Binary Pigeon-Inspired Optimization for Quadrotor Swarm Formation Control

Zhiqiang Zheng[1], Haibin Duan[1,2(✉)], and Chen Wei[1]

[1] School of Automation Science and Electrical Engineering, Beihang University, Beijing 100083, China
hbduan@buaa.edu.cn
[2] Peng Cheng Laboratory, Shenzhen 518000, China

Abstract. This paper proposes a binary pigeon-inspired optimization (BPIO) algorithm, for the quadrotor swarm formation control problem. The expected position is provided by the BPIO. Quadrotor moves to the position with control strategy, and the strategy is based on the proportional integral derivative (PID) control method. The BPIO algorithm which is based on pigeon-inspired optimization (PIO) algorithm can effectively solve the combination problem in the binary solution space. The BPIO keeps the fast convergence of the PIO, and can explore the space effectively at the same time. The parameters to be optimized are encoded with binary bits. A special fitness function is designed to avoid the happening of crash. The simulation experiment shows how the BPIO works. The results of simulation verify the feasibility and effectiveness of the BPIO to solve the swarm formation problem.

Keywords: Binary pigeon-inspired optimization (BPIO) · Quadrotor · Swarm formation

1 Introduction

The quadrotor has small size, light weight and the ability to Vertically Take-Off and Land (VTOL) [1]. For a single quadcopter, many effective control strategies are proposed, like PID control strategy [2]. On the other hand, Unmanned Aircraft System (UAS) has been applied in military and other fields, which brings many problems of swarm formation. The swarm formation requests the agents in the swarm complete a task together, and the agents usually have to form a given formation. The quadcopter swarm formation problem is one of the research focuses. However, the swarm formation control has strong coupling and nonlinearity [3]. Some kinds of methods to solve it are proposed, like artificial potential function-based [3], observer-based, leader-follower [4] and so on. The problem is how to find the best way to form the formation accurately and quickly. In this case, intelligence optimization algorithms can solve this problem efficiently because of their irreplaceable feature [3].

The PIO algorithm is presented in 2014, which is based on the natural characteristics of pigeons [5]. The homing ability of pigeons used by PIO is based on the sun, the

Y. Tan et al. (Eds.): ICSI 2020, LNCS 12145, pp. 71–82, 2020.
https://doi.org/10.1007/978-3-030-53956-6_7

Earth's magnetic field and landmarks. However, the basic PIO algorithm is easy to fall into the local optimal solution [3]. It is necessary to try to strengthen PIO's global search capability [6, 7]. The binary particle swarm optimization (BPSO) is developed by Kennedy and Eberhart [8] and used to optimize combinational problems. Over the years, many kinds of novel BPSO algorithms have been developed, like sticky binary PSO [9], chaotic BPSO [10] and so on.

In this paper, the model of quadrotor and how to control a quadrotor with PID control method are introduced firstly. Then the BPIO is developed and use it to solve the quadrotor swarm formation control problem. At last, the BPIO is compared with other binary algorithms to confirm its rapid convergence.

2 Dynamics Modeling and Control of a Quadrotor

2.1 Coordinate System

Two coordinate systems are defined and shown in Fig. 1. The coordinate system Σ_e is defined as an earth-fixed coordinate system, and also as an inertial coordinate system. The o_e is the take-off position. The x_e axis points to an arbitrary direction in the horizontal plane. The z_e axis is perpendicular and points downward, and y_e is defined by the right-hand rule. The coordinate system Σ_b is defined as the body-fixed coordinate system. The o_b is the center of gravity (COG) of the quadrotor. The x_b axis is in the symmetry plane and points to the nose direction. The z_b axis is also in the symmetry plane, and perpendicular to x_b axis, and the y_b is also defined by right-hand rule. The rotation angles around x_b, y_b and z_b axes are defined as roll angle, pitch angle and yaw angle and denoted as ϕ, θ and ψ, respectively [2].

