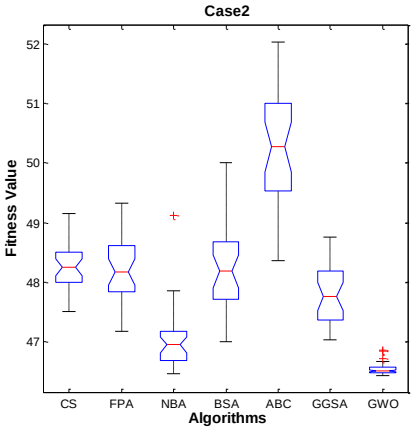

Fig. 33 Comparative ANOVA tests of algorithms in Case 2 (D = 35)

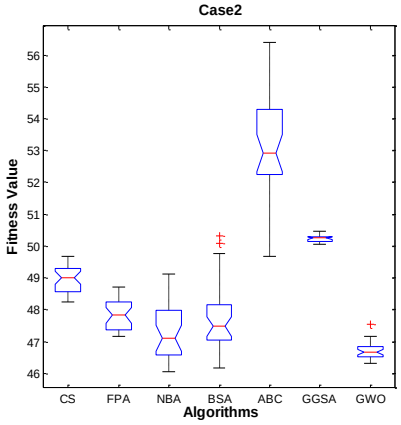

Fig. 34 Comparative ANOVA tests of algorithms in Case 2 (D = 50)

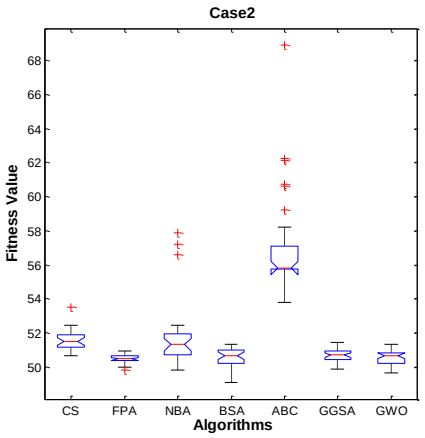

Fig. 35 Comparative ANOVA tests of algorithms in Case 2 (D = 100)

The results of the algorithms on the second case are provided in Table 6. This table shows that the Best, Worst, Mean, and Std values of the GWO on 20-*D*, 30-*D*, and 35-*D* are better than the other algorithms. These results prove that GWO also has better ability than the other algorithms to avoid local minimal in the second test case. For the Best and Mean optimization results on 15-*D*, we can see that GWO performs the best, while ABC performs the best with respect to the Worst and Std optimization results. For this case on 25-*D*, GWO provides the best results for Best, Mean, and Std values. For the Worst value on 25-*D*, GWO and ABC have very close results, but ABC still has the most accurate results. For this case on 50-*D*, GWO performs the best considering the Best, Worst, and Mean values, while for Std value GGSA is slight better than GWO. For 100-*D*, we see that, on Best value, BSA is the most effective at finding objective function minimum. However, FPA performs better than other algorithms in terms of Worst, Mean, and Std. Moreover, the *p* values of the second case with different dimensions reported in Table 7 are less than 0.05 across the majority of dimensions which is strong evidence against null hypothesis. Hence, this evidence demonstrates that the results of GWO are statistically significant not occurring by coincidence. Figs .19~21 show the UCAV flight paths obtained by GWO testing the second case when 15-*D*, 20-*D*, and 25-*D*. Regarding the paths, the trajectories optimized by GWO are smooth and can avoid the threat areas with the smallest threat cost. As can be observed in Figs. 22~27, the fastest convergence rate belongs to GWO. Through carefully looking at Fig. 28, BSA has a fast convergence initially towards the known minimum; however it is outperformed by FPA and GWO after 160 generations. The significance of the results is also illustrated in Figs. 29~35. From Figs. 30~33, the boxplot of GWO is lower and narrower than other algorithms, showing stability of GWO in converging towards the optimal value. Fig. 29 shows that CS, ABC, and GWO can obtain the relatively stable optimal value, while GWO is lower that means that GWO performs the best at finding the global minimum. Fig. 34 shows that the boxplot of ABC is super narrow. However, GWO is under the minima of other algorithms, which evidences that this superiority can be significant as well. The boxplot of Fig. 35 shows that FPA provides better results.

