Three-Dimensional Path Planning for Uninhabited Combat Aerial Vehicle Based on Predator-Prey Pigeon-Inspired Optimization in Dynamic Environment

Bo Zhang and Haibin Duan

Abstract—Three-dimension path planning of uninhabited combat aerial vehicle (UCAV) is a complicated optimal problem, which mainly focused on optimizing the flight route considering the different types of constrains under complex combating environment. A novel predator-prey pigeon-inspired optimization (PPPIO) is proposed to solve the UCAV three-dimension path planning problem in dynamic environment. Pigeon-inspired optimization (PIO) is a new bio-inspired optimization algorithm. In this algorithm, map and compass operator model and landmark operator model are used to search the best result of a function. The prey-predator concept is adopted to improve global best properties and enhance the convergence speed. The characteristics of the optimal path are presented in the form of a cost function. The comparative simulation results show that our proposed PPPIO algorithm is more efficient than the basic PIO, particle swarm optimization (PSO), and different evolution (DE) in solving UCAV three-dimensional path planning problems.

Index Terms—Pigeon-inspired optimization (PIO), uninhabited combat aerial vehicle (UCAV), path planning, predator-prey

1 Introduction

Three-dimensional path planner for uninhabited combat aerial vehicles is an essential element of UCAV autonomous control module [1]. It allows the UCAV to compute an optional or near-optional flight path from a start point to an end point autonomously [2], [3], [4]. This work is motivated by the challenge to develop a practical path planner for UCAVs since many factors are taken in modeling, such as terrain, treat information, fuel consumption and so on [5]. At the same time there are several considerations for an ideal path including optimality, completeness and computational complexity, which are also the most important requirements since path planning has to occur quickly due to fast vehicle dynamics [6].

During the past few decades, extensive work has been devoted to path planning for UCAV. Several approaches have been proposed for UCAV path planning. For example, Eva Besada-Portas presented a path planner for multiple UCAVs based on an evolutionary algorithm (EA) for realistic scenarios [7]. In [8] Duan et al. presented an intelligent water drops (IWD) optimization algorithm for solving the single unmanned aerial vehicle (UAV) path planning problems in various combating environments. Ragi [9] presented a path planning algorithm to guide autonomous UCAVs for tracking multiple ground targets based on the theory of partially observable Markov Decision processes (POMDPs). In [10], a new vibrational genetic algorithm (GA) was developed enhanced with a Voronoi diagram. In [11], Vincent used comparison of parallel genetic algorithm and particle swarm optimization (PSO) for real-time UAV path planning. In [12], Wu presented multi-step A* (MSA*) for multi-objective 4D vehicle motion planning in large dynamic environment. In [13], a hybrid differential evolution (DE) and quantum-behaved particle swarm optimization (DEQPSO) was used to plan the route for UAV on the sea. In [14], fast marching method (FMM) is used to find the visibility space, centroidal voronoi tessellation (CVT) and an improved clustered spiral-alternating algorithm are used to deploy the waypoints then construct the paths alone them. In [15], a variable probability based bidirectional rapidly-exploring random algorithm (VPB-RRT) is applied to the problem of UAV path planning. In [16], a star search algorithm is used to plan the better flight path for civil UAV.

In our paper, the path of a UCAV is defined by a series of nodes and all the considerations for an ideal path is abstracted as a cost function. The problem of building a path planner becomes an optimizing problem of the cost function by choosing the appropriate moving nodes of the path. This kind of problem can always be solved by a swarm intelligence optimizer.

Pigeon-inspired optimization (PIO) is a new population-based swarm intelligence optimizer. Swarm intelligence optimizer is a kind of algorithm that works with a population of particles to propose for optimization problems, such as PSO [17]. PIO is based on the movement of pigeons, which was firstly invented by Duan in 2014 [18]. Pigeons are the

• *The authors are with the State Key Laboratory of Virtual Reality Technology and Systems, School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, P.R., China. E-mail: zhangbo0216@163.com, hbduan@buaa.edu.cn.*

Manuscript received 13 Apr. 2015; revised 31 May 2015; accepted 6 June 2015. Date of publication 17 June 2015; date of current version 2 Feb. 2017. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TCBB.2015.2443789

most popular bird in the world, and they were once used to send the message by Egyptians, which also occurred in many military affairs. Homing pigeons can easily find their homes by using three homing tools: magnetic field, sun and landmarks. In the optimization, map and compass model is presented based on magnetic field and sun, while landmark operator model is presented based on landmarks. PIO algorithm has previously proven itself as a worthy competitor to its better known rivals. However, the basic PIO always converges rather fast and fall into a locally optimum [19].

We introduce the concept of predator-prey to solve the problem above. The dynamical relationship between predators, and their prey is one of the dominant themes in ecology [20]. The concept of predator-prey is introduced to the algorithm. Predators are included in the population of solutions for hunting the worst individuals, and making the other solutions run away from those to avoid the predator.

This novel predator-prey pigeon-inspired optimization (PPPIO) was adopted to solve the three-dimension path planning problem for UAV in our former work, which was firstly presented in "The Fifth International Conference on Swarm Intelligence" [21]. This paper is the extension of that. In this paper, we make the environment more complicated. The danger zones are moving while the UCAV is flying by. A smooth path is essential for a real UCAV, because non-smooth path can't satisfy the turning constraint [6]. In this paper, we used a κ-trajectory [22] to smooth the path. Simulation results, and comparisons verified the feasibility and effectiveness of our proposed algorithm.

The rest of the paper is organized as follows. Section 2 provides the representation and the cost function we developed to evaluate the quality of candidate trajectories. Section 3 describes the principle of basic PIO algorithm. Section 4 shows the implementation procedure of our proposed predator-prey PIO algorithm. Section 5 presents the method we used to smooth the path. Finally, we compare the quality of the trajectories produced by the PIO, PSO, differential evolution and the PPPIO in Section 5.

2 Problem Formulation

2.1 Terrain Environment and Danger Zones

The first step of three-dimensional path planning is to abstract the world space into a representation that will be meaningful to the path planning algorithm. In a real simulation of a UCAV, it is required to use 3D mapping of the environment [4]. In our implementation, we use a formula to indicate the terrain environment. The mathematical function is of the form [24]:

$$\begin{aligned} z(x,y) = {} & \sin(x/5+1) + \sin(y/5) + \\ & \cos(a\sqrt{x^2+y^2}) + \sin(b\sqrt{x^2+y^2}), \end{aligned} \tag{1}$$

where z indicate the altitude of a certain point, and a, b are constants experimentally defined. In reality, UCAV path planning is always under dynamic environment [9], [12], [23]. There are some threatening areas in the mission region, such as radars, missiles, and anti-aircraft artillery. In our paper, all threatening areas are abstracted as cylinders. We present the i-th cylindrical danger zones (or no-fly zones) by two sets of the coordinates: (X_i, Y_i) define the original location of the ranger zone and (V_{xi}, V_{zi}) is the velocity of the danger zone. The moving speed of the danger zone does not change in this paper. The radius r_i indicates the range of the danger zone. It means inside of the cylinder UCAV will be vulnerable to the threat with a certain probability proportional to the distance away from the threat center, while out of which will not be attacked. As shown in the following picture, the blue cylinders define the original locations and the black lines are the motion paths of the danger zones.

2.2 Cost Function

In the situation of UCAV path planning, the optimal path is complex and includes many different characteristics. To take into account these desired characteristics, a cost function is used and the path planning algorithm is to find a path that will minimize the cost function. We define our cost function as follows [11]:

$$\begin{aligned} F_{\text{cost}} = {} & C_{\text{length}} + C_{\text{altitude}} + C_{\text{danger zones}} \\ & + C_{\text{power}} + C_{\text{collision}} + C_{\text{smoothing}}. \end{aligned} \tag{2}$$

In the cost function, the term associated with the length of a path is defined as follows:

$$C_{\text{length}} = 1 - \left(\frac{L_{ST}}{L_{traj}}\right), \tag{3}$$

$$C_{\text{length}} \in [0,1], \tag{4}$$

where L_{ST} is the length of the straight line connecting the starting point S and the end point T and L_{traj} is the total length of the path.