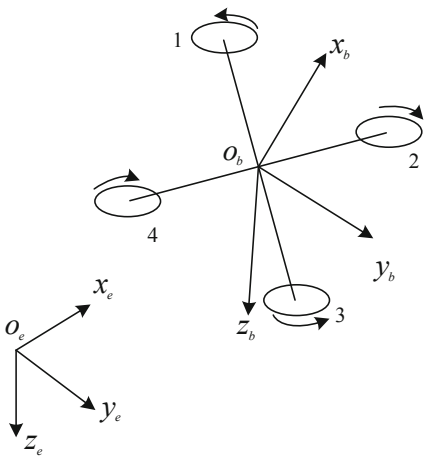

Fig. 1. The coordinate systems

2.2 Dynamics Modeling

To simplify the problem, the quadrotor is simplified as a rigid body and the deformation of the blade is ignored. The equations are expressed as follows [2]:

$$\begin{cases} {}^{e}\dot{\mathbf{p}} = {}^{e}\mathbf{v} \\ m{}^{e}\dot{\mathbf{v}} = m\mathbf{g} + \mathbf{R} \cdot ({}^{b}\mathbf{f} + {}^{b}\mathbf{D}) \\ \dot{\mathbf{\Theta}} = \mathbf{W} \cdot {}^{b}\boldsymbol{\omega} \\ \mathbf{J} \cdot {}^{b}\dot{\boldsymbol{\omega}} = -{}^{b}\boldsymbol{\omega} \times (\mathbf{J} \cdot {}^{b}\boldsymbol{\omega}) + \mathbf{G}_a + \boldsymbol{\tau} \end{cases} \tag{1}$$

where e means the vector is defined in Σ_e, and b means in Σ_b. And m is the total mass of a quadrotor. Other symbols are listed in Table 1.

Table 1. The symbols in Eq. (1)

	Explain		Explain
$\mathbf{p}$	The vector of position	$\mathbf{D}$	The vector of drag
$\mathbf{v}$	The vector of velocity	$\mathbf{\Theta}$	The vector of Euler angle[a]
$\mathbf{g}$	Acceleration of gravity	$\boldsymbol{\omega}$	The vector of angle velocity
$\mathbf{R}$	Rotation matrix from Σ_b to Σ_e	$\mathbf{W}$	See Eq. (2)
$\mathbf{f}$	The force from propellers	$\mathbf{G}_a$	The gyroscopic moments
$\mathbf{J}$	Rotational inertia	$\boldsymbol{\tau}$	The moments from propellers

[a] $\mathbf{\Theta} = [\phi, \theta, \psi]^T$

$$\mathbf{W} = \begin{bmatrix} 1 & \tan\theta \sin\phi & \tan\theta \cos\phi \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi / \cos\theta & \cos\phi / \cos\theta \end{bmatrix} \tag{2}$$

2.3 The Strategy of Control Design

To control the quadrotor effectively, PID control method is used in this paper [1, 2]. The designed control system is shown in Fig. 2.

For position controller and attitude controller, the control variables, $\mathbf{f}_d$ and $\boldsymbol{\tau}_d$, are generated by the PID control rules based on $\mathbf{v}$, $\boldsymbol{\omega}$ and the rates of errors. The rates of errors are generated by P control rules based on the error of input variables, $\mathbf{p}$ and $\mathbf{p}_d$, $\mathbf{\Theta}$ and $\mathbf{\Theta}_d$. Noting that $\mathbf{\Theta}_d = [\phi_d, \theta_d, \psi_d]^T$. ψ_d is usually set to zero, and ϕ_d and θ_d are generated by position controller. And the rates of errors are looked as $\mathbf{v}_d$ and $\boldsymbol{\omega}_d$. The control distributor can generate the desired angular speeds of rotors based on the control variables. Then, based on Eq. (1), the state of quadrotor can be calculated and the state values can be used for the next iteration [1, 2].

The main data of leader and followers is given in the Table 2, where d is the distance between COG and the motor, c_T is the thrust constant and c_M is the torque constant [2].