4.3.3 The third test case

Table 8 Experimental results for the third test case

D	Result	CS	FPA	NBA	BSA	ABC	GGSA	GWO
	Best	72.2827	72.1013	71.3349	72.8285	71.3374	72.8814	**71.2995**
15	Worst	75.6320	74.3944	74.6684	87.5891	72.3384	88.3907	**72.1822**
	Mean	73.2170	73.0011	72.0327	75.6870	71.7678	78.5506	**71.4922**
	Std	0.6851	0.5696	0.7562	2.8695	0.2505	4.5505	**0.1990**
	Best	72.4423	71.9032	71.3515	71.8029	71.3298	71.5115	**71.2018**
20	Worst	74.4146	72.5641	73.2166	77.7591	72.1959	77.0533	**71.7093**
	Mean	73.2528	72.2133	71.8015	73.5237	71.6902	73.9477	**71.4121**
	Std	0.4850	0.1701	0.4015	1.7276	0.2449	1.6757	**0.1234**
	Best	72.6744	71.5425	71.2574	71.3143	71.5727	71.4298	**71.1144**
25	Worst	75.9468	73.3156	74.1028	74.4862	74.0003	74.0945	**71.4513**
	Mean	74.2774	72.0517	71.6912	72.5842	72.6066	72.5835	**71.2093**
	Std	0.8621	0.3998	0.5460	0.7503	0.6105	0.7726	**0.0750**
	Best	71.2459	71.4009	71.2589	71.2429	72.3356	71.2864	**71.0120**
30	Worst	71.7549	72.5036	73.0197	73.1634	74.5319	73.1120	**71.3602**
	Mean	71.4763	71.9642	71.8374	72.3097	73.4402	71.9449	**71.0867**
	Std	0.1256	0.2977	0.4219	0.5300	0.5851	0.4149	**0.0624**
	Best	71.3606	71.5179	71.2993	71.3968	72.5110	71.3896	**70.9831**
35	Worst	71.8622	72.0979	71.9717	72.1467	75.9186	72.0959	**71.3855**
	Mean	71.6167	71.7946	71.6849	71.8480	73.7553	71.6603	**71.2316**
	Std	0.1405	0.1353	0.1958	0.2225	0.7274	0.2010	**0.1131**
	Best	71.6905	71.2440	**71.0485**	71.6047	72.9066	71.0893	71.2200
50	Worst	72.7542	71.8478	73.0375	73.4860	75.5024	71.4890	**71.4566**
	Mean	72.1102	71.4579	71.7320	72.1530	74.2438	**71.2174**	71.3184
	Std	0.2532	0.1512	0.5332	0.3631	0.6759	0.0848	**0.0606**
	Best	72.7186	72.7908	70.9035	71.4671	74.5453	71.2979	**70.8764**
100	Worst	73.6142	74.6903	80.5640	71.7961	78.5836	74.1133	**71.4638**
	Mean	73.2110	73.7700	73.8791	71.6285	75.6024	72.7858	**71.0879**
	Std	0.2515	0.5937	1.7267	**0.0895**	0.9602	0.8635	0.1284

Table 9 *p* values of the Wilcoxon rank-sum test via comparing GWO vs CS, FPA, NBA, BSA, ABC, and GGSA in the third case

GWO vs	CS	FPA	NBA	BSA	ABC	GGSA
$D = 15$	2.2539E-04	3.3384E-11	4.0840E-05	3.0199E-11	1.2493E-05	3.0199E-11
$D = 20$	1.6980E-08	3.0199E-11	4.1127E-07	3.0199E-11	4.4205E-06	5.4941E-11
$D = 25$	6.1210E-10	3.0199E-11	4.1997E-10	4.0772E-11	3.0199E-11	3.3384E-11
$D = 30$	4.9752E-11	3.0199E-11	4.9752E-11	3.3384E-11	3.0199E-11	3.6897E-11
$D = 35$	3.6897E-11	3.0199E-11	9.9186E-11	3.0199E-11	3.0199E-11	3.0199E-11
$D = 50$	3.0199E-11	3.3679E-04	5.2640E-04	3.0199E-11	3.0123E-11	4.1178E-06
$D = 100$	3.0199E-11	3.0199E-11	7.0881E-08	3.0199E-11	2.9935E-11	4.0772E-11

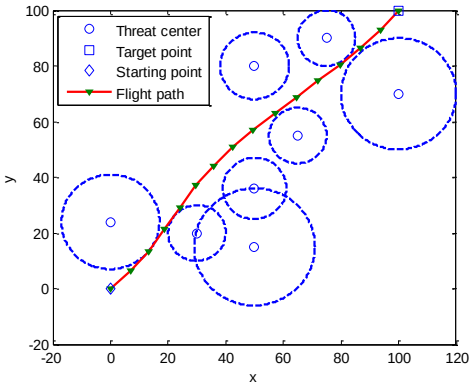

Fig. 36 Path planning result optimized by GWO algorithm in Case 3 (D = 15)

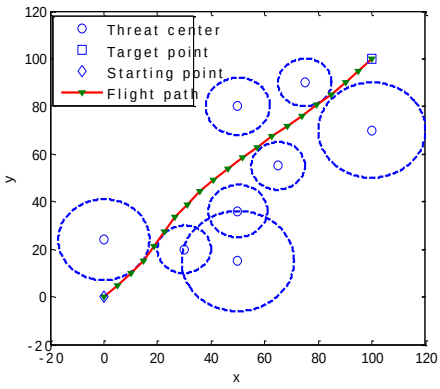

Fig. 37 Path planning result optimized by GWO algorithm in Case 3 (D = 20)

