Hindawi
Mobile Information Systems
Volume 2022, Article ID 8519708, 23 pages
https://doi.org/10.1155/2022/8519708

Research Article

A Novel UAV Path Planning Algorithm Based on Double-Dynamic Biogeography-Based Learning Particle Swarm Optimization

Yisheng Ji,[1] Xinchao Zhao ,[1] and Junling Hao[2]

[1]*School of Science, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[2]*School of Statistics, University of International Business and Economics, Beijing 100029, China*

Correspondence should be addressed to Xinchao Zhao; zhaoxc@bupt.edu.cn

Received 12 November 2021; Accepted 14 December 2021; Published 29 January 2022

Academic Editor: Jianhui Lv

Particle swarm optimization (PSO), one of the classical path planning algorithms, has been considered for unmanned aerial vehicle (UAV) path planning more frequently in recent years. A large amount of studies on UAV path planning based on modified PSO have been reported. However, most UAV path planning algorithms still optimize only one kind terrain problem which is mountain terrain. At the same time, many modified PSO algorithms also have some problems, such as insufficient convergence and unsatisfactory efficiency. In this paper, six kinds of terrain functions of UAV path planning are proposed to simulate real-world application. The terrain functions contain city, village without houses, village with houses, mountainous area without houses, mountainous area with houses, and mountainous area with a huge building. Inspired by CLPSO and BLPSO, we proposed a new double-dynamic biogeography-based learning particle swarm optimization (DDBLPSO) algorithm to solve these problems. The double-dynamic biogeography-based learning strategy replacing the traditional learning mechanism from the personal and global best particles is used to select the learning particles. In this strategy, each particle will learn from the better one of two selected particles which are not worse than itself. However, one random component of particle will replaced by corresponding component of other particle if all components of the particle only learn from itself. In this way, particles sufficiently learn from better objects and maintain the ability of jumping out of local optimality. The superiority of our algorithm is verified with four relevant algorithms, a PSO variant, and a BBO variant on the benchmark suite of CEC2015. Real-world application demonstrates that the algorithm we proposed outperforms four relevant algorithms, a PSO variant, and a BBO variant both in small-scale problems and large-scale problems. This paper shows a good application of our novel algorithm.

1. Introduction

UAV path planning is designed in a space which represents environment experienced by UAV during its flight. The path is a link formed by the combination of all the points and the lines connecting them, and each path is a solution. Let population $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_\mathbf{N}]$, where N is population size and X_i is the ith solution. Every path has m points, and it does not include the starting and the end points. $\mathbf{X_i} = [x_{i1}, y_{i1}, z_{i1}; \ldots; x_{iy}, y_{iy}, z_{iy}; \ldots; x_{im}, y_{im}, z_{im}]$, where (x_{iy}, y_{iy}, z_{iy}) is the coordinate of the jth point of the ith path. Solution is an $m \times 3$ matrix, and population is an $m \times 3 \times N$ tensor. The space is divided into $m + 1$ equal parts along X axis which ensures that the flight path goes from the starting point to the end point without forming a circle during halfway.

UAV path planning is a multiobjective constraint problem, which is described as follows:

$$\begin{aligned} \min\ & f_i(\mathbf{X}), \quad i = 1, 2, \ldots, m, \\ & c_j(\mathbf{X}) \le 0, \quad j = 1, 2, \ldots, p, \end{aligned} \tag{1}$$

where $f_i(\mathbf{X}), i = 1, 2, \ldots, m$, are objective functions and $c_j(\mathbf{X}) \le 0, j = 1, 2, \ldots, p$, are constraints. We transformed the multiobjective constrained optimization into single-objective unconstrained optimization in order to simplify the problem. All objectives and constraints are multiplied by a weight coefficient and then added. Finally, the objective

cost function is defined as equation (2) aiming to minimize the total cost.

$$F = \sum_{i=1}^{m} W_i \times f_i + \sum_{j=1}^{p} W_{j+m} \times c_j, \tag{2}$$

where $W_i\,(i = 1, 2, \ldots, m + p)$ are weight coefficients. In this paper, UAV aims to fly as short distance as possible, fly as low as possible, try not to cross danger zones as possible, and try not to cross the ground and buildings as possible. So, three objectives and one constraint are selected which are described as follows:

$$\begin{aligned} F = {} & W_1 \times f_{\text{length}} + W_2 \times f_{\text{altitude}} \\ & + W_3 \times f_{\text{danger}} + W_4 \times c_{\text{collision}}, \end{aligned} \tag{3}$$

where $W_1, W_2, W_3, W_4 \in (0, 1]$ are weight coefficients, f_{length} is a path function penalizing the longer paths, f_{altitude} is an altitude function penalizing higher average altitudes, f_{danger} is a danger function penalizing the paths going through danger zones, and $c_{\text{collision}}$ is a collision function penalizing the paths colliding with ground or buildings. f_{length}, f_{altitude}, f_{danger} are objective functions, and $c_{\text{collision}}$ is a constraint function.

Path function is used to penalize longer paths which are modelled as equation (4) in this paper.

$$f_{\text{length}} = 1 - \frac{L_{P1P2}}{L_{\text{traj}}}, \tag{4}$$

where L_{P1P2} is the length of the line segment which connects the starting point $P1$ and the end point $P2$ and L_{traj} is the actual length of the trajectory. This means that L_{traj} is the length of a broken line connecting all points from the starting point to the end point. It is easy to get that $f_{\text{length}} \in (0, 1]$.

Altitude function is used to penalize higher average altitudes which is modelled as equation (5) in this paper.

$$f_{\text{altitude}} = \frac{\left|A_{\text{traj}} - Z_{\min}\right|}{Z_{\max} - Z_{\min}}, \tag{5}$$

where A_{traj} is the average altitude and Z_{min} and $Z_{\max}$ are the upper and lower limits of the elevation in the search space. However, we usually choose a very low height rather than zero as the optimal choice which means that $Z_{\min}$ is a small positive number. It is easy to get $f_{\text{altitude}} \in (0, 1]$.

Danger function is used to penalize the paths going through danger zones which is modelled as equation (6) in this paper.

$$f_{\text{danger}} = \frac{L_{\text{danger}}}{\sum_{i=1}^{n} d_i}, \tag{6}$$

where L_{danger} is the sum of the length of the subsections of the trajectory which go through danger zones, d_i is the diameter of the ith danger zone, and n is the number of danger zones. In this paper, danger zone usually means that radar can detect and is described as a hemispheric region. If $f_{\text{danger}} > 1$, f_{danger} is set to be 1 to ensure $f_{\text{danger}} \in (0, 1]$.

Collision function is used to penalize the paths colliding with ground or buildings which is modelled as follows:

$$c_{\text{collision}} = \begin{cases} 0, & L_{\text{infeasible}} = 0, \\ P + \dfrac{L_{\text{infeasible}}}{L_{\text{traj}}}, & L_{\text{infeasible}} \neq 0, \end{cases} \tag{7}$$

where $L_{\text{infeasible}}$ is the total length of the subsections of the trajectory which travels below the ground or buildings, P is the penalty constant, and L_{traj} is the length of the trajectory. It is easy to say that $c_{\text{collision}} \in \{0\} \bigcup [P, P + 1]$. If cost function $F(\mathbf{X_i})$ is greater than P, it means that $F(\mathbf{X_i})$ is an infeasible solution.

There are many modified PSO algorithms, and some of them have been applied to UAV path planning [1–10]. However, many PSO variants still have some problems like insufficient convergence and accuracy and unsatisfactory efficiency. In this paper, we proposed a novel double-dynamic biogeography-based learning particle swarm optimization (DDBLPSO) algorithm to solve these problems. The algorithm not only considers the problem of insufficient convergence and accuracy and unsatisfactory efficiency for global optimization but also can be applied to UAV path planning. The main contributions of our paper are summarized as follows.

(i) A novel double-dynamic biogeography-based learning strategy replacing the traditional learning mechanism is proposed. This strategy makes the utmost of the advantages of particles which only learn from particles that are not worse than themselves.

(ii) Six kinds of terrain functions are designed in UAV path planning.

(iii) The experiment results show that DDBLPSO demonstrates the best performance on UAV path planning with four other representative algorithms, a PSO variant, and a BBO variant in CEC2015 benchmark functions and UAV path planning.

The remainder of the paper is organized as follows. Section 2 gives the literature review. Section 3 introduces several existing algorithms such as PSO, BBO, CLPSO, and BLPSO. Section 4 shows the details of the proposed algorithm DDBLPSO. Section 5 presents simulation results for global optimization. Application in UAV path planning is elaborated in Section 6. Section 7 draws the conclusions.

2. Literature Review

Unmanned aerial vehicle (UAV) has been applied to many areas with the rapid progress of science and technology. Evolutionary algorithm-based methods solving the UAV path planning problem are always a hot topic. In [1], the authors proposed an evolutionary algorithm based on offline/online path planning for UAV navigation. Phase angle-encoded and quantum-behaved particle swarm optimization was proposed and applied to three-dimensional route

planning for UAV [2]. Pehlivanoglu [3] introduced a new vibrational genetic algorithm which was enhanced by Voronoi diagram for path planning of autonomous UAV. An improved constrained differential evolution algorithm was introduced for UAV global route planning [4]. The algorithm demonstrates a good performance in terms of the solution quality, robustness, and the constraint-handling ability. A method was proposed to compare the planner performance by jointly employing several general and problem-specific quality indexes, which takes into account the complexity and particularities of the problem [5]. In [6], a novel predator-prey pigeon-inspired optimization (PPPIO) was proposed to solve the three-dimension path planning problem for unmanned combat air vehicle (UCAV) in dynamic environment. This algorithm mainly focuses on optimizing the flight route considering different types of constraints under complex combating environment. The modified wolf pack search (WPS) was applied to compute the quasi-optimal trajectories for the rotor wing UAVs in the complex three-dimensional (3D) spaces including the real and fake 3D spaces [7]. In [8], the authors proposed a constrained adaptive multiobjective differential evolution algorithm for bistatic SAR path planning. It generates multiple feasible paths for the UAV receiver with different trade-offs between navigation for UAV and bistatic SAR imaging performance. The authors compared genetic algorithm and particle swarm optimization for real-time UAV path planning in [9].

Particle swarm optimization (PSO) [10] is one of the most commonly used algorithms to solve the problems of UAV path planning. In recent years, many improved PSO variants have emerged one after another. In [11], the authors put forward an adaptive particle swarm optimization (APSO) algorithm, in which two main steps are conducted to adaptively adjust the parameters when the swarm lies in a different evolutionary state (exploration, exploitation, convergence, and jumping out) in each generation. Then, an elitist learning strategy was used when the evolutionary state was classified as convergence state. Nickabadi et al.[12] studied on adaptive inertia weight and introduced a novel particle swarm optimization algorithm. Numerical experiments show that this algorithm is quite effective in adapting the value of w in the dynamic and static environments. Li and Yao [13] presented a new cooperative coevolving particle swarm optimization (CCPSO) algorithm to scale up particle swarm optimization algorithms in solving large-scale optimization problems (up to 2000 real-valued variables). Comprehensive learning particle swarm optimization (CLPSO) uses a learning strategy whereby all other particles' historical best information is used to update a particle's velocity [14]. In [15], the authors proposed a novel biogeography-based optimization algorithm with momentum migration and taxonomic mutation to deal with problems whose function values change dramatically or barely. In [16], the authors proposed an enhanced particle swarm optimization algorithm (pkPSO) by combining k-nearest neighbours (k-NN) with pattern search (PS). At the same time, the cooperative effect of k-NN and PS strategies was verified. A novel particle swarm optimization algorithm was proposed for parameter determination and feature selection of support vector machines [17]. A novel PSO-GA-based hybrid training algorithm with Adam optimization which performs well in training artificial neural networks was introduced by Yadav and Anubhav [18]. In [19], the authors presented a cellular learning automata-based bare bones PSO algorithm with maximum likelihood rotated mutations. The PSO technique was used to identify the uncertain physical parameters of a real vehicle ETC system [20]. In [21], the authors proposed a new particle swarm optimization algorithm with simple PID-based strategy, which has good global optimization ability. Biogeography-based optimization (BBO) is an algorithm that simulates biological migration to search optimal solution [22]. Biogeography-based learning strategy is a nice method for optimization. In [23], biogeography-based learning particle swarm optimization (BLPSO) uses a learning strategy based on migration of biogeography-based optimization and outperforms other representative algorithms. In [24], the authors introduced a hybrid differential evolution with biogeography-based optimization for global numerical optimization.