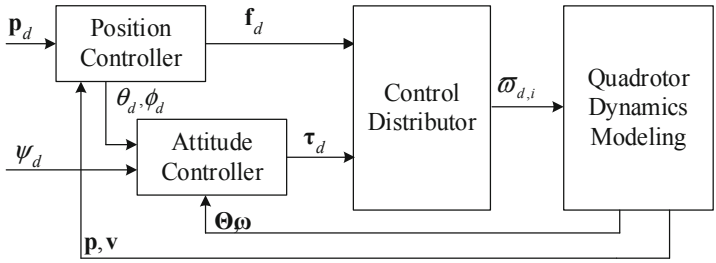

Fig. 2. The control strategy

Table 2. Data of the leader and followers

	Leader	Follower 1&3	Follower 2&4
m/kg	1.0230	1.5230	1.0830
$\mathbf{J}/kg \cdot m^2$	$diag(0.0095,$ $0.0095, 0.0186)$	$diag(0.0105,$ $0.0105, 0.0186)$	$diag(0.0088,$ $0.0088, 0.0176)$
d/m	0.2223	0.2323	0.2023
$c_T/kg \cdot m$	1.4865×10^{-7}	1.6865×10^{-7}	1.1865×10^{-7}
$c_M/kg \cdot m^2$	2.9250×10^{-9}	3.0250×10^{-9}	2.4250×10^{-9}

3 The BPIO Algorithm

3.1 Pigeon-Inspired Optimization

The PIO algorithm includes the map and compass operator and the landmark operator [5–7]. In PIO, each pigeon individual represents a feasible solution. The pigeon individual i has its position X_i and velocity V_i. The positions and velocities are in the D-dimension solution space, and the value of dimension is based on the problem under study. Every pigeon is evaluated by its fitness value. The fitness value is calculated by the fitness function. A brief presentation of basic PIO is given as follow.

Map and Compass Operator

The map and compass operator is used at the beginning of the calculation. When this operator is working, every pigeon's position and velocity are updated in every iteration as follows [5]:

$$V_i(t) = V_i(t-1) \cdot e^{-Rt} + rand \cdot (X_{gbest} - X_i(t-1)) \tag{3}$$

$$X_i(t) = X_i(t-1) + V_i(t) \tag{4}$$

where t is current iteration number, R is defined as the map and compass factor, X_{gbest} denotes the global best position among all individual in current iteration, *rand* means a random number in [0, 1].

In every iteration, the individual fitness value is calculated using the new position, and the X_{gbest} is updated only if a smaller fitness value appears under the condition that the cost function is a minimization problem.

Landmark Operator

When the landmark operator is working, the number of pigeons reduces by half in every iteration, and the positions of pigeons are updated by the strategy as follows [5].

$$N_p(t) = \frac{N_p(t-1)}{2} \tag{5}$$

$$X_c(t) = \frac{\sum X_i(t) \cdot \text{fitness}(X_i(t))}{N_p \sum \text{fitness}(X_i(t))} \tag{6}$$

$$X_i(t) = X_i(t-1) + rand \cdot (X_c(t) - X_i(t-1)) \tag{7}$$

where N_p is the population size of pigeons, X_c is the center of the current pigeons, and fitness() is the fitness function. Noting that the minimum of N_p is 1, and for the minimum problems, usually, fitness $(X_i(t)) = \frac{1}{\text{fitness}_{\min}(X_i(t))+\varepsilon}$, where ε is a constant.

According to Eq. (5), half of the pigeons are ignored in every iteration because these pigeons are far away from the destination, the center of the left individuals, and the pigeons near the destination will fly to the destination very quickly.

3.2 Binary Pigeon-Inspired Optimization

It is obvious that the original PIO algorithm is a kind of continuous algorithm, like the PSO. To solve combinational problems, the Binary PSO (BPSO) and improved algorithms are developed [8–10]. In the same way, the Binary PIO (BPIO) is developed.

In BPIO, the velocity entries are used to indicate the probability that the corresponding position entries flip either from 1 to 0 or from 0 to 1. The position entries are either 1 or 0, as defined in BPSO [9].