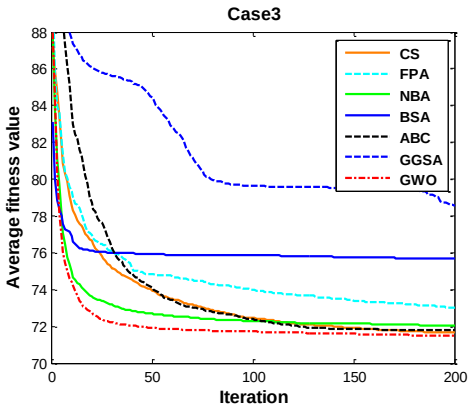

Fig. 38 Comparative convergence curves of algorithms in Case 3 (D = 15)

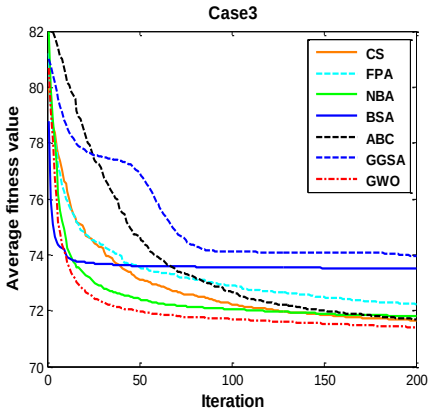

Fig. 39 Comparative convergence curves of algorithms in Case 3 (D = 20)

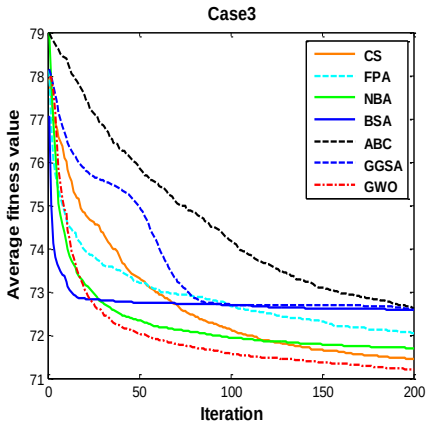

Fig. 40 Comparative convergence curves of algorithms in Case 3 (D = 25)

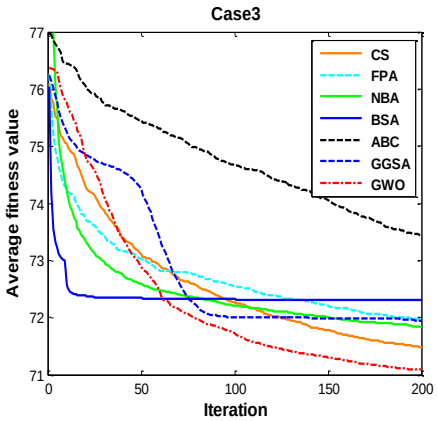

Fig. 41 Comparative convergence curves of algorithms in Case 3 (D = 30)

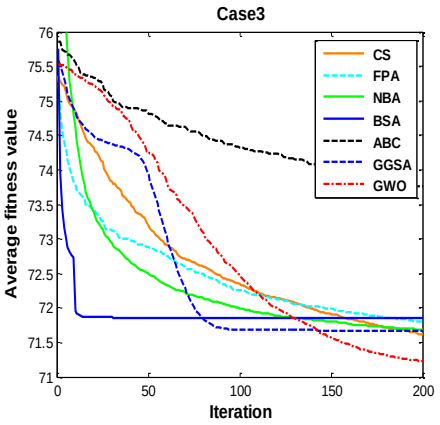

Fig. 42 Comparative convergence curves of algorithms in Case 3 (D = 35)

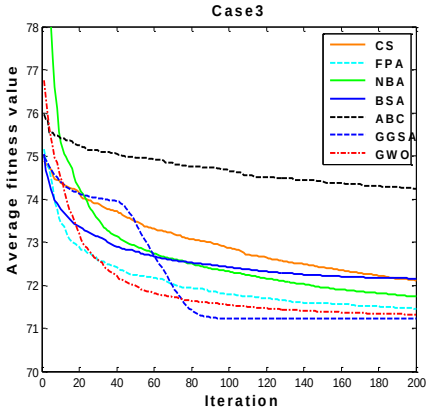

Fig. 43 Comparative convergence curves of algorithms in Case 3 (D = 50)