The term associated with the altitude of the path is defined as follows:

$$C_{\text{altitude}} = \frac{A_{\text{traj}} - Z_{\min}}{Z_{\max} - Z_{\min}}, \tag{5}$$

$$\mathrm{C}_{\text{altitude}} \in [0,1], \tag{6}$$

where $Z_{\max}$ is the upper limit of the elevation in our search space, $Z_{\min}$ is the lower limit and A_{traj} is the average altitude of the actual path. $Z_{\max}$ and $Z_{\min}$ are respectively set to be slightly above the highest and lowest point of the terrain.

The term associated with the violation of the danger zones is defined as follows:

$$C_{\text{danger zones}} = \frac{L_{\text{inside d.z.}}}{\sum_{\mathrm{i}=1}^{\mathrm{n}} d_{\mathrm{i}}}, \tag{7}$$

$$\mathrm{C}_{\text{danger zones}} \in [0,1], \tag{8}$$

where n is the total number of danger zones, $L_{\text{inside d.z.}}$ is the total length of the subsections of the path which go through danger zones and d_{i} is the diameter of the danger zone i. Since it is possible for $L_{\text{inside d.z.}}$ to be larger than $\sum_{i=1}^{n} d_i$ in a 3D environment, we just bound $C_{\text{danger zone}}$ to 1 arbitrarily [11]. Since the danger zones are moving during the crossing of UCAV, the location of the danger zones is necessary to define..

The term associated with a required power higher than the available power of the UCAV is defined as follows:

$$C_{\text{power}} = \begin{cases} 0, & L_{\text{not feasible}} > 0 \\ P + \left(\frac{L_{\text{not feasible}}}{L_{\text{traj}}}\right), & L_{\text{not feasible}} > 0, \end{cases} \tag{9}$$

$$C_{\text{power}} \in 0 \cup [P, P+1], \tag{10}$$

where $L_{\text{not feasible}}$ is the sum of the lengths of the line segments forming the path which require more power than the available power of the UCAV, L_{traj} is the total length of the path and P is the penalty constant. In our paper, we use a maximum length of the path to define the power the UCAV can use. This constant must be higher than the cost of the worst feasible path which would have, based on our cost function, a cost of 3. By adding this penalty P, we separate no feasible solutions from the feasible ones.

The term associated with ground collisions is defined as follows:

$$C_{\text{collision}} = \begin{cases} 0, & L_{\text{under terrain}} > 0 \\ P + \left(\frac{L_{\text{under terrain}}}{L_{\text{traj}}}\right), & L_{\text{under terrain}} > 0, \end{cases} \tag{11}$$

$$C_{\text{collision}} \in 0 \cup [P, P+1], \tag{12}$$

where $L_{\text{under terrain}}$ is the total length of the subsections of the path which travels below the ground level, and L_{traj} is the total length of the path.

Because our representation defines a path as a series of line segments, the path generated includes discontinuity in the velocity at the connections of those segments. As will be explained later in Section 5, we use circular constructions to remove those discontinuities in the velocity at the connections of those segments. However, to ensure the smoothing of the smoothing of the optimal solution found is possible, we verify the feasibility of the smoothing in our cost function. Therefore, the term associated with the impossible smoothing of the path is defined as follows:

$$C_{\text{smoothing}} = \begin{cases} 0, & N_{\text{impossible}} = 0 \\ P + \left(\frac{N_{\text{impossible}}}{N_{\text{total}}}\right), & N_{\text{impossible}} > 0, \end{cases} \tag{13}$$

$$C_{\text{smoothing}} \in 0 \cup [P, P+1], \tag{14}$$

where $N_{\text{impossible}}$ is the number of connection that cannot be smoothed using circular constructions as explained in Section 5, and N_{total} is the total number of connections in the path.

The search engine will be adopted to find a solution, which can minimize the cost function during the optimization phase of our path planner algorithm. This can also be explained as to find a path that best satisfies all the qualities represented by this cost function. The cost function demonstrates a specific scenario where the optimal path minimizes the distance travelled the average altitude and avoids danger zones, while respecting the UCAV performance characteristics. This cost function is highly complex and demonstrates the power of our path planning algorithm. However, this cost function could easily be modified and applied to a different scenario.

2.3 Scheme of Path Planning

In our model, the start point and the target are defined as *S* and *T*. Our task is to generate an optional path that has the minimum of the cost function above [5], [25].

We connect point *S* and *T*, and then paint a projection of the line on the XY plane. Divide the projection segment *S′T′* into (*D*+*1*) equal portions. Draw a vertical plane of *S′T′* at each segment point. Take a discrete point at each plane, and connect then in sequence to form a flight path. In this way, the path planning problem is turning into optimizing the coordinate series to achieve a superior fitness value of the objective function.

To accelerate the search speed of the algorithm, we can let line *S′T′* be the x axis and take the coordinate transformation on each discrete point $(x(k), y(k), z(k))$ according to (15) [26]. On which θ is the angle that the original x axis counterclockwise rotate to parallel segment *S′T′*, while (x_s, y_s) represent the coordinates in the original coordinate system. The formula is shown as follows

$$\begin{bmatrix} x'(k) \\ y'(k) \\ z'(k) \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x(k) - x_s \\ y(k) - y_s \\ z(k) \end{bmatrix}. \tag{15}$$

x coordinate of each point can be obtained by a simple formula $x'(k) = \frac{|S'T'|}{D+1} k$ when computing.

3 Principle of Basic PIO

PIO is a novel swam intelligence optimizer for solving global optimization problems [27]. It is based on natural pigeon behavior. Studies show that the species seem to have a system in which signals from magnetite particles are carried from the nose to the brain by the trigeminal nerve [18], [28]. Evidence that the sun is also involved in pigeon navigation has been interpreted, either partly or entirely, in terms of the pigeon's ability to distinguish differences in altitude between the sun at the home base and at the point of release [29]. Recent researches on pigeons' behaviors also show that the pigeon can follow some landmarks, such as main roads, railways and rivers rather than head for their destination directly. The migration of pigeons is summarized as two mathematical models. One is map and compass operator, and the other is landmark operator.

3.1 Map and Compass Operator

In PIO model, virtual pigeons are used. In the map and compass operator, the rules are defined with the position X_i and the velocity V_i of pigeon *i*, and the positions and velocities in a *D*-dimension search space are updated in each iteration. The new position X_i and velocity V_i of pigeon *i* at the *t*h iteration can be calculated with the follows [18]:

$$V_i(t) = V_i(t-1)e^{-Rt} + r_1(X_g - X_i(t-1)), \tag{16}$$

$$X_i(t) = X_i(t-1) + V_i(t), \tag{17}$$

where R is the map and compass factor, r_1 is a random number, and X_g is the current global best position, and which can be obtained by comparing all the positions among all the pigeons.

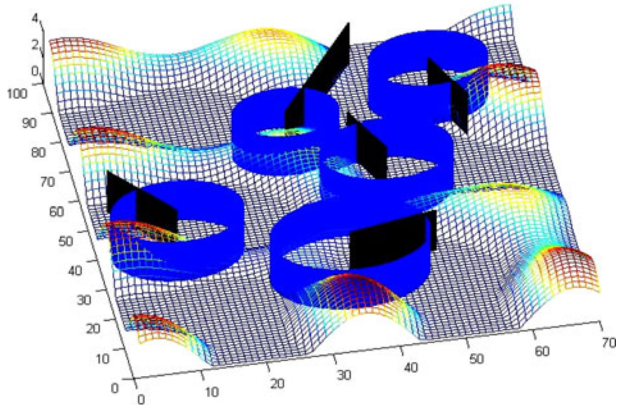
Fig. 1. One example of our 3D dynamic environment for UCAV to fly over.