3. Relevant Algorithms

3.1. Particle Swarm Optimization (PSO). Particle swarm optimization is a classical evolutionary computing technique which was firstly proposed by Eberhart and Kennedy [10]. Inspired from the study of the predation behaviour of birds, the main idea of PSO algorithm is sharing cooperation and information among individuals to find the optimal solution. PSO simulates birds in a flock by designing a massless particle, which only has velocity and position. Velocity represents how fast and which direction the particle moves along. Position represents where the particle is. For each particle, only the personal best experience and the global best experience of the entire swarm can be learned. Let $x_i = (x_{i1}, x_{i2}, \ldots, x_{i\,D})^T$ and $v_i = (v_{i1}, v_{i2}, \ldots, v_{i\,D})^T$ denote the position and velocity of particle $i (i = \{1, 2, \ldots, N\})$, respectively, where D is dimension of the initial space and N is population size. Let $\text{pbest}_i = (\text{pbest}_{i1}, \text{pbest}_{i2}, \ldots, \text{pbest}_{i\,D})^T$ and $\text{gbest} = (\text{gbest}_1, \text{gbest}_2, \ldots, \text{gbest}_D)^T$ be the personal best position of particle i and the global best position of the whole swarm. The update of velocity and position of the particle is indicated as follows:

$$\begin{aligned} v_{\text{id}} &= w * v_{\text{id}} + c_1 * \text{rand}_1(0,1) * \left(\text{pbest}_{\text{id}} - x_{id}\right) \\ &\quad + c_2 * \text{rand}_2(0,1) * \left(\text{gbest}_d - x_{id}\right), \\ x_{id} &= x_{id} + v_{id}, \end{aligned} \tag{8}$$

where $i = 1, 2, \ldots, N$; $d = 1, 2, \ldots, D$; w is the inertia weight; c_1 and c_2 are the acceleration coefficients; and $\text{rand}_1(0,1)$ and $\text{rand}_2(0,1) \in [0,1]$ are uniform random numbers. The algorithm sets the maximum velocity to prevent the velocity from getting too large. The particle will be set as the border of initial region if it flies out of initial region.

3.2. Biogeography-Based Optimization (BBO). In [22], the author proposed a new algorithm named biogeography-based optimization which solves optimization problems by

simulating biological migration. In BBO, solution is also called as habitat, fitness of solution is called as habitat suitability index (HSI), and component of solution is called as suitable index variable (SIV). The main body of BBO is migration operation and mutation operation.

Each habitat $x_i\,(i = 1, 2, \ldots, N)$ has two parameters to describe the migration, which are an immigration rate λ_i and an emigration rate μ_i. Both parameters are closely related to HSI. For high HSI habitat, there will be a high trend of outward migration. At this time, the emigration rate is high and the immigration rate is low due to the pressure of species competition. However, for low HSI habitat, the opposite is true. Assuming habitat x_i currently accommodates S_i species, $S_{\max}$ is the maximum number of species. Immigration rate λ_i and emigration rate μ_i are usually described as follows:

$$\begin{cases} \lambda_i = I * \left(1 - \dfrac{S_i}{S_{\max}}\right), \\ \\ \mu_i = E * \dfrac{S_i}{S_{\max}}, \end{cases} \tag{9}$$

where I is the maximum immigration rate and E is maximum emigration rate. The component $x_{\mathrm{id}}\,(d = 1, 2, \ldots, D)$ of habitat x_i will immigrate depend on immigration rate λ_i. The habitat x_j is selected depending on emigration rate μ_j.

The properties of the habitat such as HSI and the number of species may changes due to unexpected events. At the same time, mutation rate depends on species probability. According to biogeography, when the number of species in habitat is too large or too small, species probability is low. When the number of species in habitat is moderate, species probability is high. Mutation rate m_i is described as follows:

$$m_i = m_{\max} * \left(1 - \frac{P_i}{P_{\max}}\right), \tag{10}$$

where P_i is the species probability of x_i decided by the number of species S_i, $P_{\max}$ is the maximum species probability, and $m_{\max}$ is the maximum mutation rate.

3.3. Comprehensive Learning Particle Swarm Optimization (CLPSO). Comprehensive learning particle swarm optimization [14] adopts the strategy of comprehensive learning to select the objects to learn instead of learning from itself and the global optimal individual. The velocity updating equation in CLPSO is defined as follows:

$$v_{\mathrm{id}} = w * v_{\mathrm{id}} + c * \mathrm{rand}(0, 1) * \left(\mathrm{pbest}_{f_i(d),d} - x_{id}\right), \tag{11}$$

where f_i defines which particles' pbests that the particle i should follow and $\mathrm{rand}(0, 1) \in [0, 1]$ is a uniform random number. The inertia weight w in CLPSO is a linear attenuation coefficient. CLPSO assigns a learning probability Pc_i for each particle i using the following equation:

$$Pc_i = 0.05 + 0.45^{*}\frac{\exp\left(10\,(i-1)/N - 1\right) - 1}{\exp(10) - 1}. \tag{12}$$

For each solution x_i, it will learn from many particles instead of only two particles. Each component of particle i learn from itself or other particle depending on learning probability Pc_i. A random component of particle i will learn from other particle if all its components learn from itself. The higher fitness value a solution has, the greater possibility that particle will be learned.

3.4. Biogeography-Based Learning Particle Swarm Optimization (BLPSO). Biogeography-based learning particle swarm optimization [23] adopts the strategy of migration operation of BBO to select the objects to learn. Each component of particle i learns from itself or a selected particle j depending on immigration rate λ_i. The selected particle j depends on emigration rate μ_j. A random component of particle i will learn from other particle if all components of particle i learn from itself. The higher the fitness value of a solution is, the smaller the immigration rate and the greater the migration rate of the particle are. It means that the better particle has larger probability to be learned and the worse particle has the larger probability to learn from others. The velocity updating equation in BLPSO is the same as equation (11). The learning probability Pc_i for particle i and the selected probability Ps_j for particle j can be expressed as follows:

$$\mathrm{Pc}_i = \lambda_i, \tag{13}$$

$$\mathrm{Ps}_j = \frac{\mu_j}{\sum_{k=1}^{N} \mu_k}, \tag{14}$$

It should be noted that learning probability Pc is different from selected probability Ps. Particle i has a probability of learning probability Pc_i to learn from other particles. At the same time, particle j has a selected probability Ps_{ij} to be selected as an object which particle i learn from. The relationship between learning probability and selected probability is shown in Figure 1.

It is worth mentioning that it uses the quadratic migration model instead of the linear migration model. The quadratic migration model is as follows:

$$\begin{cases} \lambda_i = I * \left(1 - \dfrac{S_i}{S_{\max}}\right)^2, \\ \\ \mu_i = E * \left(\dfrac{S_i}{S_{\max}}\right)^2. \end{cases} \tag{15}$$

The differences of CLPSO and BLPSO are learning probability and the selection strategy of the learning particle.

4. Double-Dynamic Biogeography-Based Learning Particle Swarm Optimization (DDBLPSO)

Although many promising PSO variants have emerged, they still have some problems like insufficient convergence and accuracy and unsatisfactory efficiency. These cause them to

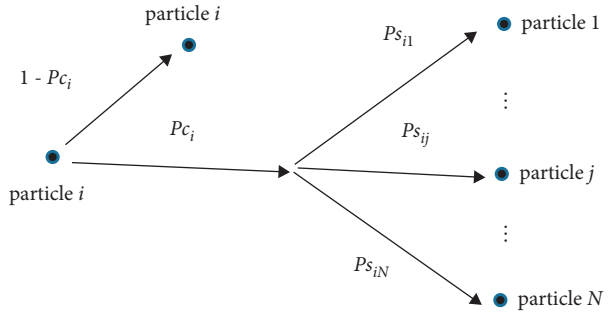


FIGURE 1: Relationship between learning probability and selected probability.

perform poorly on some complicated functions such as multimodal functions, hybrid functions, composition functions, and real-world problems. Regardless of CLPSO or BLPSO, there is a probability that particle learns from those particles with smaller fitness than itself. These strategies lead some superior particles are not taken full advantage of their superiority and result a low convergence. A double-dynamic selection strategy based on biogeography-based learning is proposed in this paper, in which particles learn from the better one of two dynamic roulette winners. This strategy guides the particle to learn from others that are not worse than itself. In this way, the advantages of the superior particles will be greatly exploited.

4.1. Dynamic Biogeography-Based Learning. Original biogeography-based learning strategy guides particles to learn from the promising particles. However, most particles still have a certain probabilities of learning from worse particles. For each particle, dynamic biogeography-based learning strategy only guides it to learn from other particles not worse than itself. All the particles are sorted in ascending order based on fitness. Particle i will learn from a selected particle if a uniform random number generated in [0,1] is smaller than predefined λ_i. The selected probability Ps_{ij} of particle j being selected by particle i is defined as follows:

$$\mathrm{Ps}_{ij} = \frac{\mu_j}{\sum_{k=i}^{N} \mu_k}, \tag{16}$$

where $j \geq i$. The greater the index number is, the higher fitness the particle has. The learning object particle j is selected with a dynamic roulette which constitutes even more competitive individuals $\mu_i, \mu_{i+1}, \ldots, \mu_N$. In this way, particle will not learn from the worse particles which can reduce the possibility for the particles getting worse and worse. Generally speaking, on the one hand, dynamic biogeography-based learning strategy keeps the worse particles larger exploration abilities to learn from others. On the other hand, the strategy gives the better particles fewer objectives to learn from. It should be pointed out that the best solution only can learn from itself.

4.2. Double-Dynamic Biogeography-Based Learning. Original biogeography-based learning strategy only has one selected particle to learn while original comprehensive learning strategy has two selected particles and then chooses the better one. The strategy has two selected particles, which has a greater possibility to have a high fitness offspring than other strategy only has one selected particle. Inspired by comprehensive learning strategy, double-dynamic biogeography-based learning also has two selected particles and leaves the better one to ensure the high rate of convergence. This strategy allows particles to adequately learn from better particles. For each component x_{id}, the learning objective $x_{Li,d}$ is selected by using the following equation:

$$x_{\mathrm{Li},d} = \begin{cases} x_{\mathrm{ad}}, & \text{if fitness of } x_a \text{ is greater than fitness of } x_b, \\ x_{\mathrm{bd}}, & \text{if fitness of } x_a \text{ is smaller than fitness of } x_b, \end{cases} \tag{17}$$

where x_a and x_b are two particles which are selected by dynamic roulette and x_{ad} and x_{bd} represent the dth component of x_a and x_b, respectively.

4.3. Mutation. According to above strategies, although the particles will fly to a better area, particles are easy to fall into local optimum. One random component of particle will be replaced by other particle's component if the particle only learns from itself. Only high fitness particles have large probability to learn from themselves. There are two reasons for this phenomenon. One is that a high fitness particle has a larger immigration rate. The other is that a high fitness particle only has few particles better than itself. This strategy helps particles to jump out of local optimum. If all components of x_i only learn from themselves, the learning objective x_{Li} is set as follows:

$$x_{\mathrm{Li},j} = x_{l,j}, \quad \text{if } x_{\mathrm{Li}} == x_i, \tag{18}$$

where j and l are random numbers and $l \in \{1, 2, \ldots, i-1, i+1, \ldots, N\}$, $j \in \{1, 2, \ldots, D\}$.

In the early stage of DDBLPSO, the algorithmic search area is large because initial particles evenly distribute in the search area. In the last stage of DDBLPSO, the algorithmic search area becomes smaller and smaller because most particles tend to be very close together. However, the better particle will have the greater probability to learn from other particle's past optimal position because of the mutation mechanism. It prevents the algorithm from falling into local optimum.

4.4. Algorithm Process. Double-dynamic biogeography-based learning particle swarm optimization is proposed based on above strategies. The process of DDBLPSO is described in Algorithm 1.

DDBLPSO, CLPSO, and BLPSO have some commonalities and differences. It is necessary to illustrate the differences among them. Two main differences are summarized as follows.