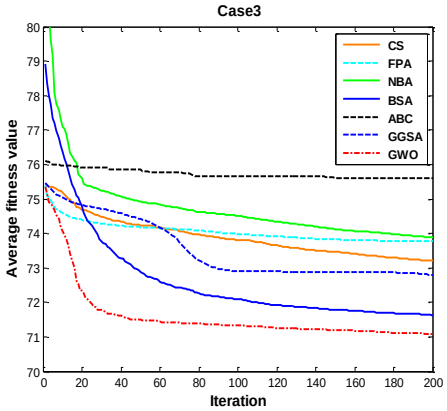

Fig. 44 Comparative convergence curves of algorithms in Case 3 (D = 100)

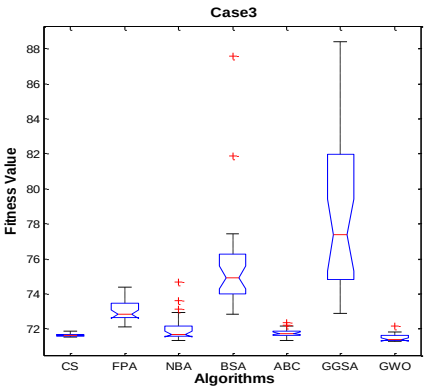

Fig. 45 Comparative ANOVA tests of algorithms in Case 3 (D = 15)

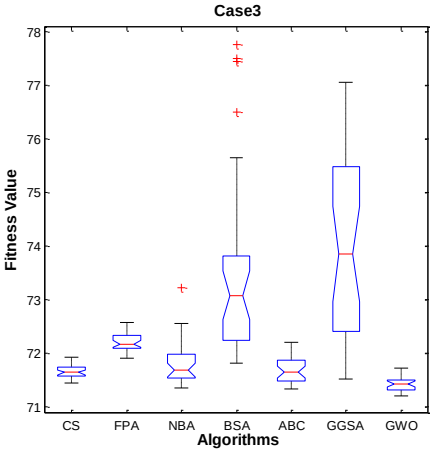

Fig. 46 Comparative ANOVA tests of algorithms in Case 3 (D = 20)

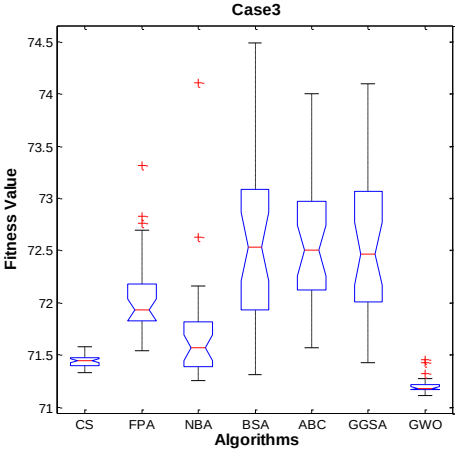

Fig. 47 Comparative ANOVA tests of algorithms in Case 3 (D = 25)

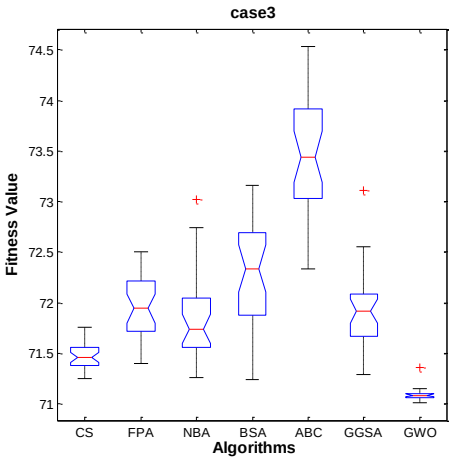

Fig. 48 Comparative ANOVA tests of algorithms in Case 3 (D = 30)

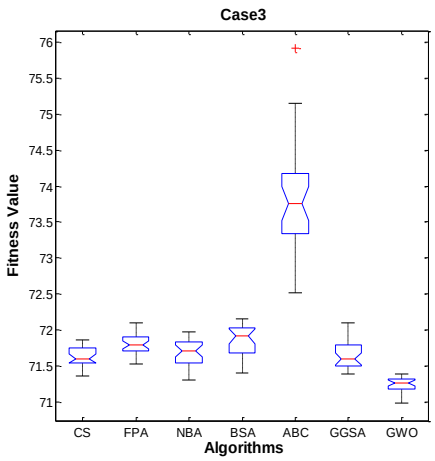

Fig. 49 Comparative ANOVA tests of algorithms in Case 3 (D = 35)

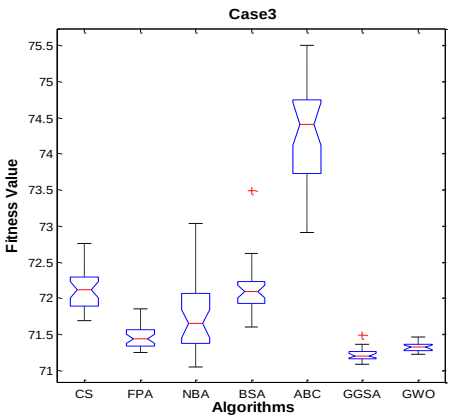

Fig. 50 Comparative ANOVA tests of algorithms in Case 3 (D = 50)