As shown is Fig. 2, the best position of all pigeons is developing by using map and compass. By comparing all the flied positions, the pigeon on the right is the best one. Each pigeon can adjust its flying direction by following this specific pigeon according to (16), which is expressed by the thick arrows. The thin arrows are its former flying direction, which has relation to$V_i(t-1)\cdot e^{-Rt}$. The vector sum of these two arrows is its next flying direction.

3.2 Landmark Operator

In the landmark operator, half of pigeons is decreased by N_p in every generation. However, the pigeons are still far from the destination, and they are unfamiliar the landmarks. Let X_c be the center of some pigeons' position at the th iteration, and suppose every pigeon can fly straight to the destination. The position updating rule for pigeon i at th iteration can be given by:

$$N_p(t) = \frac{N_p(t-1)}{2}, \tag{18}$$

$$X_c(t) = \frac{\sum_{N_p} X_i(t) fitness(X_i(t))}{\sum_{N_p} fitness(X_i(t))}, \tag{19}$$

$$X_i(t) = X_i(t-1) + r_2(X_c(t) - X_i(t-1)), \tag{20}$$

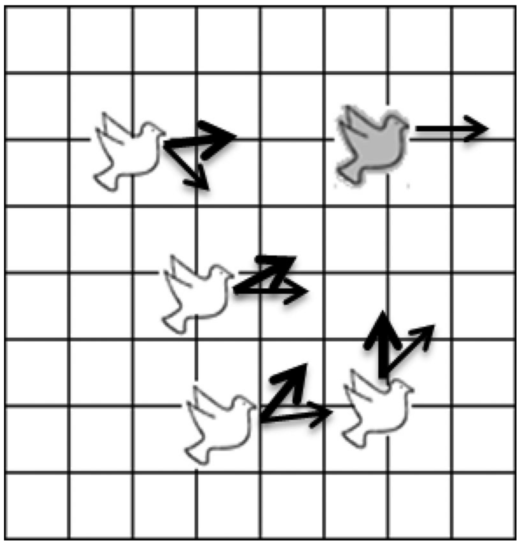
Fig. 2. Map and compass operator model of PIO.

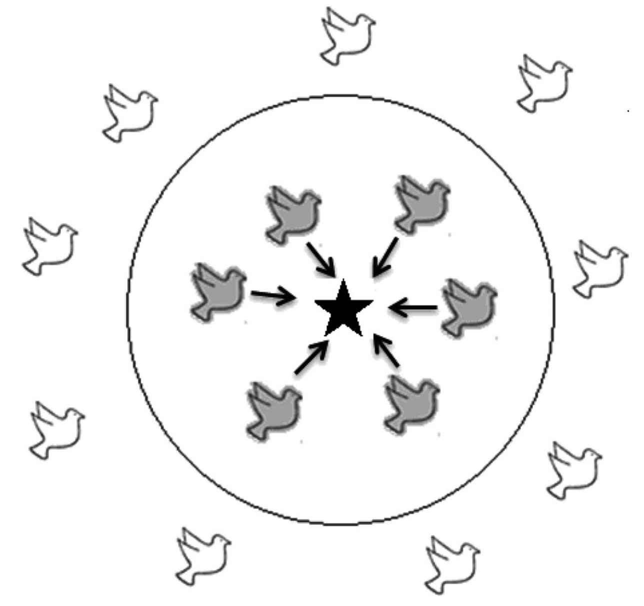
Fig. 3. Landmark operator model.

where r_2 is a random number and *fitness* is the quality of the pigeon individual. For the minimum optimization problems, we can choose the $fitness(X_i(t)) = \frac{1}{f(X_i(t))+\varepsilon}$ for maximum optimization problems, we can choose $fitness$ $(X_i(t)) = f(X_i(t))$.

As shown in Fig. 3, the center of all pigeons (the nest in the center of the circle) is their destination of each iteration. Half of all the pigeons (the pigeons out of the circle) that are far from their destination will follow the pigeon that are close to their diestination, moreover, the pigeons that are close to their destination (the pigeons in the circle) will fly to their destination very quickly.

4 PPPIO FOR THREE-DIMENSIONAL PATH PLANNING

4.1 Predator-Prey Concept

Efficiency and accuracy are the main indexes of the search algorithm [30]. In order to overcome the flaws of the basic PIO algorithm, such as the tendency to converge to local best solutions [31], PPPIO, which integrates PIO with the concept of predator-prey, was proposed in our work. After the mutation of each generation, the predator-prey behavior is been conducted in order to choose better solutions into next generation.

Predatory behavior is one of the most common phenomena in nature, and many optimization algorithms are inspired by the predator-prey strategy from ecology [25], [32], [33]. In nature, predators hunt prey to guarantee their own survival, while the preys need to be able to run away from predators. On the other hand, predators help to control the prey population while creating pressure in the prey population. In this model, an individual in predator population or prey population represents a solution, each prey in the population can expand or get killed by predators based on its fitness value, and a predator always tries to kill preys with least fitness in its neighborhood, which represents removing bad solutions in the population. In this paper, the concept of predator-prey is used to increase the diversity of the population, and the predators are modeled based on the worst solutions which are demonstrated as follows:

$$P_{\text{predator}} = P_{\text{worst}} + \rho\left(1 - \frac{t}{t_{\max}}\right), \tag{21}$$

where P_{predator} is the predator (a possible solution), P_{worst} is the worst solution in the population, t is the current iteration, while $t_{\max}$ is the maximum number of iterations and ρ is the hunting rate. To model the interactions between predator and prey, the solutions to maintain a distance of the prey from the predator is showed as follows:

$$\begin{cases} P_{\text{k}+1} = P_{\text{k}} - \rho e^{-|d|}, & d > 0, \\ P_{\text{k}+1} = P_{\text{k}} - \rho e^{-|d|}, & d < 0, \end{cases} \tag{22}$$

where *d* is the distance between the solution and the predator, and *k* is the current iteration.

In this way, our proposed algorithm takes the advantage of the predator-prey concept to make the individuals of sub generations distributed widely in the defined space, and it can avoid from the premature of the individuals, as well as to increase the speed of finding the optimal solution.

4.2 Parallelization of the Map and Compass Operations and the Landmark Operations

In the model of the basic PIO, the landmark operator of the original PIO is conducted in the last few generations. However, the algorithm could be already convergent thus the landmark operator could not work [34]. Furthermore, half of the number of pigeons is decreased by N_p in every generation on the landmark operator. The population of pigeons is decreased too rapidly according to (18), which would reach to zero after a small amount of iterations. The landmark operator would make only a small impact on the pigeons' position by this way. So we make a small modification on the basic PIO algorithm. The map and compass operation and the compass operation are used parallelly at all iterations. A parameter ω is used to define the impaction of the landmark. The parameter increases with a smoothly path. And a constant parameter *c* is used to define the number of pigeons that are in the landmark operator. The new formula of landmark operator is as follows:

$$N_p(\text{t}) = c \bullet N_{P\max} \qquad c \in (0,1), \tag{23}$$

$$X_c(t) = \frac{\sum_{N_p} X_i(t) fitness(X_i(t))}{\sum_{N_p} fitness(X_i(t))}, \tag{24}$$

$$\omega = \frac{s + (1-s)t}{N_{c\max}} \qquad s \in (0,1), \tag{25}$$

$$X_i(t) = X_i(t-1) + r_2\omega(X_c(t) - X_i(t-1)), \tag{26}$$

where *s* is a constant experimentally defined.