When the map and compass operator works, to make the pigeon have cognitive ability, $X_{pbest,i}$ is taken into consideration where $X_{pbest,i}$ is the best position of the pigeon i. The velocity is updated when map and compass operator works as follow:

$$V_i(t) = V_i(t-1)e^{-Rt} + (1-s) \cdot r_1(X_{gbest} - X_i(t-1)) + s \cdot r_2(X_{pbest,i} - X_i(t-1)) \tag{8}$$

where r_1 and r_2 are random number in [0, 1] and s is the cognitive factor that declines as the map and compass operator works.

To make the pigeon have stronger global search ability in the beginning and stronger convergence as the iteration goes on, the value of s is dynamically updated as follow [9]:

$$s = s_{\max} - (s_{\max} - s_{\min}) \cdot \frac{t}{N_1} \tag{9}$$

where $s_{\max}$ and $s_{\min}$ represent the maximum and minimum value of the cognitive factor s. N_1 is the maximum number of iterations of the map and compass operator.

The strategy of how to flip the position of the individual is given now. Because the velocity entries indicate the probability, its value should be limited in the range [0, 1]. Hence, like BPSO, the sigmoid function is also used to meet the requirement [9, 10]. The sigmoid function is given as follow.

$$\text{sigmoid}(V_i(t)) = \frac{2}{1 + e^{-|V_i(t)|}} - 1 \tag{10}$$

The position entries can be updated by the equation as follow.

$$X_i(t+1) = \begin{cases} 1 & r_3 < \text{sigmoid}(V_i(t)) \\ 0 & other \end{cases} \tag{11}$$

where r_3 is a random number uniformly distributed between 0 and 1.

When the landmark operator works, the population size reduces by half and the positions and velocities are updated in every iteration. To avoid the value being smaller than 1, the population size is updated as follow.

$$N_p(t) = \left\lceil \frac{N_p(t-1)}{2} \right\rceil \tag{12}$$

where $\lceil \bullet \rceil$ means the ceiling operation.

The pigeons left will fly to the destination X_{gbest}. At the beginning of the iteration, to make the pigeons have stronger ability to search the solution space near the destination, the velocity is redefined as follow.

$$V_i(N_1) = rand \cdot V_i(N_1) \tag{13}$$

Then the velocity is updated as follow and the position is updated in the same way as Eq. (11).

$$V_i(t) = w \cdot V_i(t-1) + rand \cdot (X_{gbest} - X_i(t-1)) \tag{14}$$

where w is the inertia factor.

Steps of the BPIO algorithm are given as follows.

Step 1: Initialize parameters of BPIO algorithm. Set the population size N_p, solution space dimension D, the map and compass factor R, the number of iterations N_1 and N_2 for two operators and the ranges of velocity for each dimensions. Encode the possible solutions of the problem with binary bits.
Step 2: Initialize the positions and velocities of pigeons with random numbers. Decode the position, calculate the fitness and record as $X_{pbest,i}$ for every pigeon. Then find the X_{gbest}.
Step 3: Map and compass operator works. Update velocity according to Eq. (8) and position according to Eq. (11) for every pigeon. Decode the position and calculate the fitness for every pigeon. Then update the $X_{pbest,i}$ and X_{gbest} if necessary.
Step 4: If $k > N_1$, update the latest velocity entries according to Eq. (13) and change to landmark operator. Otherwise, go to step 3.

Step 5: Landmark operator works. Rank pigeons according to fitness values, and ignore half of the individuals far away from the destination. Then update the other pigeons' velocities according to Eq. (14) and positions according to Eq. (11). Decode the position and calculate the fitness for every pigeon. Then find the X_{gbest}.
Step 6: If $k > N_1 + N_2$, stop the landmark operator and output the result. If not, go to step 5.

4 BPIO for the Quadrotor Swarm Formation Control

4.1 Encoding the Parameters to be Optimized

In this problem, there are three parameters to be optimized, as shown in Table 3. The definitions of these parameters are shown in Fig. 3. Noting that the coordinate system $o_b x'_e y'_e z'_e$ is produced by translating the $o_e x_e y_e z_e$ until o_e is coincident with o_b.