(i) In CLPSO, learning objective is selected uniformly and randomly. In BLPSO, learning objective is selected by roulette based on equation (14). In

(1) Initial maximum velocity v_{max}, upper bound x_{max} and lower bound x_{min} of the initial region, population x, velocity v, inertia weight w, acceleration coefficient c, the maximal number of function evaluations maxFES;
(2) Record personal best position pbest and global best position gbest;
(3) while FES < maxFES do
(4) Sort solutions in ascending order based on fitness;
(5) Calculate immigration rate λ_i and emigration rate $\mu_i (i = 1, 2, \ldots, N)$, update inertia weight w, and let learning population $xL = \text{zeros}(\text{size}(x))$;
(6) for $i = 1$ to N
(7) for $d = 1$ to D
(8) If rand $< \lambda_i$
(9) Select pbest_a and $\text{pbest}_b (a, b \in \{1, 2, \ldots, N\})$ with a dynamic roulette;
(10) if fitness of x_a is greater than fitness of x_b
(11) $xL_{id} = \text{pbest}_{ad}$;
(12) else
(13) $xL_{id} = \text{pbest}_{bd}$;
(14) end if
(15) else
(16) $xL_{id} = \text{pbest}_{id}$
(17) end if
(18) end for
(19) if all $(xL_i == \text{pbest}_i)$
(20) Randomly select $j \neq i (j \in \{1, 2, \ldots, N\})$ and $l (l \in \{1, 2, \ldots, D\})$;
(21) $xL_{id} = \text{pbest}_{id}$;
(22) end if
(23) end for
(24) $v = w * v + c * \text{rand}(\text{size}(x)). * (xL - x).$;
(25) $v(v > v_{max}) = v_{max}$; $v(v < -v_{max}) = -v_{max}$;
(26) $x = x + v$;
(27) $x(x > x_{max}) = x_{max}$; $x(x < x_{min}) = x_{min}$;
(28) Update personal best position pbest and global best position gbest;
(29) end while
(30) Output the final result.

Algorithm 1: Double-dynamic biogeography-based learning particle swarm optimization (DDBLPSO).

DDBLPSO, learning objective is selected by the dynamic roulette based on equation (16).

(ii) In BLPSO, each component of a particle only has one learning objective. However, in CLPSO and DDBLPSO, each component of a particle has two learning objectives and then selects the better one.

5. Numerical Simulations for Global Optimization

5.1. Test Functions and Parameter Settings. CEC2015 [25] benchmark functions are used to verify the performance of DDBLPSO. CEC2015 benchmark suite is simply introduced as follows. $f1$ and $f2$ are unimodal functions, $f3 - 5$ are multimodal functions, $f6 - 8$ are hybrid functions and $f9 - 15$ are composition functions.

CLPSO [14] is a classical algorithm proposed based on PSO [10], and BLPSO [23] is an algorithm proposed based on BBO [22] and PSO. Inspired by CLPSO and BLPSO, we proposed DDBLPSO. DDBLPSO has an obvious relationship with BBO, PSO, CLPSO, and BLPSO, so we selected these four algorithms for comparison experiments. At the same time, we also selected two other representative algorithms which are PBSPSO [21] and DEBBO [24] for comparison experiments. We conducted four sets of numerical experiments which are labelled as experiment one, experiment two, experiment three, and experiment four. DDBLPSO is compared with four relevant algorithms (PSO, BBO, CLPSO, and BLSPSO), a PSO variant (PBSPSO), and a BBO variant (DEBBO). Design of the four experiments is shown in Table 1. The maximal number of function evaluations (maxFES) is set as $10000 * D$. Search range is $[-100, 100]^D$, and 30 independent runs are conducted in MATLAB 2017b. Other parameters are shown in Table 2.

5.2. Experimental Results and Discussion. Results of experiment one, experiment two, experiment three, and experiment four are statistically shown in Tables 3–6, respectively. Tables 7–10 show the ranks of five algorithms according to the Friedman test of experiment one, experiment two, experiment three, and experiment four, respectively. Min, Mean, Median, and Std indicate the minimum function error value, the mean function error value, the median function error value, and the standard deviation of error values, respectively. In order to exhibit the evolution trend of five algorithms more vividly, converging curves of the average best fitness of functions in experiment one and experiment three are shown in Figures 2 and 3. The converging

Table 1: Design of experiment one, experiment two, experiment three, and experiment four.

Experiment	Population size N	Points number mp	Comparison algorithm
Experiment one	30	30	PSO, BBO, CLPSO, BLSPSO
Experiment two	100	100	PSO, BBO, CLPSO, BLSPSO
Experiment three	30	30	PBSPSO, DEBBO
Experiment four	100	100	PBSPSO, DEBBO

Table 2: Parameters of seven comparison algorithms.

Algorithm	Parameters
PSO	Inertia weight $w = 0.55$, acceleration coefficient $c_1 = c_2 = 2.0$, maximum velocity $v_{\max} = 5.0$
BBO	Maximum immigration rate $I = 1$, maximum emigration rate $E = 1$, maximum mutation rate $m_{\max} = 0.005$
CLPSO	Inertia weight w is linearly decrease from 0.9 to 0.2, acceleration coefficient $c = 2.0$, maximum velocity $v_{\max} = 5.0$
BLPSO	Inertia weight w is linearly decrease from 0.9 to 0.2, acceleration coefficient $c = 2.0$, maximum velocity $v_{\max} = 5.0$, maximum immigration rate $I = 1$, maximum emigration rate $E = 1$
PBSPSO	Inertia weight $w = 0.55$, acceleration coefficient $c = 2.0$, maximum velocity $v_{\max} = 5.0$, decay term $\alpha = 0.9$, the weight of the derivative term $K_d = 0.03$
DEPSO	Maximum immigration rate $I = 1$, maximum emigration rate $E = 1$, maximum mutation rate $m_{\max} = 0.005$, crossover rate CR = 0.5, difference coefficient $F_X = 0.7$
DDBLPSO	Inertia weight w is linearly decrease from 0.9 to 0.2, acceleration coefficient $c = 2.0$, maximum velocity $v_{\max} = 5.0$, maximum immigration rate \$I = 1\$, maximum emigration rate $E = 1$

Table 3: Performance comparison of five algorithms in experiment one.

	BBO	PSO	CLPSO	BLPSO	DDBLPSO
$f1(x)$					
Min	1.6583 E+06	1.7321 E+06	7.7577 E+05	5.6816 E+05	2.0171 E+059
Mean	4.1619 E+06	1.1235 E+07	1.5641 E+06	1.3061 E+06	.9974 E+058
Median	4.2666 E+06	9.4901 E+06	1.5160 E+06	1.0850 E+06	0404E+05
Std	1.2125 E+06	8.1069 E+06	3.9988 E+05	6.4276 E+05	7.7346 E+05
$f2(x)$					
Min	5.9570 E+07	2.1247 E+06	8.6937 E+04	1.0573 E+03	**2.0057 E+02**
Mean	1.0746 E+08	6.4067 E+06	4.1390 E+05	7.6477 E+03	**3.0520 E+03**
Median	1.0441 E+08	5.5692 E+06	3.5350 E+05	6.7947 E+03	**1.3179 E+03**
Std	2.2645 E+07	3.1246 E+06	2.9045 E+05	5.3207 E+03	**4.2472 E+03**
$f3(x)$					
Min	3.2033E+02	**3.2014E+02**	3.2118 E+02	3.2109 E+02	3.2104 E+02
Mean	3.2093E+02	**3.2026E+02**	3.2134 E+02	3.2131 E+02	3.2128 E+02
Median	3.2092E+02	**3.2026E+02**	3.2135 E+02	3.2131 E+02	3.2129 E+02
Std	3.1412E−01	**6.9821E-02**	7.9537E-02	9.3207E-02	1.1575E-01
$f4(x)$					
Min	5.5276 E+02	4.3839 E+02	4.1498 E+02	4.1592 E+02	4.0995 E+02
Mean	5.8999 E+02	4.5826 E+02	4.2232 E+02	4.2498 E+02	4.1878 E+02
Median	5.8819 E+02	4.5863 E+02	4.2201 E+02	4.2388 E+02	4.1840 E+02
Std	2.3338 E+01	1.1451 E+01	**4.1135 E+00**	6.1503 E+00	5.2623 E+00
$f5(x)$					
Min	2.7940 E+03	1.8947 E+03	1.2380 E+03	1.2387 E+03	1.3481 E+03
Mean	4.2481 E+03	2.6178 E+03	1.9363 E+03	1.6544 E+03	1.7612 E+03
Median	4.1462 E+03	2.5958 E+03	1.9451 E+03	1.5976 E+03	1.7362 E+03
Std	7.4098 E+02	4.0091 E+02	3.1450 E+02	2.9163 E+02	2.5296 E+02
$f6(x)$					
Min	8.5892 E+04	4.7158 E+05	1.3661 E+05	3.4194 E+04	2.7682 E+04
Mean	3.3191 E+05	5.7133 E+06	3.0127 E+05	1.5594 E+05	1.1685 E+05
Median	2.8357 E+05	5.4783 E+06	2.7053 E+05	1.5365 E+05	9.0779 E+04
Std	1.8363 E+05	3.7012 E+06	1.1982E+05	8.9082 E+04	8.6045 E+04
$f7(x)$					
Min	7.0813 E+02	7.0836 E+02	7.0626 E+02	7.0570 E+02	7.0517 E+02
Mean	7.1215 E+02	7.1799 E+02	7.1138 E+02	7.0934 E+02	7.0754 E+02

Table 3: Continued.

	BBO	PSO	CLPSO	BLPSO	DDBLPSO
Median	7.1023 E+02	7.1372 E+02	7.1242 E+02	7.0795 E+02	7.0586 E+02
Std	4.0321 E+00	1.7801 E+01	2.8718 E+00	3.1576 E+00	2.8994 E+00
$f8(x)$					
Min	2.9403 E+04	3.7955 E+04	2.1194 E+04	1.1270 E+04	1.1820 E+04
Mean	1.1430 E+05	1.5521 E+06	6.0038 E+04	4.7864 E+04	4.5841 E+04
Median	9.0699 E+04	1.1775 E+06	5.8822 E+04	4.5294 E+04	3.1882 E+04
Std	9.6038 E+04	1.4589 E+06	2.6934 E+04	2.8916 E+04	3.5138 E+04
$f9(x)$					
Min	1.0045 E+03	1.0046 E+03	1.0033 E+03	1.0032 E+03	1.0024 E+03
Mean	1.0740 E+03	1.0059 E+03	1.0040 E+03	1.0076 E+03	1.0037 E+03
Median	1.0064 E+03	1.0058 E+03	1.0040 E+03	1.0038 E+03	1.0037 E+03
Std	1.3863 E+02	7.2713E−01	2.7210E−01	2.0779 E+01	5.1535E−01
$f10(x)$					
Min	2.0119 E+04	4.0609 E+05	7.5343 E+04	2.9240 E+04	2.1702 E+04
Mean	5.0981 E+05	2.5287 E+06	2.0769 E+05	2.0526 E+05	1.0437 E+05
Median	7.9737 E+04	1.7787 E+06	2.1072 E+05	1.1130 E+05	7.1822 E+04
Std	9.4063 E+05	2.2614 E+06	7.6381 E+04	4.0970 E+05	1.0132 E+05
$f11(x)$					
Min	1.4042 E+03	1.8057 E+03	1.4035 E+03	1.4015 E+03	1.4013 E+03
Mean	1.6772 E+03	1.9459 E+03	1.4617 E+03	1.5100 E+03	1.4753 E+03
Median	1.4285 E+03	1.9471 E+03	1.4056 E+03	1.4031 E+03	1.4021 E+03
Std	3.1792 E+02	6.2006E+01	1.1830 E+02	1.5473 E+02	1.1947E+02
$f12(x)$					
Min	1.3100 E+03	1.3084 E+03	1.3040 E+03	1.3049 E+03	1.3040 E+03
Mean	1.3214 E+03	1.3111 E+03	1.3058 E+03	1.3066 E+03	1.3071 E+03
Median	1.3139 E+03	1.3110 E+03	1.3055 E+03	1.3064 E+03	1.3075 E+03
Std	2.1131 E+01	1.5834 E+00	9.8335E-01	1.1492 E+00	1.4291 E+00
$f13(x)$					
Min	1.4175 E+03	1.3933 E+03	1.4352 E+03	1.4144 E+03	1.3995 E+03
Mean	1.4252 E+03	1.4078 E+03	1.4420 E+03	1.4259 E+03	1.4173 E+03
Median	1.4244 E+03	1.4086 E+03	1.4416 E+03	1.4254 E+03	1.4164 E+03
Std	4.8717 E+00	6.6629 E+00	3.4109 E+00	6.0813 E+00	7.2068E+00
$f14(x)$					
Min	3.7298 E+03	3.2799 E+04	3.2503 E+04	3.2503 E+04	1.5000 E+03
Mean	3.7381 E+04	3.4675 E+04	3.3058 E+04	3.3235 E+04	3.2591 E+04
Median	3.8788 E+04	3.4793 E+04	3.2651 E+04	3.2651 E+04	3.3648 E+04
Std	1.0738 E+04	9.1682 E+02	7.4020 E+02	1.0085 E+03	5.9283 E+03
$f15(x)$					
Min	1.6042 E+03	1.6007 E+03	1.6000 E+03	1.6000 E+03	1.6000 E+03
Mean	1.6105 E+03	1.6012 E+03	1.6001 E+03	1.6000 E+03	1.6004 E+03
Median	1.6127 E+03	1.6011 E+03	1.6001 E+03	1.6000 E+03	1.6000 E+03
Std	5.5047 E+00	1.9659E-01	3.2672E-02	4.7803E-03	2.3806 E+00

curves of functions in experiment two and experiment four are not shown due to their similarities with experiment one and experiment three, respectively. The boldface in Tables 3–6 indicates the best experimental results in experiment one to four, respectively.