4.3 Procedures of PPPIO for Path Planning

The implementation procedure of our proposed PIO approach to UCAV path planning can be described as follows:

Step 1: According to the environmental modeling in Section 2, initialize the detailed information about the path planning task.

Step 2: Initialize the PIO parameters, such as solution space dimension D, the population sizeN_p, map and compass factor R, the number of iteration N_c.

Step 3: Set each pigeon with a randomized velocity and path. Compare the fitness of each pigeons, and find the current best path.

Step 4: Operate map and compass operator. Firstly, we update the velocity and path of every pigeon by using (16) and (17).

Step 5: Rank all pigeons according their fitness values. Some of pigeons whose fitness values are low will follow those pigeons with high fitness according to (23). We then find the center of all pigeons according to (24), and this center is the desirable destination. All pigeons will fly to the destination by adjusting their flying direction according to (26). Next, store the best solution parameters and the best cost value.

Step 6: Model the predators based on the worst solution as (15) demonstrates. Then, use (16) to provide the other solutions to maintain a distance between the predator and the prey.

Step 7: If $N_c > N_{c\max}$, stop the iteration, and output the results. If not, go to step 6.

4.4 Convergence and Complexity Analysis of the PPPIO Algorithm

Without considering the impact of the predator-prey concept, and the random number on the formulas the PPPIO algorithm is defined by (17) (23) (24) (25) and

$$V_i(t) = V_i(t-1)e^{-Rt} + c(X_g - X_i(t-1)), \tag{27}$$

$$X_i(t) = X_i(t-1) + \omega \bullet (X_c - X_i(t-1)). \tag{28}$$

Definition 1. *Assuming that* $\{y^{(k)}, k \geq 0\}$ *is a discrete random variable; the finite set of the discrete random variables value is* $S = \{j\}$ *called state space. If for any* $k \geq 1$, $i^{(l)} \in S(l \leq k+1)$, *satisfying (27),*$\{y^{(k)}, k \geq 0\}$ *is Markov Chain*

$$\begin{aligned} p(y(t+1) = i(t+1)|y(t) = i(t), \ldots, y(0) = i(0)) \\ = p(y(t+1) = i(t+1)|y(t) = i(t)). \end{aligned} \tag{29}$$

Lemma 1. *The state of population sequence of the PIO algorithm* $X(t)$ *is a finite Markov chain.*

Proof. The solution space S a defined bound. Moreover, the parameters and the size of the swarm of the pigeons are determined, thus, the state space for the pigeon swarm is finite. □

From the description of the algorithm above we can obtain that the value of $X(t)$ is a list of the discrete random variable, and the value of $X(t+1)$ is only determined by the former moment state $X(t)$. We can declare that the state of the pigeon swarm $X(t)$ is a finite homogeneous Markov chain.

Definition 2. *The global optimal solution sets can be described as* $M = \{X; \forall Y \in S, s.t. f(X) \geq f(Y)\}$.

Definition 3. *For any initial position distributions of pigeons $X(0) = S_0 \in S$, $\lim_{n\to\infty} P\{X(t) \in M | X(0) = S_0\} = 1$ means that the algorithm strongly converges to the global optimal solution sets in probability, while $\lim_{n\to\infty} P\{X(t) \cap M \neq 0 | X(0) = S_0\} = 1$ indicates a weak convergence to the global optimal solution sets.*

Lemma 2. *The evolution direction of the swarm of PIO algorithm is unchangeable,* i.e.$f(X(t+1)) \leq f(X(t))$.

Proof. It is obvious that the PIO algorithm tends to make the individual move toward the position with lower cost function. As a result, the lowest cost function can monotonically decrease with generations. □

Theorem 1. *The Markov chain of the PIO algorithm converges to the global optimal solution sets M in probability 1,* i.e. $\lim_{t\to\infty} P\{X(t) \in M\} = 1$.

Proof. From Lemma 1, it is obvious that if $X(t)$ has already entered the global optimal solution sets M, $X(t+1)$ is bound to be concluded in M

$$P\{X(t+1) \in M | X(t) \in M\} = 1.$$

□

Then

$$\begin{aligned} P\{X(t+1) \in M\} &= (1 - P\{X(t) \in M\})\bullet P\{X(t+1) \notin M\} \\ &\quad + P\{X(t) \in M\}\bullet P\{X(t+1) \in M | X(t) \in M\} \\ &= (1 - P\{X(t) \in M\})\bullet P\{X(t+1) \notin M\} \\ &\quad + P\{X(t) \in M\}. \end{aligned}$$

Suppose

$$P\{X(t+1) \in M | X(t) \notin M\} \geq d(t) \geq 0.$$

Then

$$\lim_{n\to\infty} \prod_{i=1}^{n} 1 - d(i) = 0.$$

Thus

$$\begin{aligned} &1 - P\{X(t+1) \in M\} \leq [1 - d(t)][1 - p\{X(t) \in M\}] \\ &\quad \Rightarrow 1 - P\{X(t+1) \in M\} \\ &\quad \leq [1 - p\{X(0) \in M\}]\bullet \prod_{i=1}^{n} 1 - d(i) \\ &\quad \Rightarrow \lim_{n\to\infty} P\{X(t+1) \in M\} \geq \\ &1 - [1 - P\{X(0) \in M\}]\bullet \lim_{n\to\infty} \prod_{i=1}^{n} 1 - d(i) \\ &\quad \Rightarrow \lim_{n\to\infty} P\{X(t+1) \in M\} \geq 1. \end{aligned}$$

However, $P\{X(t+1) \in M\} \leq 1$, as a result, $\lim_{n\to\infty} P\{X(t+1) \in M\} = 1$. It is equal to that $\lim_{n\to\infty} P\{X(t) \in M\} = 1$

It is obvious that the PPPIO algorithm excluding predator-prey concept can converge to the global optimal solution sets M in probability 1. If taking predator-prey concept into consideration, the algorithm's ability to jump out of local optimum can be increased, and accordingly, the algorithm can converge to the global optimized solution set M.

From the mathematical description of the PPPIO algorithm, we can define the computation complexity of the algorithm. Time complexity of the map and compass operator on one generation is $O(DN_p)$ because we need to use (16) (17) to update every dimensionality of every pigeon. The landmark operator's time complexity on one generation is $O(DN_p + N_p \log N_p)$. The quick sort is used to divide the part of the pigeons in order to calculate X_c. The time complexity of predator-prey concept is $O(DN_p)$ which is same as map compass operator.

Since the number of iterations is N_c, we can sum them up and find out the computation complexity of the algorithm which is $O((DN_p + N_p \log N_p)N_c)$.

5 Path Smoothing

Our solution generates a path composed of line segments. This may be sufficient for multidirectional ground robots, but improper for fixed-wing UCAVs [11], [35]. We adopt a class of dynamically feasible path smooth strategy called κ-trajectories [6], [22]. Consider the turning point path defined by the three waypoints w_{i-1}, w_i and w_{i+1} let

$$q_i = (w_i - w_{i-1}) / \| w_i - w_{i-1} \| \tag{30}$$

$$q_{i+1} = (w_{i+1} - w_i) / \| w_{i+1} - w_i \| \tag{31}$$

denote the unit vectors along the corresponding path segments, which can be shown in Fig. 5.