Table 3. Encoding of the parameters

Parameter	Range	Binary bits
λ/rad	$\frac{\pi}{3}, \frac{\pi}{2}, \frac{2\pi}{3}, \frac{5\pi}{6}, \frac{7\pi}{6}, \frac{4\pi}{3}, \frac{3\pi}{2}, \frac{5\pi}{3}$	000, 001, 010, 011, 100, 101, 110, 111
ρ/m	2, 3, 3.5, 4	00, 01, 10, 11
h/m	$-3, -1.5, 1.5, 3$	00, 01, 10, 11

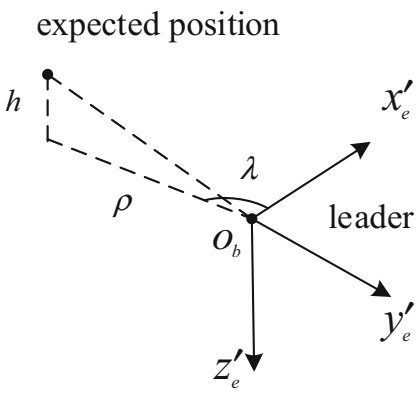

Fig. 3. The definitions of the parameters

From the Table 3, the dimension number is seven which means that every pigeon represents seven binary bits in the binary solution space. The meaning of every bit and the map of bits and parameters' values are also shown in Table 3.

For a follower, when the combination of three parameters' values is selected by the BPIO, the desired position of the follower is selected.

4.2 The Fitness Function

When selecting a suitable fitness function for the quadcopter swarm formation problem, there are two basic parts that must be included. Firstly, the fitness function should include the information of the expected position and the current position of the followers. Secondly, the crash of the followers must be avoided. To satisfy these requests, the fitness function of *i-th* follower is defined as follow:

$$\text{fitness}_i = \begin{cases} \left\|\mathbf{p}_{i,cur} - \mathbf{p}_{i,exp}\right\|^2 & \left\|\mathbf{p}_{i,cur} - \mathbf{p}_{j,cur}\right\|_{\min} > 1 \\ f_{\max} & \left\|\mathbf{p}_{i,cur} - \mathbf{p}_{j,cur}\right\|_{\min} < 1 \end{cases} \tag{15}$$

where $\mathbf{p}_{i,cur}$ is the current position, $\mathbf{p}_{i,exp}$ is the desired position, and $f_{\max}$ is a large number which is always larger than $\left\|\mathbf{p}_{i,cur} - \mathbf{p}_{i,exp}\right\|^2$. Also, i is not equal to j. The minimum distance of followers is set to 1 m. If the distance between *i-th* follower and another follower is smaller than 1 m, $f_{\max}$ will be assigned to fitness_i.

4.3 Simulation Results and Analysis

In the simulation, there are five quadrotors including a leader and four followers. Figures 4, 5, 6 and 7 show the result when the BPIO is used to optimize the flight path. The leader starts at the position $[0, 0, -3]^T$, and then flies to the position $[10, 10, -13]^T$. After hovering for 3 s, the leader flies to the position $[20, 10, -13]^T$ and arrives at 22 s. After that, the leader flies to the position $[45, 10, -18]^T$ and arrives at 45 s and lastly, it hovers at there until the simulation stops (at 52 s). The followers take off at the same time, and their desired positions are given by BPIO for every a second. The step of simulation is 0.01 s. The followers then update their desired position according to the results of optimization and the real time location of the leader for each iteration. Then, followers can flight to the desired position by the control strategy shown in Fig. 2.