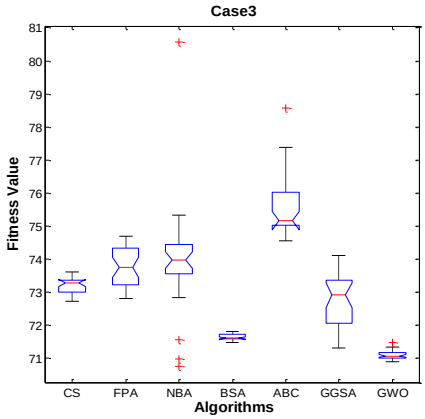

Fig. 51 Comparative ANOVA tests of algorithms in Case 3 (D = 100)

The results of the third case are provided in Table 8. As the results of Best, Worst, Mean, and Std show, GWO provides the best results across the majority of all the dimensions. These results prove that GWO has the capability of avoiding local minima in the third test case. For this case on 50-*D*, we see that NBA and GWO perform better than other algorithms on Best value and Worst value, respectively. The Mean values of GGSA and GWO are close, while Std value shows that GWO is more stable. For this case on 100-*D*, GWO obtains the best results on Best, Worst, and Mean values. And Std value shows that BSA and GWO has better stability. In addition, the *p* values of Wilcoxon's rank-sum in Table 9 show that the results of GWO across all the dimensions are significantly better than the other algorithms. Figs. 36 and 37 show the UCAV flight paths got by GWO testing the third case on 15-*D* and 20-*D*, and it is clear that the obtained flight paths have successfully avoided entering the threat regions. Figs. 38~44 depict the convergence curves of algorithms on different dimensions. Figs. 38~40 and Fig. 44 show that GWO has the fastest convergence rate. For Fig. 41, BSA has a fast convergence initially towards the known minimum; however, it is outperformed by GWO after about 60 iterations.

For Fig. 42, BSA has a fast convergence initially towards the known minimum, as the procedure proceeds GWO gets closer and closer to the minimum, while BSA comes into being premature and traps into the local minimum. Fig. 43 shows the results obtained on 50-*D*. From Fig. 43, we can draw the conclusion that GGSA is significantly superior to other algorithms during the process of optimization. For other algorithms, although slower, GWO eventually finds the global minimum close to GGSA, while CS, FPA, NBA, BSA, and ABC fail to search the global minimum within the limited generations. According to Figs. 45~49, the boxplots for GWO have lower and narrower than those of other algorithms. From Fig. 50, GWO and GGSA show a better stability. From Fig. 51, the boxplot of BSA is supper narrow, which is closed followed by GWO. Through carefully looking at Fig. 51, GWO is lower than other algorithms. This means that this algorithm has superior local optimal avoidance and accuracy simultaneously.

4.4 Uncertainty appraisal analysis

All the parameter optimization are ill-posed and posterior sampling is the way of providing an estimate of the uncertainty based on a finite set of the family of models that fit the observed data within the same tolerance [48-50]. The Markov Chain Monte Carlo (MCMC) method is one of the most important algorithms in the modern statistical calculations [51]. This algorithm provides a useful tool for realistic statistical modeling, and has become very popular for Bayesian computation in complex statistical models. The two usual sampling methods in MCMC are the Gibbs sampler and the M-H Strategies. In this paper, by using the random walk Metropolis-Hastings strategies in the MCMC method, with a bivariate normal distribution proposal we estimate the parameters of the UCAV model. A detailed description of the Markov Chain Monte Carlo method can be found in [51].

The choice of parameter λ all between 0 and 1 gives the designer certain flexibility to dispose relations between the threat exposition degree and the fuel consumption. To further see how statistically significant its quantitative results are, we take parameter λ equal to 0.5 and perform M-H method using 200 sets of experimental data obtained from UCAV model by Matlab. Then we apply iterative computation using M-H method in order to estimate parameter for the known experimental data. Fig. 52 shows the iteration process of parameter λ. Fig. 53 shows the histogram of posterior probability distribution for parameter λ.

The average value of parameter λ is 0.5164 which is obtained through calculating 1000 iteration values in Fig. 52. The posterior probability distribution of parameter λ (Fig. 53) can give valuable information with respect to the parameter λ determination and identification. The average value shows that it is fairly ideal for the estimate of the true value. In addition, expectation of histogram of the parameter λ is also a symmetric one.