Let β represent the angle between q_i and q_{i+1}, then we can get $\beta = \arccos(-q_{i+1} \cdot q_i)$. As shown in Fig. 5, let $\overline{C}$ be a circle of radius

$$R = 0.5\min\{\|w_i - w_{i-1}\|, \|w_{i+1} - w_i\|\} \tan \frac{\beta}{2}, \tag{32}$$

where center C_i lies on the bisector of the angle formed by the three waypoints, such that the circle intersects both the lines $\overline{w_{i-1}w_i}$ and $\overline{w_i w_{i+1}}$ at exactly one point each. The bisector of β will intersect $\overline{C}$ at two locations. So the center C_i is given by

$$C_i = w_i + \left(R \Big/ \sin \frac{\beta}{2}\right)(q_{i+1} - q_i) / \|q_{i+1} - q_i\| \tag{33}$$

After this process, the original path $\overline{w_{i-1}w_i} \rightarrow \overline{w_i w_{i+1}}$ could be replaced by $\overline{w_{i-1}A} \rightarrow \widehat{A}\ \widehat{C}\widehat{B} \rightarrow \overline{Bw_{i+1}}$. In this way, the optimized path can be smoothed for feasible flying. This path smoothing algorithm has a small computational load and can be run in real-time.

Although smoothing of the path is performed after the optimization process, the feasibility of this operation is verified in our cost function to ensure the smoothing of the final path is always possible. The result of path smoothing is shown in Fig. 6, the connecting points are replaced by yellow circular arcs.

6 Comparative Experimental Results

In order to evaluate the performance of our proposed PPPIO algorithm in this work, series of experiments are conducted in Matlab 2012a programming environment on a PC with 2.0 GHz CPU. Coordinates of a starting point are set as (10, 16, 0), and the target point as (55, 100, 0).

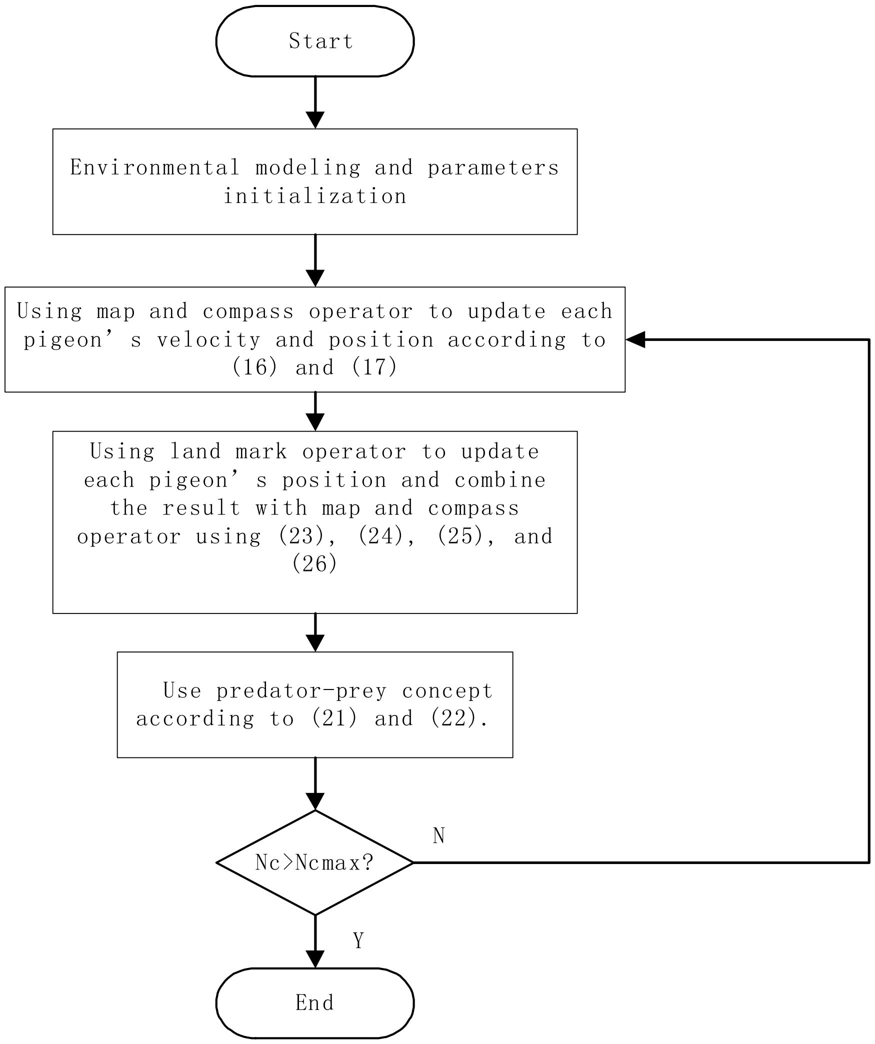
Start
Environmental modeling and parameters initialization
Using map and compass operator to update each pigeon' s velocity and position according to (16) and (17)
Using land mark operator to update each pigeon' s position and combine the result with map and compass operator using (23), (24), (25), and (26)
Use predator-prey concept according to (21) and (22).
Nc>Ncmax?
N
Y
End

Fig. 4. The procedure of PPPIO.

The initial parameters of both classical PIO algorithm and PPPIO algorithm were adjusted as:

Population size: $N_p = 150$;

The map and compass factor: $R = 0.2$;

Number of pigeons that are in the landmark operator: $c = 0.5$;

Increasing rate factor of landmark operator: $s = 0.2$;

Number of segment points: $D = 10$;

The comparative results of PPPIO with PIO, PSO and DE are showed as follows. We create three different environments to test the algorithm. The case 1 is shown in Figs. 7, 8 and 9, on which every danger zones are moving on a constant velocity, while in Figs. 10, 11 and 12, which is case 2, some of the danger zones are moving while one is fixed which is more approximate to the real environment. Case 3 is a stable environment which is shown in Figs. 13, 14 and 15.

From evolution curves of four algorithms, it shows PIO sometimes converged faster than PPIO. This phenomenon actually shows the effect of predator-prey concept, because the concept aims to increase the diversity of the population and prevent the population getting into local minimum

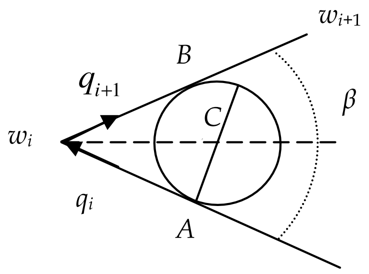

Fig. 5. Circular constructions used to smooth the final path and remove all discontinuity in the velocity.

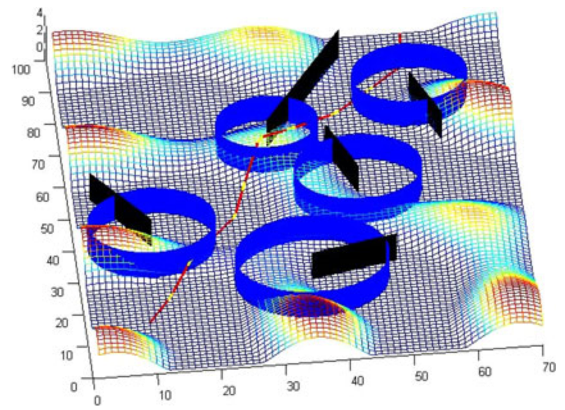

Fig. 6. Result of path smoothing.

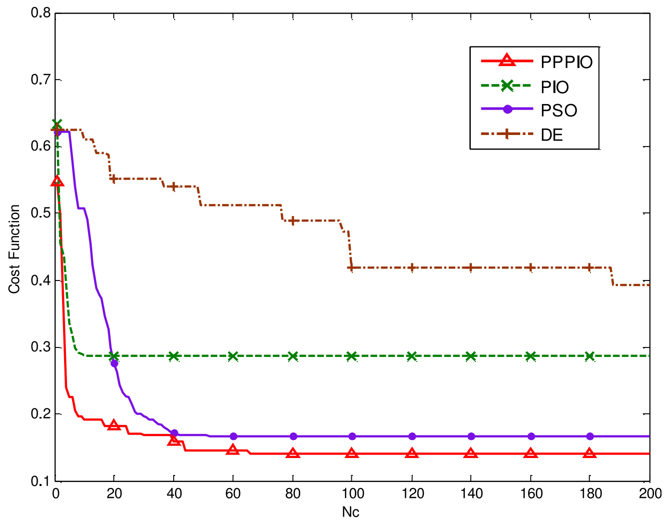

Fig. 7. Comparative evolutionary curves of PPPIO, PIO, PSO, and DE in dynamic environment on the case 1.