Unimodal functions are used to explore the convergence rate of the optimization problem. Multimodal functions are used to explore the ability to jump out of local optimum. Hybrid functions consider that in the real-world optimization problems, different subcomponents of the variables may have different properties. Composition functions consider that in the real-world optimization problems, different simple problems composite a more complex problem. This is a challenge for algorithmic convergence rate. Most of the multimodal functions contain a number of local optima, which may lead to premature convergence of traditional algorithms. It is difficult for traditional algorithms to locate the global optimum for these functions.

It is easy to observe that DDBLPSO shows its dominance over other competitors on unimodal functions and hybrid functions from Table 3. However, DDBLPSO performs less on multimodal functions and composition functions. This means that DDBLPSO has much greater convergence rate than its competitors and performs slightly better than its competitors in jumping out of local optimum. The double-dynamic biogeography-based learning strategy is the main reason for this phenomenon. This

Table 4: Performance comparison of five algorithms in experiment two.

	BBO	PSO	CLPSO	BLPSO	DDBLPSO
$f1(x)$					
Min	1.6872 E+07	1.2995 E+07	1.2592 E+07	9.6047 E+06	**8.8196 E+06**
Mean	3.3932 E+07	2.4028 E+07	2.0317 E+07	1.2281 E+07	**1.1151 E+07**
Median	3.3742 E+07	2.3245 E+07	2.0301 E+07	1.2062E+07	**1.1181 E+07**
Std	5.2575 E+06	6.4139 E+06	2.7277 E+06	1.4734 E+06	**1.0671 E+06**
$f2(x)$					
Min	5.5282 E+08	**1.3908 E + 06**	2.1847 E+08	9.6042 E+07	2.3631 E+07
Mean	6.4517 E+08	**2.2934 E + 06**	2.5524 E+08	1.2074E+08	3.2844 E+07
Median	6.3717 E+08	**2.1154 E + 06**	2.5419 E+08	1.2030E+08	3.2473 E+07
Std	5.2834 E+07	**5.7400 E + 05**	1.7095 E+07	1.3223 E+07	4.2897 E+06
$f3(x)$					
Min	3.2133 E+02	**3.2079E + 02**	3.2134 E+02	3.2139 E+02	3.2136 E+02
Mean	3.2145 E+02	**3.2085E + 02**	3.2148 E+02	3.2146 E+02	3.2145 E+02
Median	3.2147 E+02	**3.2086E + 02**	3.2148 E+02	3.2145 E+02	3.2145 E+02
Std	5.8776E-02	**3.7493 E - 02**	3.9590E-02	4.0692E-02	3.9328E-02
$f4(x)$					
Min	9.1048 E+02	5.0448 E+02	5.9808 E+02	5.2056E+02	**4.8020 E+02**
Mean	1.0131 E+03	7.5271 E+02	6.2837 E+02	5.4878 E+02	**5.0111 E+02**
Median	1.0112 E+03	8.3702 E+02	6.2944 E+02	5.4789 E+02	**4.9888 E+02**
Std	4.6330 E+01	1.5583 E+02	1.3892 E+01	1.3776 E+01	**1.0271 E+01**
$f5(x)$					
Min	1.2581 E+04	1.7639 E+04	8.4258 E+03	5.5564 E+03	**4.7234 E+03**
Mean	1.4282 E+04	1.9160 E+04	9.5008 E+03	7.1715 E+03	**5.5794 E+03**
Median	1.4090 E+04	1.9176 E+04	9.3825 E+03	7.2019E+03	**5.3980 E+03**
Std	9.6948 E+02	8.2684 E+02	6.2791 E+02	6.5159 E+02	**6.1390 E+02**
$f6(x)$					
Min	1.8440 E+06	7.3864 E+06	2.3964 E+06	1.6318 E+06	**1.3833 E+06**
Mean	3.7058 E+06	1.3158 E+07	3.3193 E+06	2.1834 E+06	**1.9272 E +06**
Median	3.6446 E+06	1.2457 E+07	3.2200 E+06	2.2014E+06	**1.9522 E+06**
Std	9.8093 E+05	4.1777 E+06	5.9871 E+05	3.0671 E+05	**2.6129 E+05**
$f7(x)$					
Min	7.3969 E+02	8.1342 E+02	7.9195 E+02	7.3534 E+02	**7.3350 E+02**
Mean	**7.8438 E+02**	8.4624 E+02	8.2456 E+02	7.9700 E+02	8.0765 E+02
Median	**7.7023 E+02**	8.5004 E+02	8.2193 E+02	8.0104 E+02	8.0020 E+02
Std	4.1385 E+01	2.0194 E+01	**1.8298 E+01**	3.2732 E+01	2.8476 E+01
$f8(x)$					
Min	1.2878 E+06	4.1243 E+06	1.1840 E+06	**5.0520 E+05**	5.1323 E+05
Mean	1.9175 E+06	1.0590 E+07	1.9332 E+06	8.9139 E+05	**6.9234 E+05**
Median	1.8616 E+06	1.0212 E+07	1.9618 E+06	8.9816 E+05	**6.8432 E+05**
Std	3.7847 E+05	3.8045 E+06	3.2658 E+05	2.2895 E+05	**1.3364 E+05**
$f9(x)$					
Min	1.0125 E+03	1.0110 E+03	1.0072 E+03	1.0069 E+03	**1.0068 E+03**
Mean	1.1742 E+03	1.0138 E+03	1.0079 E+03	1.0079 E+03	**1.0077 E+03**
Median	1.0414 E+03	1.0138 E+03	1.0078 E+03	1.0079 E+03	**1.0076 E+03**
Std	2.8267 E+02	1.2870 E+00	**5.2094E-01**	5.5162E-01	5.2679E-01
$f10(x)$					
Min	2.0038 E+06	2.1053 E+06	2.7821 E+05	4.7205 E+04	**1.4355 E+04**
Mean	3.8010 E+06	3.7765 E+06	4.9280 E+05	5.5835 E+04	**1.6831 E+04**
Median	3.5081 E+06	3.3635 E+06	4.9422 E+05	5.6250 E+04	**1.6362 E+04**
Std	1.5857 E+06	1.2352 E+06	9.5961 E+04	5.1752 E+03	**2.1156 E+03**
$f11(x)$					
Min	1.4231 E+03	2.5629 E+03	1.4173 E+03	**1.4100 E+03**	1.4102 E+03
Mean	2.1106 E+03	2.8176 E+03	1.7756 E+03	**1.4136 E+03**	1.6127 E+03
Median	1.4395 E+03	2.8267 E+03	1.4220 E+03	1.4139 E+03	**1.4126 E+03**
Std	7.9489 E+02	1.1589 E+02	6.5399 E+02	**1.5746 E+00**	4.5786 E+02
$f12(x)$					
Min	1.3561 E+03	1.3177 E+03	1.3137 E+03	1.3126 E+03	**1.3111 E+03**
Mean	1.3821 E+03	1.3202 E+03	1.3154 E+03	1.3153 E+03	**1.3147 E+03**

Table 4: Continued.

	BBO	PSO	CLPSO	BLPSO	DDBLPSO
Median	1.3811 E+03	1.3202 E+03	1.3155 E+03	1.3154 E+03	**1.3149 E+03**
Std	1.3010 E+01	1.1930E+00	**8.0780E-01**	1.1071 E+00	1.3710 E+00
$f13(x)$					
Min	1.7453 E+03	**1.6946 E+03**	1.7725 E+03	1.7411 E+03	1.7344 E+03
Mean	1.7600 E+03	**1.7135 E+03**	1.7825 E+03	1.7534 E+03	1.7481 E+03
Median	1.7602 E+03	**1.7137 E+03**	1.7833 E+03	1.7540 E+03	1.7487 E+03
Std	7.6554 E+00	7.6113 E+00	**5.0447 E+00**	5.9510 E+00	6.3159 E+00
$f14(x)$					
Min	1.5092 E+04	1.1046 E+05	1.1030 E+05	5.0142 E+03	**3.1265 E+03**
Mean	1.5690 E+05	1.1051 E+05	1.1094 E+05	**3.2405 E+04**	1.1036 E+05
Median	1.5755 E+05	1.1051 E+05	1.1032 E+05	**5.5990 E+03**	1.1770 E+05
Std	4.4260 E+04	**4.0365 E+01**	1.8805 E+03	4.7720 E+04	2.9479 E+04
$f15(x)$					
Min	1.6086 E+03	**1.6010 E+03**	1.6063 E+03	1.6048 E+03	1.6031 E+03
Mean	1.6159 E+03	1.6041 E+03	1.6066 E+03	1.6052 E+03	**1.6037 E+03**
Median	1.6157 E+03	1.6042 E+03	1.6066 E+03	1.6052 E+03	**1.6038 E+03**
Std	3.5097 E+00	1.6093 E+00	**1.4693E−01**	1.9835E−01	3.1219E−01

Table 5: Performance comparison of three algorithms in experiment three.

	PBSPSO	DEBBO	DDBLPSO
$f1(x)$			
Min	1.8592 E+06	4.6232 E+05	**2.0171 E+05**
Mean	4.0977 E+06	2.0454 E+06	**9.9974 E+05**
Median	3.8062 E+06	1.6998 E+06	**8.0404 E+05**
Std	1.4323 E+06	1.3320 E+06	**7.7346 E+05**
$f2(x)$			
Min	8.0625 E+07	2.0358 E+02	**2.0057 E+02**
Mean	1.2518 E+08	4.0561 E+03	**3.0520 E+03**
Median	1.3096 E+08	**1.1378 E+03**	1.3179 E+03
Std	2.6223 E+07	4.5224 E+03	**4.2472 E+03**
$f3(x)$			
Min	3.2058E+02	**3.2026E+02**	3.2104 E+02
Mean	3.2103 E+02	**3.2034E+02**	3.2128 E+02
Median	3.2106 E+02	**3.2035E+02**	3.2129 E+02
Std	2.5954E−01	**4.7224E-02**	1.1575E−01
$f4(x)$			
Min	5.3829 E+02	4.2006E+02	**4.0995 E+02**
Mean	5.9001 E+02	4.3346 E+02	**4.1878 E+02**
Median	5.9107 E+02	4.3369 E+02	**4.1840 E+02**
Std	2.6829 E+01	7.5401 E+00	**5.2623 E+00**
$f5(x)$			
Min	3.2276 E+03	2.4576 E+03	**1.3481 E+03**
Mean	4.1617 E+03	3.1351 E+03	**1.7612 E+03**
Median	4.1973E+03	3.1853 E+03	**1.7362 E+03**
Std	5.1598 E+02	3.1687 E+02	**2.5296 E+02**
$f6(x)$			
Min	9.2791 E+04	1.1225 E+05	**2.7682 E+04**
Mean	3.3549 E+05	3.7463 E+05	**1.1685 E+05**
Median	3.1758 E+05	2.9917 E+05	**9.0779 E+04**
Std	1.8022 E+05	2.4642 E+05	**8.6045 E+04**
$f7(x)$			
Min	7.0838 E+02	7.0717 E+02	**7.0517 E+02**
Mean	7.1403 E+02	7.1034 E+02	**7.0754 E+02**
Median	7.1384 E+02	7.1015 E+02	**7.0586 E+02**
Std	4.4317 E+00	**1.6748 E+00**	2.8994 E+00

Table 5: Continued.