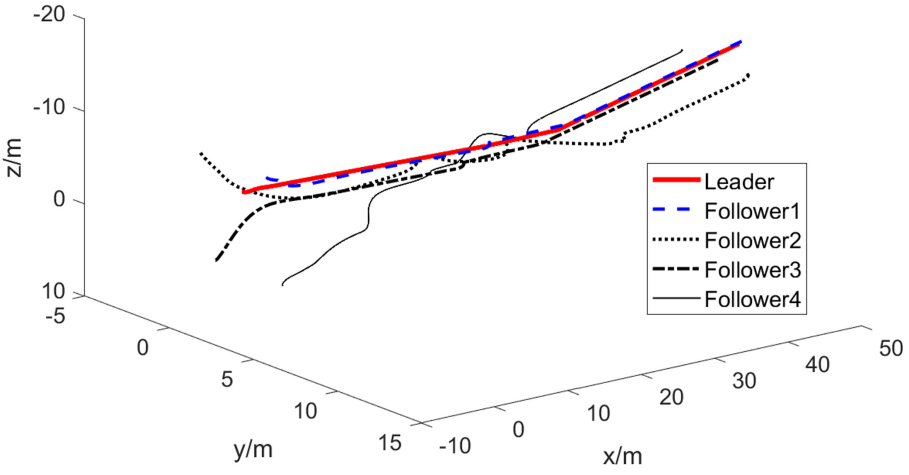

Fig. 4. Simulation result in a 3-D view

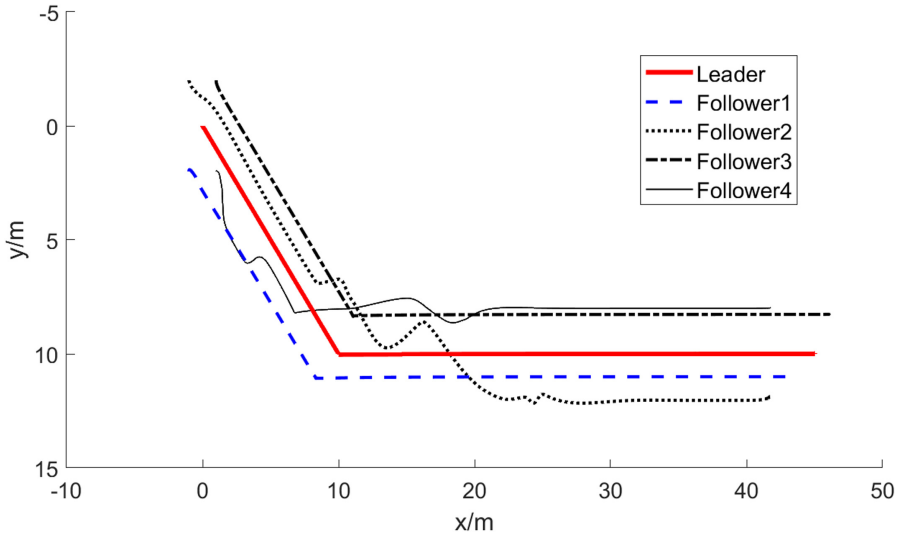

Fig. 5. Simulation result in a top-down view

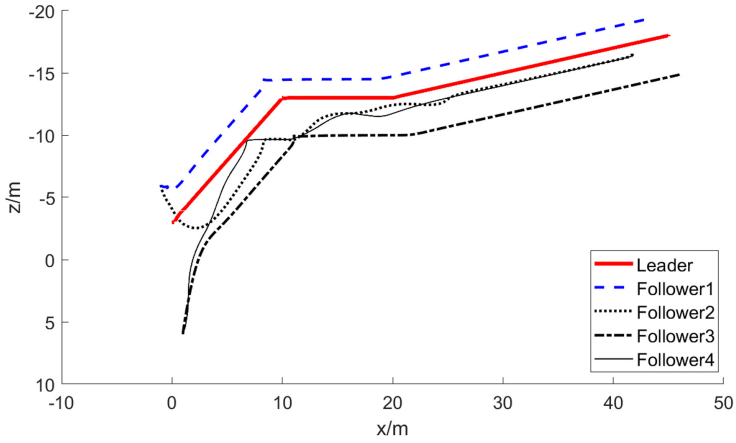

Fig. 6. Simulation result in a side view

From Figs. 4, 5 and 6, it is obvious that BPIO algorithm can form a feasible swarm. Because responding to the position signals, the velocity of leader almost always changes with time. On the other hand, the followers have their own mobility. Thus the followers sometimes may be unable to response to the current desired position quickly and need to get a new expect position. The BPIO algorithm can optimize a new desired position for the followers immediately. Specially, the follower 2 and 4 have more complex flight path. Because follower 2 and 4 optimize their path later than follower 1 and 3, follower 2