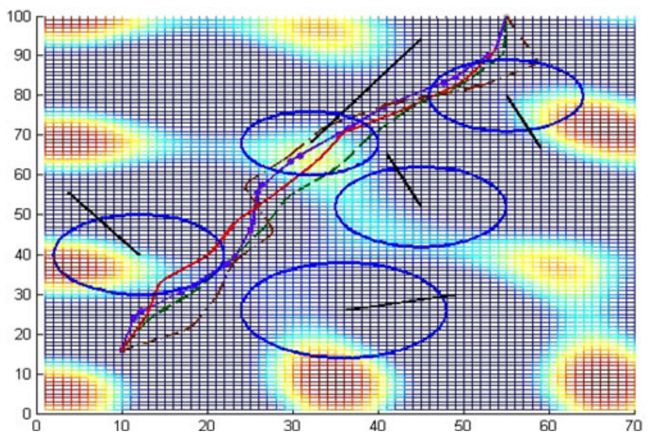

Fig. 8. Comparative path planning results of PPPIO, PIO, PSO, and DE on the case 1.

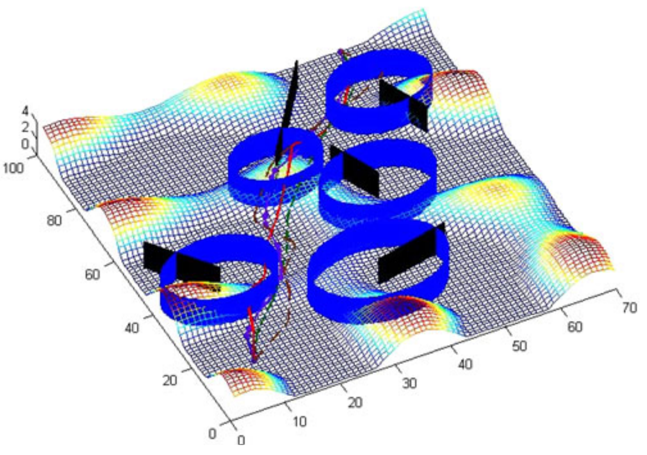

Fig. 9. Comparative path planning results of PPPIO, PIO, PSO, and DE in dynamic environment on 3D version on the case 1.

solution, thus it reduces the convergence speed in some degree. However, the final result of PPPIO is better than PIO as predator-prey concept really help the optimization process, especially in the last phase.

We have also made some analyses about parameters of the proposed algorithm:

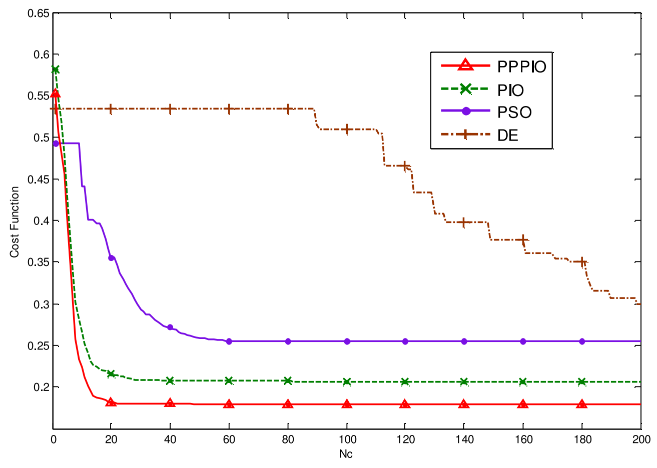

Fig. 10. Comparative evolutionary curves of PPPIO, PIO, PSO, and DE on the case 2.

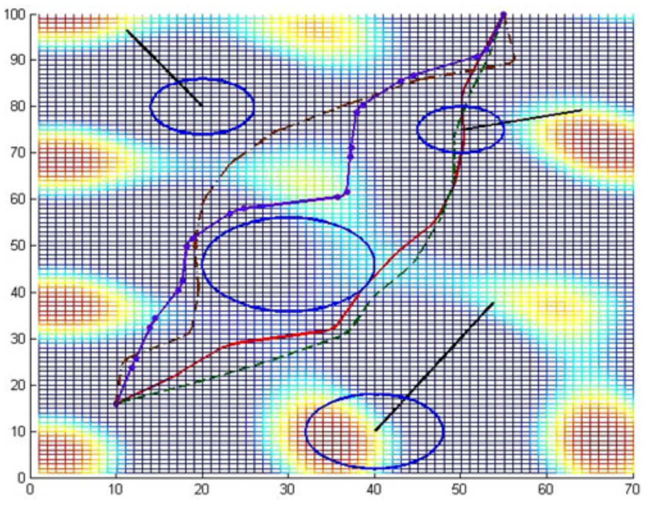

Fig. 11. Comparative path planning results of PPPIO, PIO, PSO, and DE in dynamic environment on the case 2.

1) Population size N_P: N_P is an important parameter in PIO algorithm. Increase of N_P would enlarge the search space and result in a diversity of solution. However, if N_P is too large, it will result in a slowly converge process and a large amount of computational effort is needed. On the other hand, a too small N_P would lead to a local best solution. After a number of simulation under different conditions ($N_P = 50$, 100, 150, 200, 300), we finally decide $N_P = 50$ as it is suitable for the path planning problem in this paper.
2) The map and compass factor$R$:R means how the velocity before ($V_i(t-1)$) influence the next velocity ($V_i(t)$) in the map and compass operator. A too large-Rwill lead to the weight $V_i(t-1)$ decrease too fast. After numbers of simulation, we choose$R$ to be 0.2 which is suitable for this problem.
3) Number of segment points D: A larger number of nodes on the path can lead to a more precise path. However, too many nodes will enlarge the dimension of the searching space, and will take a lot of time for computation. After checking several papers on UCAV path planning [5], [22], [25] and simulation, we choose 10 as the value of D.

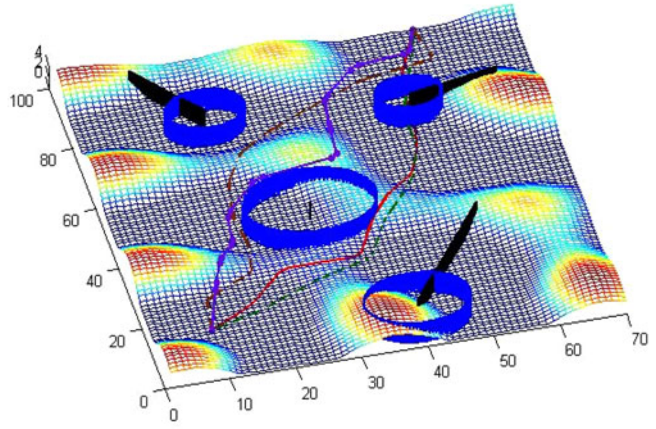
Fig. 12. Comparative path planning results of PPPIO, PIO, PSO, and DE in dynamic environment on 3D version on the case 2.