	PBSPSO	DEBBO	DDBLPSO
$f8(x)$			
Min	3.1112 E+04	**5.3259 E+03**	1.1820 E+04
Mean	1.4377 E+05	7.7449 E+04	**4.5841 E+04**
Median	1.1602 E+05	4.8418 E+04	**3.1882 E+04**
Std	1.0130 E+05	1.0253 E+05	**3.5138 E+04**
$f9(x)$			
Min	1.0043 E+03	**1.0022 E+03**	1.0024 E+03
Mean	1.0677 E+03	**1.0031 E+03**	1.0037 E+03
Median	1.0061 E+03	**1.0025 E+03**	1.0037 E+03
Std	1.2610 E+02	2.0847 E+00	**5.1535E-01**
$f10(x)$			
Min	2.0553 E+04	**1.1722 E+04**	2.1702 E+04
Mean	1.6465 E+05	1.2081E+05	**1.0437 E+05**
Median	1.1912E+05	8.6437 E+04	**7.1822 E+04**
Std	1.3543 E+05	1.0541 E+05	**1.0132 E+05**
$f11(x)$			
Min	1.4049 E+03	1.5319 E+03	**1.4013 E+03**
Mean	1.6647 E+03	1.6533 E+03	**1.4753 E+03**
Median	1.4082 E+03	1.6380 E+03	**1.4021 E+03**
Std	3.0497 E+02	**6.1809 E+01**	1.1947E+02
$f12(x)$			
Min	1.3108 E+03	1.3052 E+03	**1.3040 E+03**
Mean	1.3152 E+03	**1.3061 E+03**	1.3071 E+03
Median	1.3139 E+03	**1.3058 E+03**	1.3075 E+03
Std	4.6593 E+00	**1.1863 E+00**	1.4291 E+00
$f13(x)$			
Min	1.4153 E+03	**1.3928 E+03**	1.3995 E+03
Mean	1.4281 E+03	**1.3999 E +03**	1.4173 E+03
Median	1.4289 E+03	**1.3996 E+03**	1.4164 E+03
Std	5.1713 E+00	**4.3428 E+00**	7.2068E+00
$f14(x)$			
Min	3.2641 E+04	3.2795 E+04	**1.5000 E+03**
Mean	3.6568 E+04	3.5561 E+04	**3.2591 E+04**
Median	3.7333 E+04	3.5346 E+04	**3.3648 E+04**
Std	2.0662 E+03	**9.2865 E+02**	5.9283 E+03
$f15(x)$			
Min	1.6034 E+03	**1.6000 E+03**	**1.6000 E+03**
Mean	1.6076 E+03	**1.6000 E+03**	1.6004 E+03
Median	1.6048 E+03	**1.6000 E+03**	**1.6000 E+03**
Std	4.8441 E+00	**1.0394E-02**	2.3806 E+00

strategy guides particle to learn from other particles not worse than itself, which leads a greater convergence rate of DDBLPSO. At the same time, DDBLPSO has a slightly better ability of jumping out of local optimum than other competitors because of simple mutation and complexity of functions. One the other hand, learning from the better particles means that the exploration ability of algorithm will be reduced slightly. For this reason, DDBLPSO does not have such eye-catching performance on multimodal functions and composition functions. Table 7 shows that DDBLPSO attains the best rank, BLPSO attains the second, and CLPSO attains the third, followed by BBO and PSO. It is obvious that DDBLPSO is the most competitive algorithm in this group of experimental comparison.

Table 4 reveals that DDBLPSO performs much better than other four algorithms on $f1$, $f4$, $f5$, $f6$, $f8$, $f9$, $f10$, and $f12$. DDBLPSO cannot achieve obvious dominance on each kind of functions due to the problems of being different types and large scale. On the whole, DDBLPSO still performs best. Table 8 presents the same ranking result as Table 7. Figure 2 shows that DDBLPSO has the strongest evolving trend on $f1$, $f2$, $f4$, $f6$, $f7$, $f8$, $f10$, and $f14$. DDBLPSO is also the most competitive algorithm in this group of experimental comparison with relatively large-scale problems.

Generally speaking, it is observed from Tables 3, 4, 7 and 9 and Figure 2 that DDBLPSO achieves the best performance compared with its competitors.

Table 5 shows that DDBLPSO beats PBSPSO and DEBBO on most functions, and Table 9 shows that DDBLPSO attains the best rank, DEBBO attains the second, and PBSPSO attains the third. Table 6 reveals that DDBLPSO

Table 6: Performance comparison of three algorithms in experiment four.

	PBSPSO	DEBBO	DDBLPSO
$f1(x)$			
Min	2.8403 E+07	1.2931 E+08	**8.8196 E+06**
Mean	3.5153 E+07	1.7284 E+08	**1.1151 E+07**
Median	3.4872 E+07	1.7385 E+08	**1.1181 E+07**
Std	4.0420 E+06	2.4193 E+07	**1.0671 E+06**
$f2(x)$			
Min	5.3463 E+08	**2.0005 E+02**	2.3631 E+07
Mean	1.2518 E+08	**1.6110 E+03**	3.2844 E+07
Median	1.3096 E+08	**1.0518 E+03**	3.2473 E+07
Std	2.6223 E+07	**1.6068 E+03**	4.2897 E+06
$f3(x)$			
Min	3.2131 E+02	**3.2073E+02**	3.2136 E+0
Mean	3.2144 E+02	**3.2080E+02**	3.2145 E+02
Median	3.2144 E+02	**3.2083E+02**	3.2145 E+02
Std	6.3151E−02	4.2976E−02	**3.9328 E−02**
$f4(x)$			
Min	9.1895 E+02	7.2243 E+02	**4.8020 E+02**
Mean	1.0124 E+03	7.6390 E+02	**5.0111 E+02**
Median	1.0051 E+03	7.5774 E+02	**4.9888 E+02**
Std	4.9600 E+01	2.5379 E+01	**1.0271 E+01**
$f5(x)$			
Min	1.2411 E+04	1.6916 E+04	**4.7234 E+03**
Mean	1.4988 E+04	1.7798 E+04	**5.5794 E+03**
Median	1.4942 E+04	1.7938 E+04	**5.3980 E+03**
Std	1.2888 E+03	7.2976 E+02	**6.1390 E+02**
$f6(x)$			
Min	2.1414 E+06	2.3652 E+07	**1.3833 E+06**
Mean	3.6713 E+06	3.3346 E+07	**1.9272 E+06**
Median	3.5486 E+06	3.4022 E+07	**1.9522 E+06**
Std	9.3569 E+05	5.1752 E+06	**2.6129 E+05**
$f7(x)$			
Min	7.4025 E+02	8.4813 E+02	**7.3350 E+02**
Mean	**7.8440 E+02**	8.4978 E+02	8.0765 E+02
Median	**7.7255 E+02**	8.4967 E+02	8.0020 E+02
Std	3.6466 E+01	**8.3256E−01**	2.8476 E+01
$f8(x)$			
Min	1.2536 E+06	1.0239 E+07	**5.1323 E+05**
Mean	1.9625 E+06	1.7527 E+07	**6.9234 E+05**
Median	1.9220 E+06	1.7616 E+07	**6.8432 E+05**
Std	3.8544 E+05	4.3476 E+06	**1.3364 E+05**
$f9(x)$			
Min	1.0124 E+03	**1.0073 E+03**	1.0068 E+03
Mean	1.2043E+03	**1.0176 E+03**	1.0077 E+03
Median	1.0155 E+03	**1.0175 E+03**	1.0076 E+03
Std	3.1718 E+02	**2.2394E−01**	5.2679E−01
$f10(x)$			
Min	1.6908 E+06	**5.3905 E+03**	1.4355 E+04
Mean	3.7009 E+06	**6.7693 E+03**	1.6831 E+04
Median	3.3489 E+06	**6.4983 E+03**	1.6362 E+04
Std	1.7822 E+06	**1.0052 E+03**	2.1156 E+03
$f11(x)$			
Min	1.4281 E+03	1.7210 E+03	**1.4102 E+03**
Mean	2.1024 E+03	2.8620 E+03	**1.6127 E+03**
Median	1.4369 E+03	3.1818 E+03	**1.4126 E+03**
Std	8.3776 E+02	7.4285 E+02	**4.5786 E+02**
$f12(x)$			
Min	1.3512 E+03	1.3172 E+03	**1.3111 E+03**
Mean	1.3720 E+03	1.3176 E+03	**1.3147 E+03**

Table 6: Continued.

	PBSPSO	DEBBO	DDBLPSO
Median	1.3688 E+03	1.3176 E+03	**1.3149 E+03**
Std	1.6991 E+01	**2.6835E−01**	1.3710 E+00
$f13(x)$			
Min	1.7474 E+03	**1.6929 E+03**	1.7344 E+03
Mean	1.7625 E+03	**1.7138 E+03**	1.7481 E+03
Median	1.7631 E+03	**1.7158 E+03**	1.7487 E+03
Std	7.3313 E+00	7.9928 E+00	**6.3159 E+00**
$f14(x)$			
Min	1.5392 E+04	1.1024 E+05	**3.1265 E+03**
Mean	1.4095 E+05	**1.1027 E+05**	1.1036 E+05
Median	1.4401 E+05	**1.1027 E+05**	1.1770 E+05
Std	2.7916 E+04	**1.9162 E+01**	2.9479 E+04
$f15(x)$			
Min	1.6084 E+03	**1.6000 E+03**	1.6031 E+03
Mean	1.6140 E+03	**1.6000 E+03**	1.6037 E+03
Median	1.6138 E+03	**1.6000 E+03**	1.6038 E+03
Std	3.8763 E+00	**2.3967E−02**	3.1219E−01

Table 7: Ranks of five algorithms according to Friedman test in experiment one.

	PSO	BBO	CLPSO	BLPSO	DDBLPSO
Friedman rank	4.15	3.93	2.63	2.44	1.85
Final rank	5	4	3	2	1

Table 8: Ranks of five algorithms according to Friedman test in experiment two.

	PSO	BBO	CLPSO	BLPSO	DDBLPSO
Friedman rank	4.29	3.48	3.21	2.37	1.65
Final rank	5	4	3	2	1

Table 9: Ranks of three algorithms according to Friedman test in experiment three.

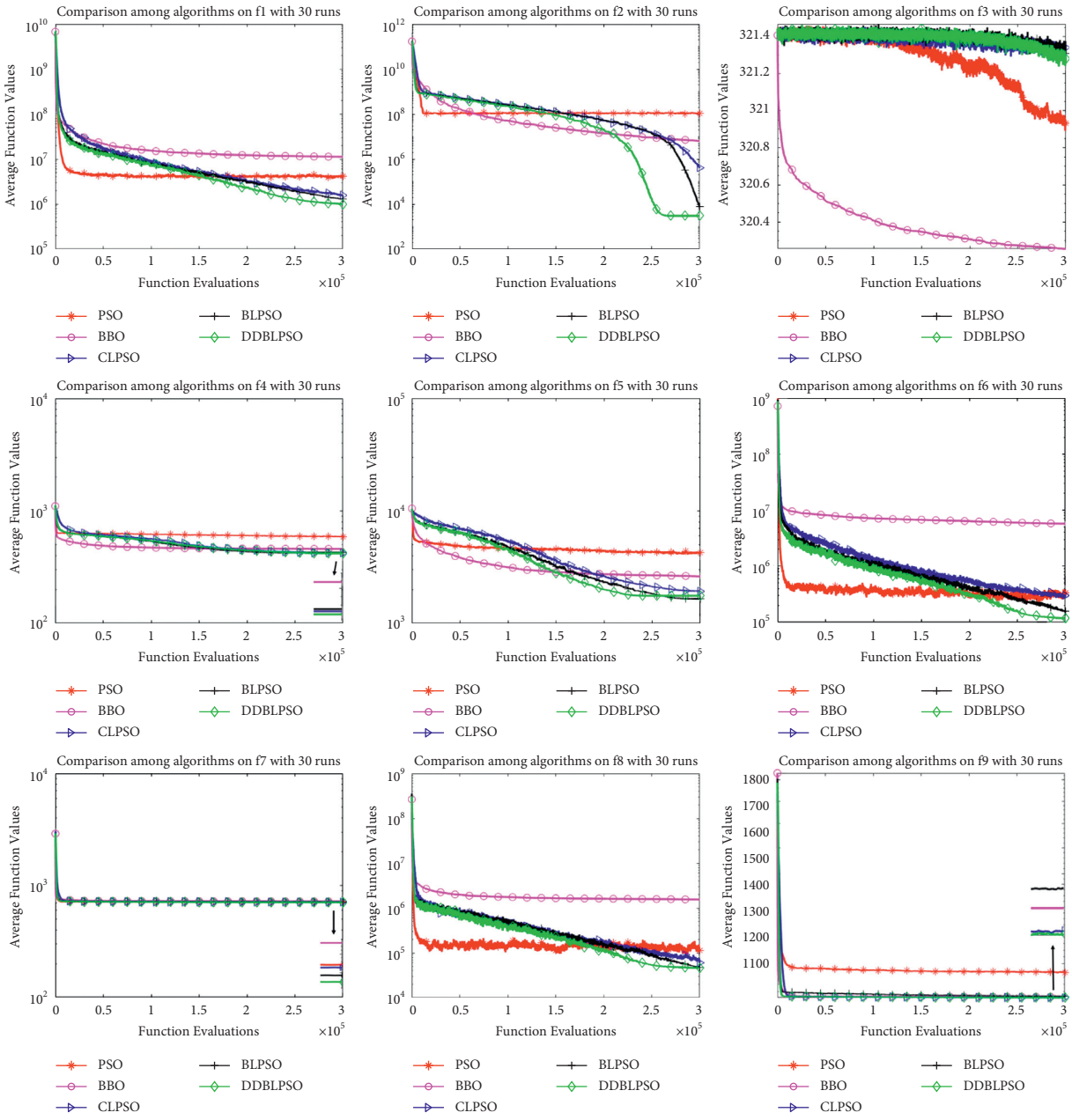
	PBSPSO	DEBBO	DDBLPSO
Friedman rank	2.78	1.73	1.48
Final rank	3	2	1

Table 10: Ranks of three algorithms according to Friedman test in experiment four.