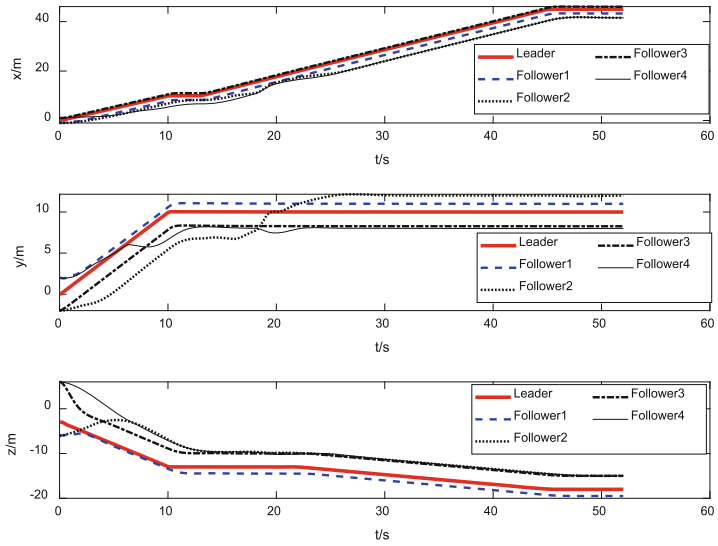

Fig. 7. Simulation result in position-time figure

and 4 must respond when near to other followers. The simulation result also verifies that the fitness function (15) is feasible to avoid the crash even if the equation is so simple.

To confirm the rapid convergence, the BPIO is compared with the novel binary PSO in [11] and the binary PSO in [12]. The range of the particles (or pigeons) is set to $[-50, 50]$, the population size is 100 and the number of iteration is set to 1000 as in [11]. For BPIO, N_1 is set to 600, and N_2 is set to 400. The expression of the test function is as follow:

$$f(x) = \frac{1}{4000}\sum_{i=1}^{3} x_i^2 - \prod_{i=1}^{3} \cos\frac{x_i}{\sqrt{i}} + 1 \tag{16}$$

Figure 8 shows that the BPIO can converge more quickly. Table 4 shows the optimization results after running the algorithms for 20 times.