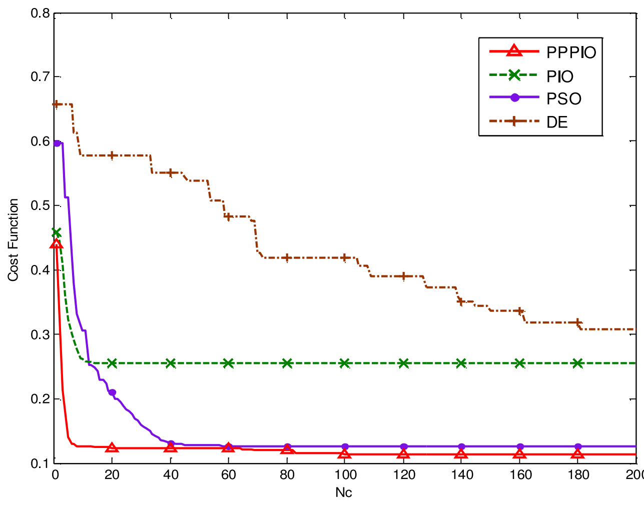

Fig. 13. Comparative evolutionary curves of PPPIO, PIO, PSO, and DE in constant environment on the case 3.

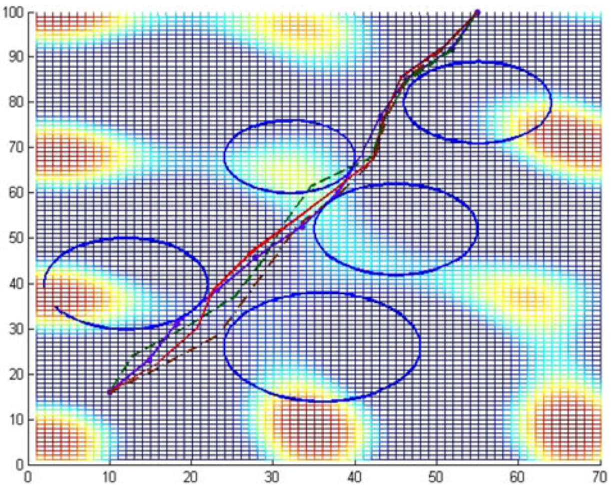
Fig. 14. Comparative path planning results of PPPIO, PIO, PSO, and DE in constant environment on the case 3.

In the environment of Figs. 8 and 9, every danger zones are moving on a constant velocity. The blue cylinders define the initial locations of danger zones. The moving path the zones are indicated by the black bars. From the evolutionary

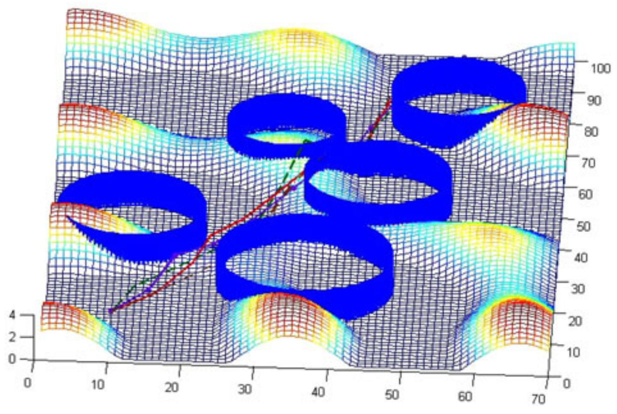
Fig. 15. Comparative path planning results of PPPIO, PIO, PSO, and DE in constant environment on 3D version on the case 3.

TABLE 1
The Comparison Results of PPPIO, PIO, PSO, and DE

Cost Function	PPPIO	PIO	PSO	DE
Case 1	0.1411	0.2866	0.1670	0.3933
Case 2	0.1789	0.2068	0.2548	0.2995
Case 3	0.1146	0.2547	0.1267	0.3072

curves we can figure out that PPPIO algorithm act the best both on efficiency and converging speed. Original PIO algorithm convergences as fast as PPPIO at the initio several iterations, but falls into a local optimum soon.

Figs. 11 and 12 show an environment that has both moving and still danger zones. The converging speed and the final result of PPPIO are better than the original PIO.

The algorithm is also adapted in the environment that every danger zone is maintaining on a fixed position. This model makes sense when a UCAV attempt to fly over an area with radar only. The UCAV even fly into a danger zone using the original PIO algorithm. Our optimized PPPIO algorithm still displays the best in this test case.

The values of each optimal solution searched by the different algorithms could be given by the values of the result of the cost function, which can be shown in Table 1.

From the experimental results above, we can clearly see that using PPPIO will lead to a better result and is more efficient than other algorithms.

7 Conclusions

This paper focused on a novel PPPIO algorithm for solving the UCAV three-dimensional path planning problem in complex dynamic environment. An improved type of PIO model has been formulated for path planning, and the concept of predator-prey is applied to improve the capability of finding satisfactory solutions and increasing the diversity of the populations. Then the UCAV can find the safe path by connecting the chosen nodes of the three-dimensional mesh with minimum cost function. In order to make the optimized UCAV path more feasible, the κ-path is adopted for smoothing the optimized path, which can be run in real-time. Series of comparative simulation results were given to show that our presented PPPIO algorithm is faster than

PSO and DE while the result of cost function is lower than basic PIO in solving UCAV three-dimensional path planning problems in dynamic environment.

Our future work will focus on applying this newly presented PPPIO algorithm to solve the UCAV path planning problem in more complex situations. Furthermore, we will also try to use the PPPIO algorithm to solve other complicated optimization problems.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (Nos. 61425008, 61333004, and 61273054), Top-Notch Young Talents Program of China, and Aeronautical Foundation of China (No. 20135851042).

References

[1] H. Chen, X. M. Wang and Y. Li, "A survey of autonomous control for UAV," in *Proc. IEEE Int. Conf. Artif. Intell. Comput. Intell.*, Shanghai, China, Nov. 2009, vol. 2, pp. 267—271.

[2] H. B. Duan and P. Li, *Bio-Inspired Computation in Unmanned Aerial Vehicles*. Berlin, Germany: Springer-Verlag, 2014.

[3] H. B. Duan, Q. N. Luo, G. J. Ma and Y. H. Shi, "Hybrid particle swarm optimization and genetic algorithm for multi-UAVs formation reconfiguration," *IEEE Comput. Intell. Mag.*, vol. 8, no. 3, pp. 16–27, Aug. 2013.

[4] J. Ni, W. Wu, J. Shen, X. Fan, "An improved VFF approach for robot path planning in unknown and dynamic environments," *Math. Problems Eng.*, vol. 2014, p. 10, 2014.

[5] P. Li and H. B. Duan, "Path planning of unmanned aerial vehicle based on improved gravitational search algorithm," *Sci. China Technological Sci.*, vol. 55, no. 10, pp. 2712–2719, 2012.

[6] H. B. Duan, Y. Yu, X. Zhang and S. Shao, "Three-dimension path planning for UCAV using hybrid meta-heuristic ACO-DE algorithm," *Simulation Model. Practice Theory*, vol. 18, no. 8, pp.1104–1115, 2010.

[7] E. Besada-Portas, L. de la Torre, J. M. de la Cruz and B. de Andrés-Toro, "Evolutionary trajectory planner for multiple UAVs in realistic scenarios," *IEEE Trans. Robot.*, vol. 26, no. 4, pp. 619–634, Aug. 2010.

[8] H. B. Duan, S. Liu, and J. Wu, "Novel intelligent water drops optimization approach to single UCAV smooth trajectory planning," *Aerospace Sci. Technol.*, vol. 13, no. 8, pp. 442–449, 2009.

[9] S. Ragi and E. K. Chong, "UAV path planning in a dynamic environment via partially observable Markov decision process," *IEEE Trans. Aerospace Electron. Syst.*, vol. 49, no. 4, pp. 2397–2412, Oct. 2013.

[10] Y. V. Pehlivanoglu, "A new vibrational genetic algorithm enhanced with a voronoi diagram for path planning of autonomous UAV," *Aerospace Sci. Technol.*, vol. 16, no. 1, pp. 47–55, 2012.