	PBSPSO	DEBBO	DDBLPSO
Friedman rank	2.52	1.95	1.53
Final rank	3	2	1

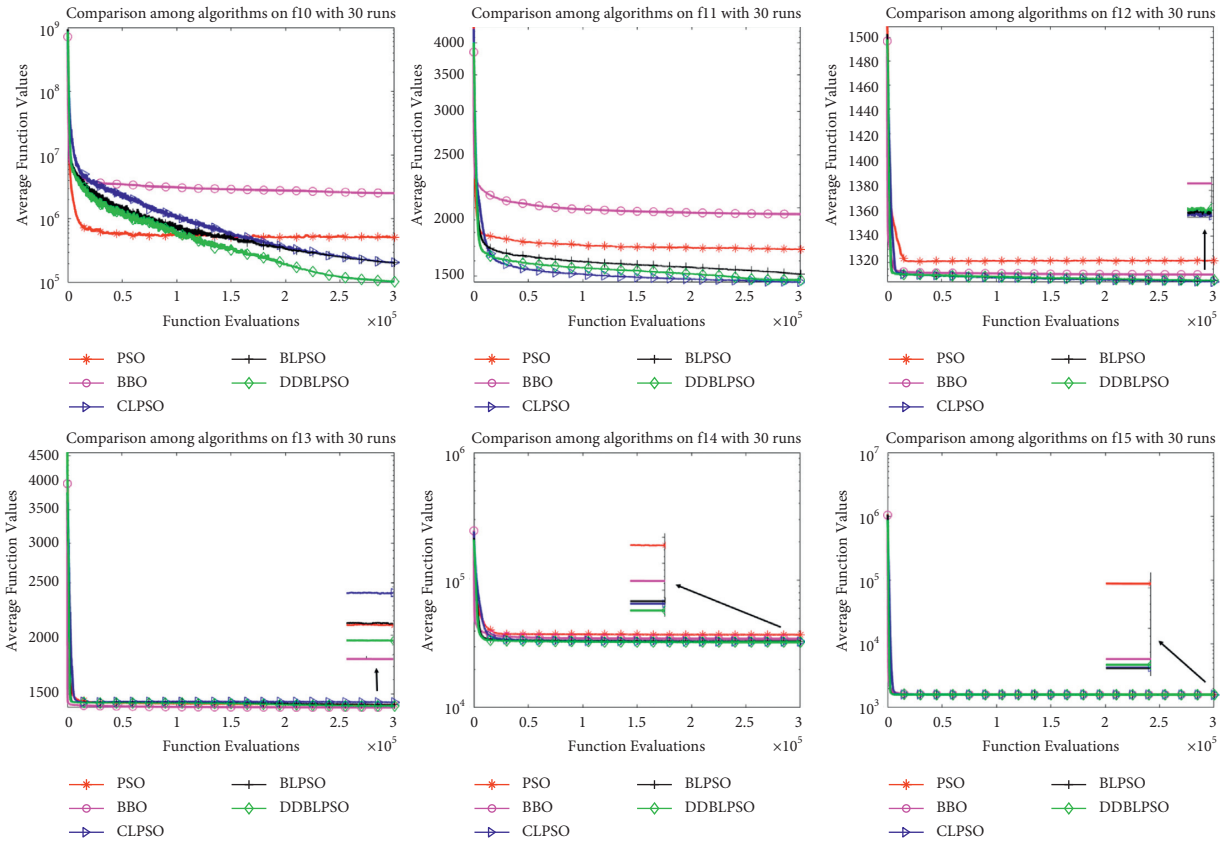
Table 11: Design of experiment five, experiment six, experiment seven, and experiment eight.

Experiment	Population size N	Points number mp	Comparison algorithm
Experiment five	30	14	PSO, BBO, CLPSO, BLSPSO
Experiment six	100	49	PSO, BBO, CLPSO, BLSPSO
Experiment seven	30	14	PBSPSO, DEBBO
Experiment eight	100	49	PBSPSO, DEBBO



(a)

Figure 2: Continued.



(b)

Figure 2: Converging curves of the average best fitness in experiment one.

performs much better than other two algorithms on $f1, f4, f5, f6, f8, f11$, and $f12$. Table 10 presents the same ranking result as Table 9. Figure 3 shows that DDBLPSO has the strongest evolving trend on $f1, f2, f4, f5, f6, f7, f8, f10, f11$, and $f14$. Generally speaking, it is observed from Tables 5, 6, 9 and 10 and Figure 3 that DDBLPSO beats a PSO variant and a BBO variant.

6. Applications in UAV Path Planning

6.1. Six Kinds of Terrain Functions. City, village without houses, village with houses, mountainous area without houses, mountainous area with houses, and mountainous area with a huge building are the most common terrains in UAV path planning. Six kinds of terrain functions are designed depending on these conditions. City has flat ground and tall buildings. Village without houses means uneven ground. Village with houses means uneven ground and low buildings. Mountainous area without houses means uneven ground and mountains. Mountainous area with houses has uneven ground, mountains, and low buildings. Mountainous area with a huge building has uneven ground, mountains, and a huge building. (x', y') is the coordinate of any point on the plane. The details of six terrain functions are as follows.

The terrain function of city f_{terrain1} is defined as follows:

$$f_{\text{terrain1}} = \begin{cases} T_i, & \text{if } (x', y') \in \Omega_i, i = 1, 2, \ldots, T, \\ 0, & \text{if } (x', y') \in \Omega_i, i = 1, 2, \ldots, T, \end{cases} \tag{19}$$

where T_i is the height of the tall building i, Ω_i is the area occupied by the tall building i, and T is the number of tall buildings.

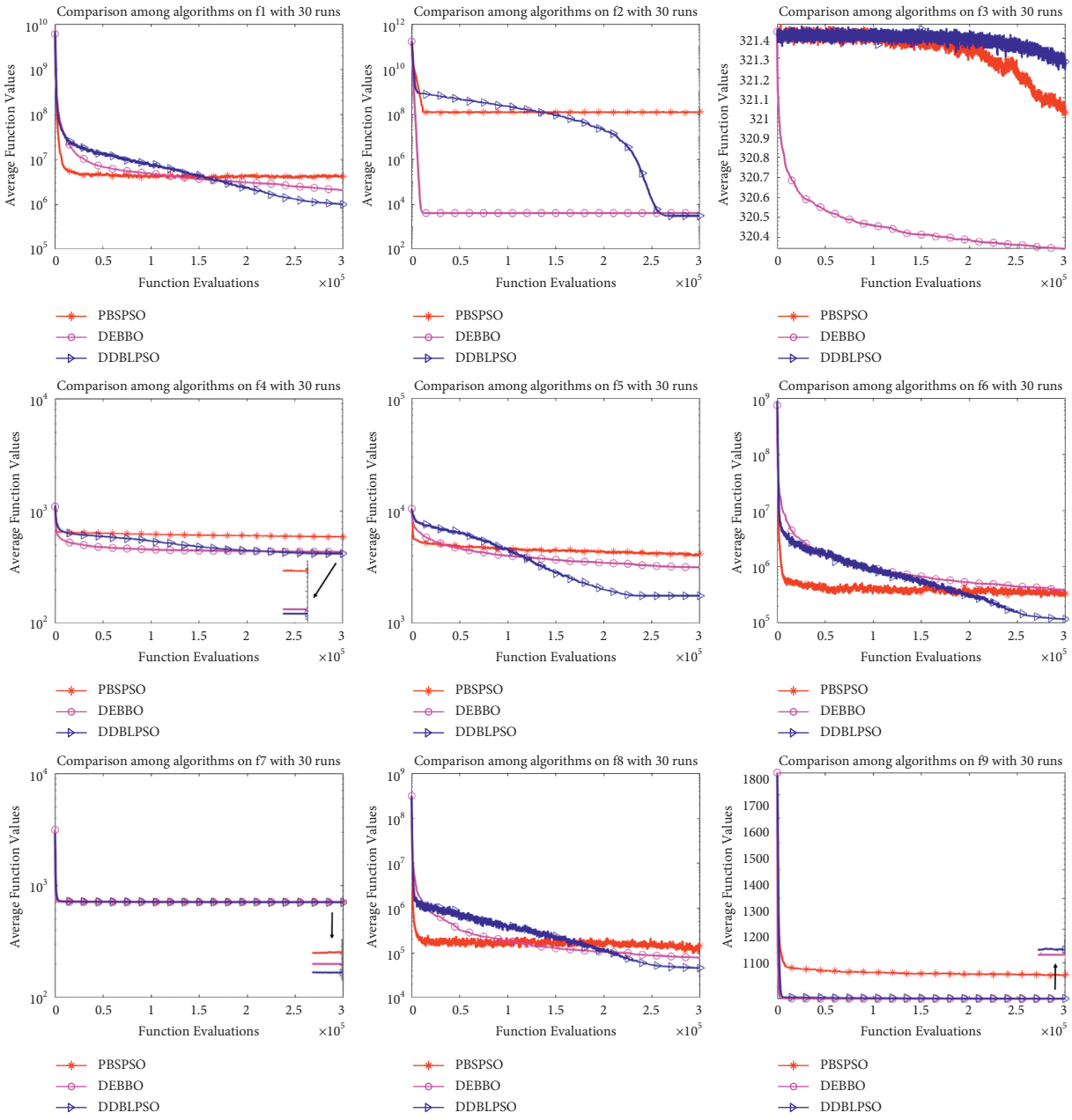
The terrain function of village without houses f_{terrain2} is defined as follows:

$$\begin{aligned} f_{\text{terrain2}} = {} & \sin(x' + a_1) + a_2 * \sin(y') + \cos(y' + a_3) \\ & + a_4 * \cos(x') \\ & + \sin\left(\sqrt{x'^2 + y'^2} + a_5\right) + a_6 * \cos\left(a_6 * \sqrt{x'^2 + y'^2}\right), \end{aligned} \tag{20}$$

where a_1, a_2, a_3, a_4, a_5, and a_6 are terrain parameters, which decide the uneven degree of the terrain.