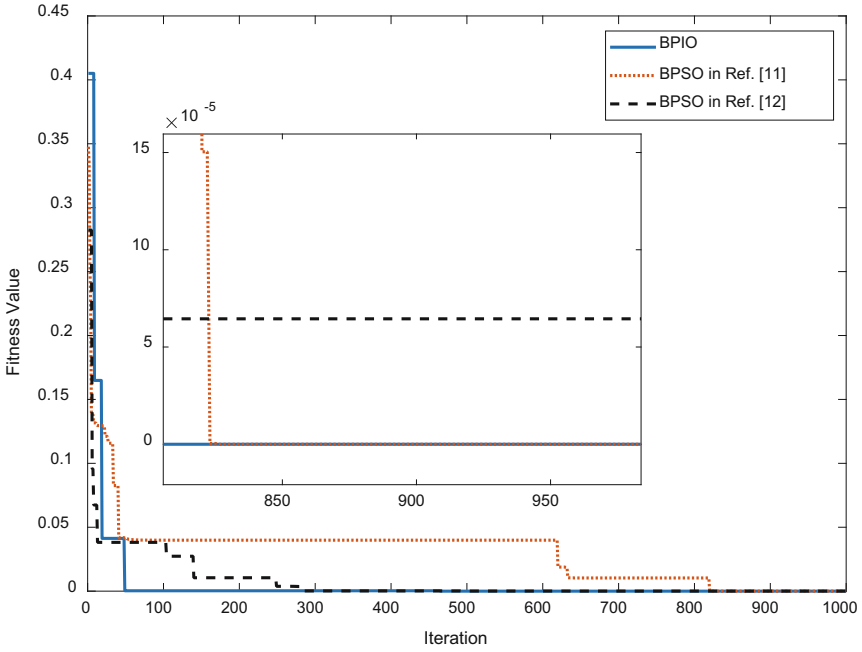

Fig. 8. Iterative curve of the algorithms

Table 4. The results after running the algorithms 20 times

	BPIO	BPSO as in [11]	BPSO as in [12]
Best optimization result	2.085968×10^{-9}	2.085968×10^{-9}	7.354714×10^{-6}
The cost time of the best result	1.949235 s	2.155005 s	2.770884 s
Mean of the optimization results	0.000110	0.014247	0.007056
Mean of the cost time	2.087806 s	2.382623 s	2.904913 s

5 Conclusion

The swarm formation is a challenging problem. In this paper, the rigid body model of quadrotor is used for simulation. The BPIO algorithm is used to optimize the expected position for the followers, and then the followers move to the expected position based on the PID control method. At the same time, the method to avoid the crash is applied when selecting the fitness function. The simulation verifies that the BPIO algorithm can effectively produce the expected position, avoid the crash and converge quickly. From the comparison of BPIO algorithm and other two kinds of BPSO algorithm, the high performance of BPIO algorithm is confirmed.

References

1. Qasim, M., Susanto, E., Wibowo, A.S.: PID control for attitude stabilization of an unmanned aerial vehicle quad-copter. In: 2017 5th International Conference on Instrumentation, Control, and Automation (ICA), Yogyakarta, pp. 109–114 (2017)
2. Quan, Q.: Introduction to Multicopter Design and Control. Springer, Singapore (2017). https://doi.org/10.1007/978-981-10-3382-7
3. Duan, H., Tong, B., Wang, Y., Wei, C.: Mixed game pigeon-inspired optimization for unmanned aircraft system swarm formation. In: Tan, Y., Shi, Y., Niu, B. (eds.) ICSI 2019. LNCS, vol. 11655, pp. 429–438. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26369-0_40
4. Panagou, D., Kumar, V.: Cooperative visibility maintenance for leader–follower formations in obstacle environments. IEEE Trans. Robot. **30**(4), 831–844 (2014)
5. Duan, H., Qiao, P.: Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning. Int. J. Intell. Comput. Cybern. **7**(1), 24–37 (2014)
6. Duan, H., Huo, M., Yang, Z., Shi, Y., Luo, Q.: Predator-prey pigeon-inspired optimization for UAV ALS longitudinal parameters tuning. IEEE Trans. Aerosp. Electron. Syst. **55**(5), 2347–2358 (2019). https://doi.org/10.1109/TAES.2018.2886612
7. Qiu, H., Duan, H.: A multi-objective pigeon-inspired optimization approach to UAV distributed flocking among obstacles. Inf. Sci. **509**, 515–529 (2020). https://doi.org/10.1016/j.ins.2018.06.061
8. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference of Neural Networks, vol. 4, no. 2, pp. 1942–1948 (1995)
9. Nguyen, B.H., Xue, B., Zhang, M.: A new binary particle swarm optimization approach: momentum and dynamic balance between exploration and exploitation. IEEE Trans. Cybern. 1–15 (2019). https://doi.org/10.1109/TCYB.2019.2944141
10. Li, P., Xu, D., Zhou, Z., Lee, W., Zhao, B.: Stochastic optimal operation of microgrid based on chaotic binary particle swarm optimization. IEEE Trans. Smart Grid **7**(1), 66–73 (2016)
11. Khanesar, M.A., Teshnehlab, M., Shoorehdeli, M.A.: A novel binary particle swarm optimization. In: Mediterranean Conference on Control & Automation 2007, MED 2007, pp. 1–6 (2007)
12. Kennedy, J., Eberhart, R.C.: A discrete binary version of the particle swarm algorithm. In: 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, vol. 5, pp. 4104–4108 (1997)