[11] R. Vincent, T. Mohammed, and L. Gilles, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 132–141, Feb. 2013.

[12] P. Y. Wu, D. Campbell and T. Merz, "Multi-objective four-dimensional vehicle motion planning in large dynamic environments," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 3, pp. 621–634, Jun. 2011.

[13] Y. Fu, M. Ding, C. Zhou and H. Hu, "Route planning for unmanned aerial vehicle (UAV) on the sea using hybrid differential evolution and quantum-behaved particle swarm optimization," *IEEE Trans. Systems, Man, Cybern.: Syst.*, vol. 43, no. 6, pp. 1451–1465, 2013.

[14] V. Govindaraju, G. Leng, and Q. Zhang, "Visibility-based UAV path planning for surveillance in cluttered environments," in *Proc. IEEE Int. Symp. Safety, Security, Rescue Robot.*, 2014, pp. 1–6.

[15] X. G. Wu, C. Guo, and Y. B. Li, "Variable probability based bidirectional RRT algorithm for UAV path planning," in *Proc. 26th Chin. Control Decision Conf.*, pp. 2217–2222, May. 2014.

[16] F. H. Tseng, T. T. Liang, C. H. Lee, L. D. Chou, and H. C. Chao, "A star search algorithm for Civil UAV path planning with 3G communication," in *Proc. 10th Int. Conf. Intell. Inf. Hiding Multimedia Signal Process.*, Aug. 2014, pp. 942–945.

[17] Q. Cai, M. Gong, L. Ma, S. Ruan, F. Yuan, and L. Jiao, "Greedy discrete particle swarm optimization for large-scale social network clustering," *Inf. Sci.*, vol. 316, pp. 503–516, Sep. 2014.

[18] H. B. Duan and P. X. Qiao, "Pigeon-inspired optimization: A new swarm intelligence optimizer for air robot path planning," *Int. J. Intell. Comput. Cybern.*, vol. 7, no. 1, pp. 24–37, 2014.

[19] S. J. Zhang, and H. B. Duan, "Gaussian pigeon-inspired optimization approach to orbital spacecraft formation reconfiguration," *Chin. J. Aeronautics*, vol. 28, no. 1, pp. 200–205, 2015.

[20] A. A. Berryman, "The origins and evolution of predator-prey theory," *Ecology*, vol. 73, pp. 1530–1535, 1992.

[21] B. Zhang and H. B. Duan, "Predator-prey pigeon-inspired optimization for UAV three-dimensional path planning," in *Proc. 5th Int. Conf. Adv. Swarm Intell., Part II*, Hefei, China, 2014, pp. 96–105.

[22] E. P. Anderson, R. W. Beard and T. W. McLain, "Real-time dynamic trajectory smoothing for unmanned air vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 13, no. 3, pp. 471–477, May 2005.

[23] X. Chen and J. Zhang, "The three-dimension path planning of UAV based on improved artificial potential field in dynamic environment," in *Proc. IEEE 5th Int. Conf. Intell. Human-Mach. Syst. Cybern.*, Hangzhou, China, Aug. 2013, vol. 2, pp. 144–147.

[24] I. K. Nikolos and A. N. Brintaki, "Coordinated UAV path planning using differential evolution," in *Proc IEEE Symp. Intell. Control, Mediterranean Conf. Control Autom.*, Limassol, Cyprus, Jun. 2005, pp. 549–556.

[25] W. R. Zhu and H. B. Duan, "Chaotic predator-prey biogeography-based optimization approach for UCAV Path planning.," *Aerospace Sci. Technol.*, vol. 32, no. 1, pp. 153–161, 2014

[26] B. Li, L. Gong and W. Yang, "An improved artificial bee colony algorithm based on balance-evolution strategy for unmanned combat aerial vehicle path planning," *Sci. World J*, vol. 2014, p. 10, 2014

[27] R. Dou, and H. B. Duan, "Pigeon-inspired optimization approach to model prediction control for unmanned air vehicle," *Aircraft Eng. Aerospace Technol.*, vol. 87, in press, 2015, Doi:10.1108/AEAT-05-2014-0073.R2.

[28] C. V. Mora, M. Davison, J. M. Wild and M. M. Walker, "Magnetoreception and its trigeminal mediation in the homing pigeon," *Nature*, vol. 432, no. 7016, pp. 508–511, 2004.

[29] A. Whiten, "Operant study of sun altitude and pigeon navigation," *Nature*, vol. 237, pp. 405–406, 1972.

[30] Y. Wu, and X. J. Qu, "Path planning for taxi of carrier aircraft launching," *Sci. China Technological Sci.*, vol. 56, no. 6, pp. 1561–1570, 2013.

[31] H. B. Duan, H. X. Qiu, and Y. M. Fan, "Unmanned aerial vehicle close formation cooperative control based on predatory escaping pigeon-inspired optimization," *SCIENTIA SINICA Technologica*, vol. 45, no. 6, pp. 559–572, 2015.

[32] M. D. A. Costa e Silva, L. D. S. Coelho and L. Lebensztajn, "Multiobjective biogeography-based optimization based on predator-prey approach," *IEEE Trans. Magn.*, vol. 48, no. 2, pp. 951–954, Feb. 2012.

[33] H. B. Duan, Y. Yu and Z. Zhao, "Parameters identification of UCAV flight control system based on predator-prey particle swarm optimization," *Sci. China Inf. Sci.*, vol. 56, no. 1, pp. 1–12, 2013.

[34] C. Li and H. B. Duan, "Target detection approach for UAVs via improved pigeon-inspired optimization and edge potential function," *Aerospace Sci. Technol.*, vol. 39, pp. 352–360, 2014.

[35] C. L. Bottasso, D. Leonello and B. Savini, "Path planning for autonomous vehicles by trajectory smoothing using motion primitives," *IEEE Trans. Control Syst. Technol.*, vol. 16, no. 6, pp. 1152–1168, Nov. 2008.

Bo Zhang is currently a student with the School of Automation Science and Electrical Engineering, Beihang University (formerly Beijing University of Aeronautics and Astronautics, BUAA). He is a member of BUAA Bio-inspired Autonomous Flight Systems (BAFS) Research Group. He received the ICSI Best Student Paper Award in 2014. His research interests include bioinpsired computing, and multiple UAVs autonomous formation control.

Haibin Duan (M'07-SM'08) received the PhD degree from Nanjing University of Aeronautics and Astronautics (NUAA) in 2005. He was an academic visitor at National University of Singapore (NUS) in 2007, a senior visiting scholar at The University of Suwon (USW) of South Korea in 2011, an engineer at Shenyang Aircraft Design Research Institute, AVIC in 2006, and a technician of AVIC Aviation Motor Control System Institute from 1996 to 2000. He is currently a full professor with the School of Automation Science and Electrical Engineering, Beihang University (formerly Beijing University of Aeronautics and Astronautics, BUAA). He is the head at the BUAA Bio-inspired Autonomous Flight Systems (BAFS) Research Group. He received the National Science Fund for Distinguished Young Scholars of China, the 16th MAO Yi-sheng Beijing Youth Science and Technology Award, and the Sixth National Outstanding Scientific and Technological Worker of China in 2014. He received the 19th National Youth Five-Four Medal Award in 2015, the 13th China Youth Science and Technology Award, the 12th Youth Science and Technology Award of Chinese Association of Aeronautics and Astronautics in 2013. He is also enrolled in the Top-Notch Young Talents Program of China, Program for New Century Excellent Talents in University of China, and Beijing NOVA Program. He has authored or coauthored more than 70 publications and three monographs. He is the editor-in-chief of "*International Journal of Intelligent Computing and Cybernetics*" (IJICC). His current research interests include bioinpsired computing, biological computer vision, and multiple UAVs autonomous formation control. He is a senior member of the IEEE.