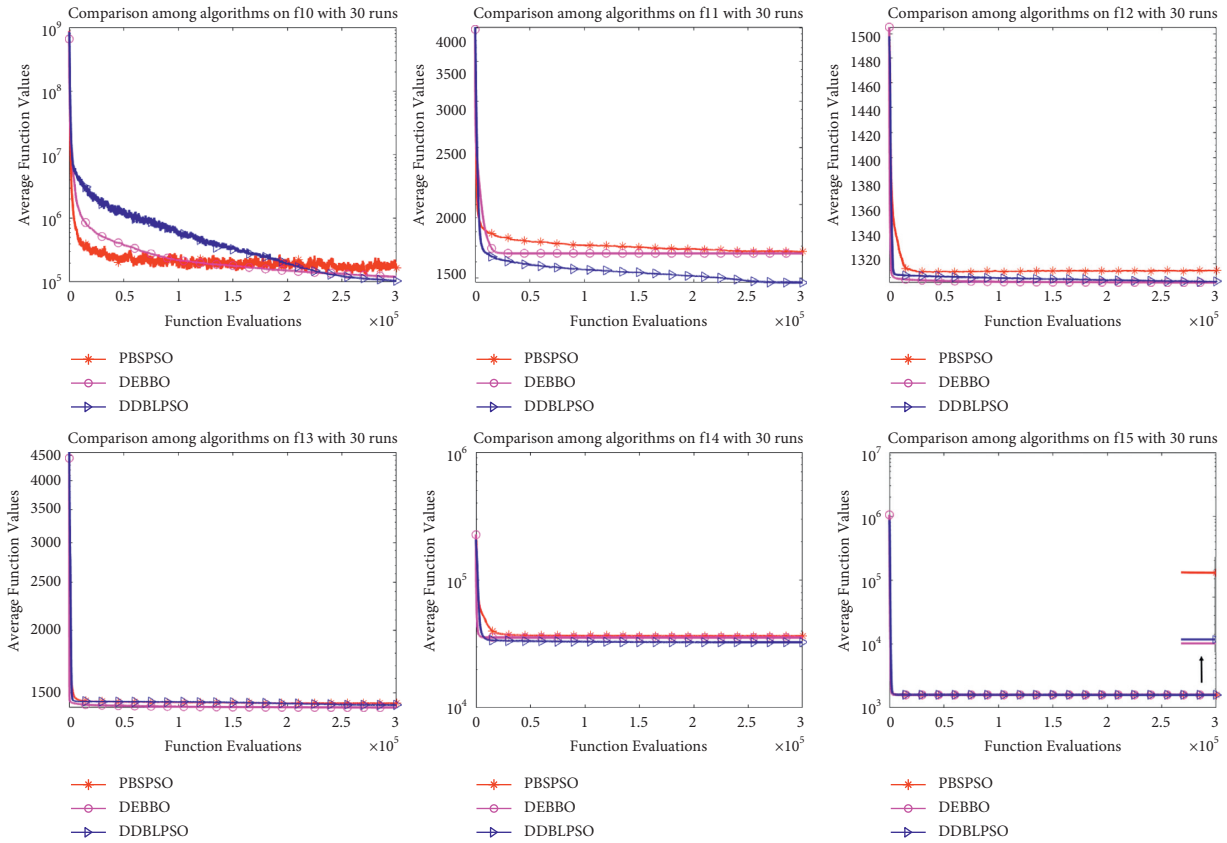
The terrain function of village with houses f_{terrain3} is defined as follows:

$$f_{\text{terrain3}} = \begin{cases} \max\{S_i, f_{\text{terrain2}}\}, & \text{if } (x', y') \in \Gamma_i, i = 1, 2, \ldots, S, \\ f_{\text{terrain2}}, & \text{if } (x', y') \notin \Gamma_i, i = 1, 2, \ldots, S, \end{cases} \tag{21}$$



(a)

Figure 3: Continued.



(b)

Figure 3: Converging curves of the average best fitness in experiment three.

Table 12: Performance comparison of five algorithms in experiment five.

	BBO	PSO	CLPSO	BLPSO	DDBLPSO
costfun1					
Min	0.333786	0.361611	0.333033	0.333038	**0.332735**
Mean	0.354893	0.408034	0.348755	0.347850	**0.338909**
Median	0.348119	0.401855	0.346445	0.340743	**0.335515**
Std	0.020209	0.035911	0.019765	0.020109	**0.005750**
costfun2					
Min	0.333514	0.358298	0.332679	0.332662	**0.332363**
Mean	0.338082	0.400207	0.333074	0.333387	**0.332846**
Median	0.336249	0.392837	0.333044	0.333043	**0.332717**
Std	0.005453	0.030524	0.000334	**0.000680**	0.000376
costfun3					
Min	0.333434	0.363159	0.332662	0.332662	**0.332470**
Mean	0.338407	0.396183	0.333221	0.333186	**0.332957**
Median	0.336943	0.385252	0.333044	0.333043	**0.332687**
Std	0.004669	0.028417	0.000525	**0.000418**	0.001214
costfun4					
Min	0.333579	0.354768	0.332679	0.332679	**0.332454**
Mean	0.336364	0.400745	0.333237	0.333581	**0.333045**
Median	0.335141	0.396060	0.333043	0.333043	**0.332773**
Std	0.003654	0.032782	0.001182	0.001667	**0.001155**
costfun5					
Min	0.333159	0.358298	0.332679	0.332662	**0.332422**
Mean	0.337872	0.399360	0.333108	0.333139	**0.332759**

Table 12: Continued.

	BBO	PSO	CLPSO	BLPSO	DDBLPSO
Median	0.336249	0.391714	0.333043	0.333043	**0.332678**
Std	0.005734	0.030770	0.000442	0.000558	**0.000374**
costfun6					
Min	0.334377	0.362532	0.332445	0.332692	**0.332414**
Mean	0.340639	0.395483	0.333808	0.333623	**0.333442**
Median	0.337852	0.386125	0.334080	0.333738	**0.333598**
Std	0.008596	0.026440	0.000625	**0.000484**	0.000673

Table 13: Performance comparison of five algorithms in experiment six.

	BBO	PSO	CLPSO	BLPSO	DDBLPSO
costfun1					
Min	0.409522	0.452563	0.407823	0.390225	**0.384962**
Mean	0.485857	0.539211	0.423850	0.417825	**0.417211**
Median	0.478369	0.545726	0.414287	0.407063	**0.394661**
Std	0.047892	0.037861	**0.020093**	0.026866	0.037308
costfun2					
Min	0.380857	0.464534	0.375254	0.366452	**0.363554**
Mean	0.388581	0.529762	0.380145	0.371376	**0.368538**
Median	0.388856	0.531348	0.380259	0.371162	**0.368096**
Std	0.003426	0.035597	**0.002097**	0.002428	0.002839
costfun3					
Min	0.380579	0.464561	0.376183	0.366838	**0.363259**
Mean	0.390465	0.530220	0.380478	0.370835	**0.369487**
Median	0.389378	0.526718	0.381116	0.370512	**0.368922**
Std	0.004844	0.042232	0.003074	**0.001831**	0.003726
costfun4					
Min	0.382432	0.472434	0.374150	0.367105	**0.362359**
Mean	0.390443	0.523642	0.380130	0.370614	**0.368372**
Median	0.390181	0.522455	0.380235	0.370568	**0.368463**
Std	0.003990	0.030173	0.002600	**0.001943**	0.002333
costfun5					
Min	0.381921	0.436542	0.375358	0.367693	**0.364425**
Mean	0.389006	0.524626	0.380941	0.371797	**0.368500**
Median	0.389857	0.531651	0.380743	0.371605	**0.367672**
Std	0.003858	0.031882	**0.002198**	0.003142	0.003034
costfun6					
Min	0.382586	0.459559	0.377788	0.368291	**0.363963**
Mean	0.392547	0.517871	0.383203	0.372213	**0.369134**
Median	0.391984	0.523437	0.383597	0.371327	**0.369174**
Std	0.006611	0.033132	0.003701	0.002866	**0.002664**

where S_i is the height of the low building i, Γ_i is the area occupied by the low building i, and S is the number of low buildings.

The terrain function of mountainous area without houses f_{terrain4} is defined as follows:

$$M_i = R_i * \exp\left[-\left(\frac{x' - P_i}{K_i}\right)^2 - \left(\frac{y' - Q_i}{K_i}\right)^2\right], \quad i = 1, 2, \ldots, M,$$

$$f_{\text{terrain4}} = \max\{M_i, f_{\text{terrain2}}\}, \tag{22}$$

where M_i is the mountain function, R_i, R_i, Q_i, and K_i describe the position and the shape of mountain i, and M is the number of mountains.

The terrain function of mountainous area with houses f_{terrain5} is defined as follows:

$$f_{\text{terrain5}} = \begin{cases} \max\{S_i, f_{\text{terrain4}}\}, & \text{if } (x', y') \in \Gamma_i, i = 1, 2, \ldots, S, \\ f_{\text{terrain4}}, & \text{if } (x', y') \notin \Gamma_i, i = 1, 2, \ldots, S, \end{cases} \tag{23}$$

where S_i is the height of the low building i, Γ_i is the area occupied by the low building i, and S is the number of low buildings.

The terrain function of mountainous area with a huge building f_{terrain6} is defined as follows:

$$f_{\text{terrain6}} = \begin{cases} \max\{\text{Huge}, f_{\text{terrain4}}\}, & \text{if } (x', y') \in \Omega', \\ f_{\text{terrain4}}, & \text{if } (x', y') \notin \Omega', \end{cases} \tag{24}$$

Table 14: Performance comparison of three algorithms in experiment seven.

	PBSPSO	DEBBO	DDBLPSO
costfun1			
Min	0.333917	0.332859	**0.332735**
Mean	0.351685	0.341414	**0.338909**
Median	0.340660	0.335595	**0.335515**
Std	0.025157	0.010353	**0.005750**
costfun2			
Min	0.333801	**0.332310**	0.332363
Mean	0.338538	0.339693	**0.332846**
Median	0.337244	0.339361	**0.332717**
Std	0.004749	0.007640	**0.000376**
costfun3			
Min	0.333631	**0.332355**	0.332470
Mean	0.335685	0.340312	**0.332957**
Median	0.334800	0.338883	**0.332687**
Std	0.002172	0.006869	**0.001214**
costfun4			
Min	0.333443	0.332611	**0.332454**
Mean	0.336772	0.339582	**0.333045**
Median	0.334911	0.336338	**0.332773**
Std	0.005649	0.010488	**0.001155**
Costfun5			
Min	0.333852	**0.332310**	0.332422
Mean	0.338636	0.339687	**0.332759**
Median	0.337244	0.339196	**0.332678**
Std	0.004892	0.007473	**0.000374**
costfun6			
Min	0.334083	0.333078	**0.332414**
Mean	0.337138	0.342282	**0.333442**
Median	0.336396	0.339118	**0.333598**
Std	0.002739	0.012053	**0.000673**

where Huge is the height of the huge building and Ω' is the area occupied by the huge building.

6.2. Parameter Settings. Cost functions with city, village without houses, village with houses, mountainous area without houses, mountainous area with houses, and mountainous area with a huge building are labelled as costfun1 – costfun6. Four sets of UAV path planning experiments are conducted which are labelled as experiment five, experiment six, experiment seven, and experiment eight. Design of the four experiments are shown in Table 11. The boldface in Tables 12–15 indicates the best experimental results in experiment five to eight, respectively. Points number means the number of points in each path without starting point and end point. Dimension is equal to twice the point number. The path is divided into mp + 1 small lines by mp points from the starting point to the end point. To simplify the calculation, if a point is in the danger zone, it is assumed that the halves of both small lines before and after and adjacent to the point are also in the danger zone. It is the same for a point under the ground or under a building. UAV search space is $[-50, 50]*[-50, 50]*[0, 25]$, terrain parameters $a_1 = 10$, $a_2 = 1$, $a_3 = 20$, $a_4 = 0.8$, $a_5 = 30$, $a_6 = 0.9$, weight coefficients $W_1 = W_2 = W_3 = W_4 = 1$, penalty constant $P = 3$, and a small positive number $Z_{\min} = 0.1$. Starting point and end point coordinates are (−50,−50, 0.1) and (50, 50, 0.1), respectively. The maximum velocity along Y axis and Z axis is 2.5 and 0.625, respectively. The details of tall, low, and huge buildings are given in Table 16. The detailed parameter settings of mountains are given in Table 17. Danger areas are designed as equation (25). Other parameter settings of algorithms are the same as in Section 5.

$$\begin{aligned} (x' + 25)^2 + (y' + 25)^2 + z'^2 &= 10^2, \\ (x' - 25)^2 + (y' - 25)^2 + z'^2 &= 10^2. \end{aligned} \tag{25}$$

6.3. Results and Discussion of UAV Path Planning. Tables 12–15 show the UAV path planning results of six kinds of terrains in experiment five, experiment six, experiment seven, and experiment eight, respectively. The ranks of five algorithms according to Friedman test of experiment three and experiment four are shown in Tables 18–21, respectively. Min, Mean, Median, and Std indicate the minimum function error value, the mean function error value, the median function error value, and the standard deviation of error values, respectively. In order to exhibit the evolution trend of five algorithms more vividly, the converging curves of the average best fitness in experiment five and experiment seven

Table 15: Performance comparison of three algorithms in experiment eight.