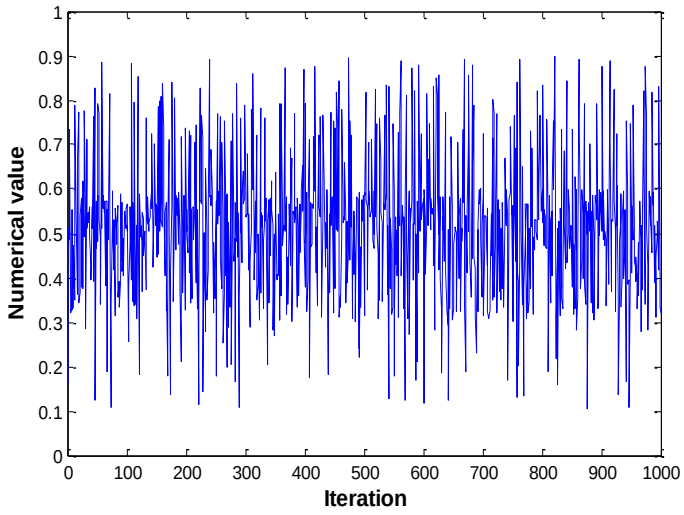

Fig. 52 The iteration process of parameter λ

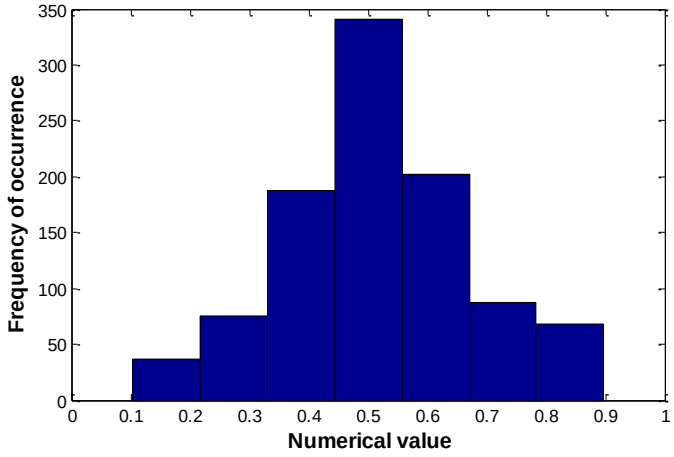

Fig. 53 Histogram of posterior probability distribution for parameter λ

4.5 Discussion and analysis of the results

Statistically speaking, the GWO method has been demonstrated to perform better than or at least competitive with the other six population-based algorithms. The reason for improved Best, Worst, Mean, and Std is due to the high local optima avoidance of this algorithm. According to equations of the GWO, half of the iterations are devoted to exploration of the search space (when $|A|>1$). This promotes exploration of the search space that leads to finding diverse search areas during optimization. Moreover, the C parameter always randomly obliges the search agents to take random steps towards/outwards the prey. This mechanism is very helpful in case of local optima stagnation even when the GWO is in the exploitation phase. The reason why GWO provided the high accuracy of the solutions is because this algorithm is equipped with adaptive parameters to smoothly balance exploration and exploitation. Half of the iteration is devoted to exploration (when $|A|>1$) and the rest to exploitation (when $|A|<1$). In addition, GWO always saves the three best

obtained solutions at any stage of optimization. Therefore, there are always guiding search agents for exploitation of the most promising regions of the search space. In other words, GWO benefits from intrinsic exploitation guides that also assist this algorithm to provide remarkable results. These mechanisms described above assist GWO to provide very good exploration, exploitation, local optima avoidance, and fast convergence simultaneously [15, 31].

Another finding in the results is poor performance of CS, NBA, BSA, ABC, and GGSA. CS, NBA, BSA and ABC algorithms belong to the class of swarm-based algorithms, while GGSA is physics-based algorithm. In comparison with to evolutionary algorithms, there is no mechanism for significant abrupt movements in the search space and this is likely to be the reason for the poor performance of CS, NBA, BSA, ABC, and GGSA. It is also worth discussing the poor performance of the FPA algorithm. In FPA, the interaction or switching of local pollination and global pollination can be controlled by a switch probability $p \in [0,1]$, slightly biased toward local pollination [21]. Local pollination in essence is implemented by "DE/rand/1" mutation strategy. This means that FPA depends on sophisticated mutation operators with a high probability. Consequently, mutation mostly emphasizes exploitation rather than exploration which results in very poor outcomes on the test cases. Although GWO is also a swarm-based algorithm, its mechanisms described in the preceding paragraph are the reasons why it is advantageous in solving UCAV path planning problem.

In summary, it can be concluded that exploration is very important in the problem of UCAV path planning. There is a need to have more random and abrupt search steps (emphasizing exploration) in order to avoid local minima for solving complex path planning problem using population-based algorithms. This study shows that the operators of GWO are highly suitable for solving this problem.

5 Conclusions and future works

In this paper, the recently proposed GWO algorithm was employed to find an optimal path for UCAV two-dimensional path planning problem in difficult combating environments. The high level of exploration and exploitation of this algorithm were the motivations for this study. The proposed algorithm was applied to three test cases. For verification, the results of the GWO were compared to six other stochastic optimization algorithms, such as CS, FPA, NBA, BSA, ABC, and GGSA. The results showed that the proposed method is able to be very effective in UCAV path planning problem. For one, the GWO has very a high level of local optima avoidance, which enhances the probability of finding proper approximations of the optimal weighted sum cost of this path. In addition, the accuracy of the obtained optimal values for weighted sum cost is very high, which is due to the high exploitation of the GWO. This paper also identified and discussed the reasons for strong and poor performances of other algorithms. It was observed that the swarm-based algorithms suffer from low exploration, whereas the GWO does not.

Our future work will focus on the two issues. On the one hand, we will apply GWO to solve the UCAV path planning in three-dimensional space. On the other hand, we will use other recent algorithms such as whale optimization algorithm, ant lion optimizer, virus colony search algorithm, dolphin echolocation, and big bang-big crunch algorithm to solve the same problem.

Acknowledgements

The authors would like to sincerely thank anonymous reviewers, Dr. Xin Chen, Dr. Rui Wang, and Ms. Ying Ling for their valuable and constructive suggestions and encouraging comments. This work is supported by National Science Foundation of China under Grants No. 61563008, 61463007.

References

[1] Zhang Y, Jun Y, Wei G, Wu L (2010) Find multi-objective paths in stochastic networks via chaotic immune PSO. Expert Systems with Applications 37(3): 1911-1919

[2] Roberge V, Tarbouchi M, Labonté G (2013) Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning. IEEE Transactions on Industrial Informatics 9(1): 132-141

[3] Cui Z, Gao X (2012) Theory and applications of swarm intelligence. Neural Computing and Applications 21(2): 205-206

[4] Kennedy J, Eberhart R (1995) Particle swarm optimization, in: Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942–1948

[5] Dorigo M, Maniezzo V, Colorni A (1996) Ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 26(1): 29-41

[6] Mirjalili S, Mirjalili S M, Lewis A (2014) Grey wolf optimizer. Advances in Engineering Software 69:46–61

[7] Mirjalili S, Lewis A (2016) The Whale Optimization Algorithm. Advances in Engineering Software 95: 51-67

[8] Mirjalili S (2015) The ant lion optimizer. Advances in Engineering Software 83: 80-98

[9] Li M D, Zhao H, Weng X W, Han T (2016) A novel nature-inspired algorithm for optimization: Virus colony search. Advances in Engineering Software 92: 65-88

[10] Kaveh A, Farhoudi N (2013) A new optimization method: Dolphin echolocation. Advances in Engineering Software 59: 53-70

[11] Erol O K, Eksin I (2006) A new optimization method: big bang–big crunch. Advances in Engineering Software 37(2): 106-111

[12] Cheng C T, Fallahi K, Leung H, Tse C K (2012) A genetic algorithm-inspired UUV path planner based on dynamic programming. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 42(6): 1128-1134

[13] Zheng C, Li L, Xu F, Sun F, Ding M (2005) Evolutionary route planner for unmanned air vehicles. IEEE Transactions on Robotics 21(4): 609-620

[14] Wang Y X, Chen Z J (1999) Genetic algorithms (GA) based flight path planning with constraints. Journal of Beijing University of Aeronautics and Astronautics 25(3): 355-358

[15] Duan H B, Ma G J, Luo D L (2008) Optimal formation reconfiguration control of multiple UCAVs using improved particle swarm optimization. Journal of Bionic Engineering 5(4): 340-347

[16] Wen Y E, Fan H D (2005) Algorithm for low altitude penetration aircraft path planning with improved ant colony algorithm. Chinese Journal of Aeronautics 18(4): 304-309

[17] Li B, Gong G L, Yang W L (2014) An improved Artificial Bee Colony algorithm based on balance-evolution strategy for unmanned combat aerial vehicle path planning. The Scientific World Journal, vol. 2014, Article ID 232704, 10 pages

[18] Xu C F, Duan H B, Liu F (2010) Chaotic artificial bee colony approach to Uninhabited Combat Air Vehicle (UCAV) path planning. Aerospace Science and Technology 14(8): 535-541

[19] Duan H B, Zhang X Y, Xu CF (2011) Bio-Inspired Computing, Science Press, Beijing, China, 2011

[20] Zhu W R, Duan H B (2014) Chaotic predator–prey biogeography-based optimization approach for UCAV path planning. Aerospace Science and Technology 32(1): 153-161

[21] Wang G, Guo L, Duan H, Liu L, Wang H (2012) A modified firefly algorithm for UCAV path planning. International Journal of Hybrid Information Technology 5(3): 123-144

[22] Wang G, Guo L, Duan H, Liu L, Wang H, Wang B (2012) A hybrid meta-heuristic DE/CS algorithm for UCAV path planning. Journal of Information and Computational Science 5(16): 4811-4818

[23] Wang G, Guo L, Duan H, Liu L, Wang H (2012) A bat algorithm with mutation for UCAV path planning. The Scientific World Journal, pp. 1-15

[24] Zhou Y, Wang R (2016) An Improved Flower Pollination Algorithm for Optimal Unmanned Undersea Vehicle Path Planning Problem. International Journal of Pattern Recognition and Artificial Intelligence. DOI: 10.1142/S0218001416590102

[25] Duan H, Liu S, Wu J (2009) Novel intelligent water drops optimization approach to single UCAV smooth trajectory planning. Aerospace science and technology 13(8): 442-449

[26] Duan H B, Huang L Z (2014) Imperialist competitive algorithm optimized artificial neural networks for UCAV global path planning. Neurocomputing 125: 166–171

[27] Wolpert D H, Macready W G (1997) No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation 1(1): 67-82

[28] Sulaiman M H, Mustaffa Z, Mohamed M R, Aliman O (2015) Using the gray wolf optimizer for solving optimal reactive power dispatch problem. Applied Soft Computing 32:286-292

[29] Song X H, Tang L, Zhao S T, Zhang X Q, Li L, Huang J Q, Cai W (2015) Grey Wolf Optimizer for parameter estimation in surface waves. Soil Dynamics and Earthquake Engineering 75:147-157

[30] Komaki G M, Kayvanfar V (2015) Grey Wolf Optimizer algorithm for the two-stage assembly flow shop scheduling problem with release time. Journal of Computational Science 8:109-120

[31] Medjahed SA, Saadi TA, Benyettou A, Ouali M (2016) Gray Wolf Optimizer for hyperspectral band selection. Applied Soft Computing 40: 178-186

[32] Emary E, Yamany W, Hassanien A E, Snasel V (2015) Multi-Objective Gray-Wolf Optimization for Attribute Reduction. Procedia Computer Science 65: 623-632

[33] Emary E, Zawbaa H M, Hassanien A E (2016) Binary grey wolf optimization approaches for feature selection. Neurocomputing 172: 371-381

[34] Mirjalili S (2015) How effective is the Grey Wolf optimizer in training multi-layer perceptrons. Applied Intelligence 43(1): 150-161

[35] Saremi S, Mirjalili S Z, Mirjalili S M (2015) Evolutionary population dynamics and grey wolf optimizer. Neural Computing and Applications 26(5): 1257-1263

[36] Song H M, Sulaiman M H, Mohamed M R (2014) An application of grey wolf optimizer for solving combined economic emission dispatch problems. International Review on Modelling and Simulations (IREMOS) 7(5): 838-844

[37] Mirjalili S, Saremi S, Mirjalili S M, Coelho L D S (2016) Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. Expert Systems with Applications 47: 106-119

[38] Yang X S, Deb S (2009) Cuckoo search via Lévy flights. In: World congress on nature and biologically inspired computing, Coimbatore, India, pp 210–214

[39] Yang X S (2012) Flower pollination algorithm for global optimization. In Unconventional

computation and natural computation, Springer Berlin Heidelberg, pp 240-249

[40] Meng X B, Gao X Z, Liu Y, Zhang H (2015) A novel bat algorithm with habitat selection and Doppler effect in echoes for optimization. Expert Systems with Applications 42(17): 6350-6364

[41] Meng X B, Gao X Z, Lu L, Liu Y, Zhang H (2015) A new bio-inspired optimisation algorithm: Bird Swarm Algorithm. Journal of Experimental and Theoretical Artificial Intelligence, pp 1-15

[42] Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of global optimization 39(3): 459-471

[43] Mirjalili S, Lewis A (2014) Adaptive gbest-guided gravitational search algorithm. Neural Computing and Applications 25(7-8): 1569-1584

[44] Ye W, Fan H D (2007) Research on mission planning system key techniques of UCAV. Journal of Naval Aeronautical Engineering Institute 22(2): 201-207

[45] Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation 1(1): 3-18

[46] Gibbons JD, Chakraborti S (2011) Nonparametric statistical inference. Springer Berlin Heidelberg, pp 977-979

[47] Hollander M, Wolfe D A, Chicken E (2013) Nonparametric statistical methods. John Wiley and Sons

[48] Song X H, Tang L, Lv X C, Fang H, Gu H M (2012) Application of particle swarm optimization to interpret Rayleigh wave dispersion curves. Journal of Applied Geophysics 84: 1-13

[49] Song X H, Li L, Zhang X Q, Huang J Q, Shi X C, Jin S, Bai Y M (2014) Differential evolution algorithm for nonlinear inversion of high-frequency Rayleigh wave dispersion curves. Journal of Applied Geophysics 109: 47-61

[50] Song X H, Tang L, Lv X C, Fang H, Gu H M (2012) Shuffled complex evolution approach for effective and efficient surface wave analysis.Computers & Geosciences 42: 7-17

[51] Martinez W L, Martinez A R (2007) Computational statistics handbook with MATLAB. CRC press