	PBSPSO	DEBBO	DDBLPSO
costfun1			
Min	0.408638	0.740178	**0.384962**
Mean	0.507099	0.855135	**0.417211**
Median	0.500030	0.857994	**0.394661**
Std	0.057598	**0.031987**	0.037308
costfun2			
Min	0.383974	0.373691	**0.363554**
Mean	0.392406	0.400978	**0.368538**
Median	0.392086	0.398095	**0.368096**
Std	0.005132	0.016330	**0.002839**
costfun3			
Min	0.383351	**0.362185**	0.363259
Mean	0.394280	0.402638	**0.369487**
Median	0.393606	0.402676	**0.368922**
Std	0.004780	0.020725	**0.003726**
costfun4			
Min	0.384865	0.368213	**0.362359**
Mean	0.393153	0.407062	**0.368372**
Median	0.392543	0.406902	**0.368463**
Std	0.005216	0.020424	**0.002333**
costfun5			
Min	0.385425	**0.363051**	0.364425
Mean	0.392956	0.409671	**0.368500**
Median	0.393330	0.414376	**0.367672**
Std	0.004038	0.019970	**0.003034**
costfun6			
Min	0.386878	0.379041	**0.363963**
Mean	0.393760	0.407583	**0.369134**
Median	0.394691	0.408277	**0.369174**
Std	0.003657	0.016136	**0.002664**

Table 16: Parameter settings of tall, short, and huge buildings.

Tall buildings	[−20,−10] ∗ [5,15] ∗ [0,16] [6,15] ∗ [−8,3] ∗ [0,14]	[10,20] ∗ [5,15] ∗ [0,15] [−14,-3] ∗ [−20,-10] ∗ [0,14]	[−15,−6] ∗ [−8,3] ∗ [0,15] [3,14] ∗ [−2,−10] ∗ [0,16]
Short buildings	[−20,−10] ∗ [5,15] ∗ [0,6] [6,15] ∗ [−8,3] ∗ [0,4]	[10,20] ∗ [5,15] ∗ [0,5] [−14,−3] ∗ [−20,−10] ∗ [0,4]	[−15,−6] ∗ [−8,3] ∗ [0,5] [3,14] ∗ [−20,−10] ∗ [0,6]
Huge building	[−45,45] ∗ [−45,45] ∗ [0,20]		

Table 17: Parameter settings of tall, short, and huge buildings.

Mountains	$R_1 = 20$, $P_1 = 0, Q_1 = 30$, $K_1 = 5$ $R_3 = 21$, $P_3 = 30$, $Q_3 = 0$, $K_3 = 3$	$R_2 = 18$, $P_2 = 0$, $Q_2 = -30$, $K_2 = 2$ $R_4 = 19$, $P_4 = -30$, $Q_4 = 0$, $K_4 = 4$

are shown in Figures 4 and 5, respectively. The converging curves of six kinds of terrain functions in experiment six and experiment eight are not shown due to the similar converging behaviours with experiment five and experiment seven, respectively.

Table 12 shows that DDBLPSO is the most competitive algorithm for the relatively small-scale problems with 14 points. It is easy to see that DDBLPSO attains the best rank and BLPSO is the second, followed by CLPSO, PSO, and BBO in Table 18. For the relatively large-scale problems with 49 points, Tables 13 and 19 reveal that DDBLPSO achieves the best results in experiment six. Table 19 shows that DDBLPSO attains the best rank, BLPSO is the second, and CLPSO is the third, followed by PSO and BBO. Furthermore, the experimental results indicate that it is much better than other competitors. However, Tables 12 and 13 show that DDBLPSO has relatively larger Std, which indicates a possible enhancement on the convergence capability of DDBLPSO. Figure 4 shows that DDBLPSO has the strongest evolving trend on all the terrain functions for

Table 18: Ranks of five algorithms according to Friedman test in experiment five.

	PSO	BBO	CLPSO	BLPSO	DDBLPSO
Friedman rank	4.00	5.00	2.42	2.38	1.21
Final rank	4	5	3	2	1

Table 19: Ranks of five algorithms according to Friedman test in experiment six.

	PSO	BBO	CLPSO	BLPSO	DDBLPSO
Friedman rank	4.04	4.96	2.71	1.96	1.33
Final rank	4	5	3	2	1

Table 20: Ranks of three algorithms according to Friedman test in experiment seven.

	PBSPSO	DEBBO	DDBLPSO
Friedman rank	2.38	2.50	1.13
Final rank	2	3	1

Table 21: Ranks of three algorithms according to Friedman test in experiment eight.

	PBSPSO	DEBBO	DDBLPSO
Friedman rank	2.25	2.63	1.13
Final rank	2	3	1

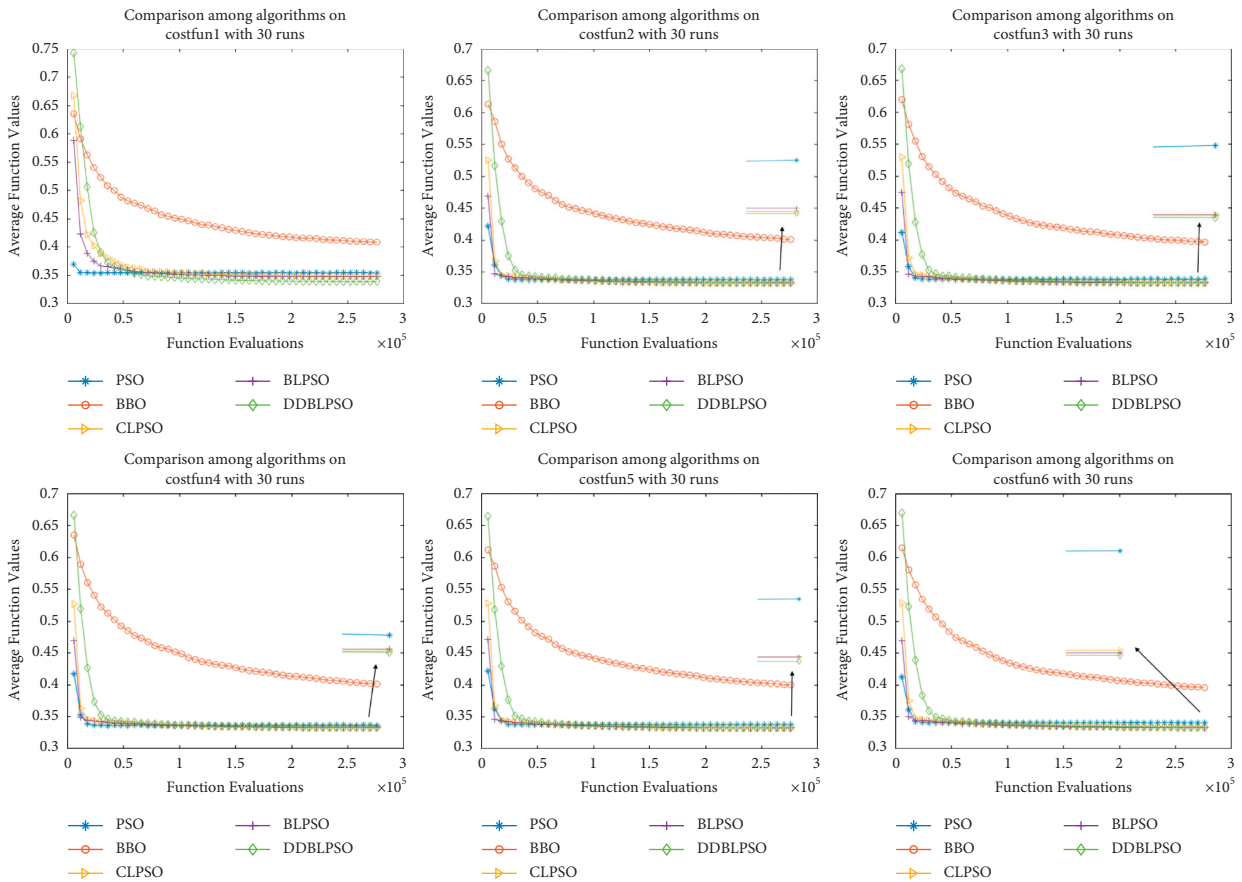


Figure 4: Converging curves of different terrain functions in experiment five.

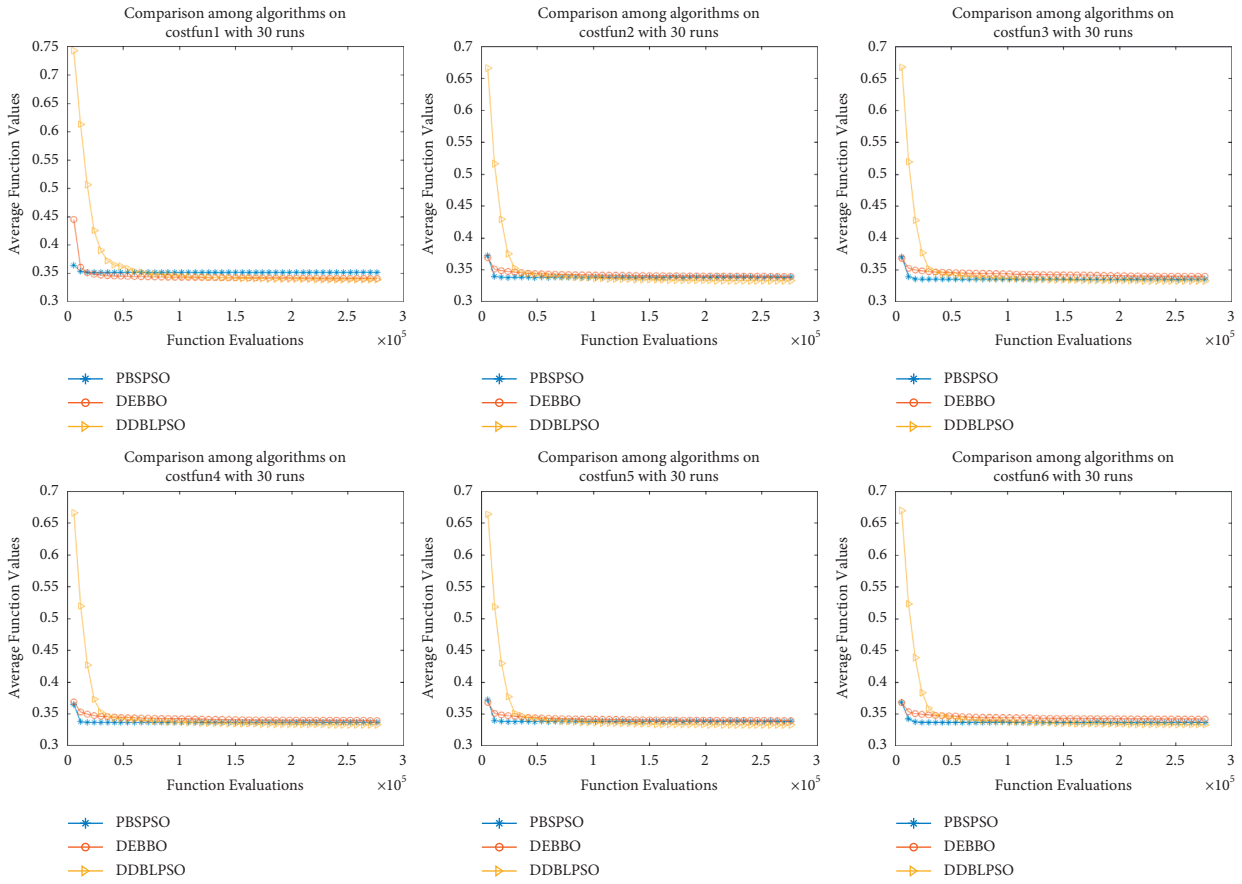


Figure 5: Converging curves of different terrain functions in experiment seven.

the relatively large-scale problem in experiment five. Generally speaking, it can be observed from Tables 12, 13, 18, 19 and Figure 4 that DDBLPSO performs very encouraging in UAV path planning, but there is still room for improvement in terms of Std.

It is easy to see that DDBLPSO beats DEBBO and PBSPSO for the relatively small-scale problems with 14 points in Table 14. Table 15 shows that DDBLPSO attains the best rank, DEBBO is the second, and PBSPSO is the third in small-scale problems of UAV path planning. Figure 5 shows that DDBLPSO has the stronger evolving trend on all the terrain functions than DEBBO and PBSPSO in experiment seven. Tables 17 and 21 reveal the same results in large-scale problems.

7. Conclusions

This paper presents a novel UAV path planning algorithm based on double-dynamic biogeography-based learning particle swarm optimization. DDBLPSO adopts double-dynamic biogeography-based learning strategy. Under this strategy, particle only learns from itself or other even better individuals. Six terrain functions are presented to simulate the most six common terrains in UAV path planning. Computational experiments and simulations demonstrate the performance advantages of the algorithm both in global optimization and UAV path planning. Especially for the relatively large-scale problems, it performs even more competitive.

Our algorithm has the potential to be applied to other problems such as oil exploration, scheduling, and deep learning. The UAV path planning problem also can be extended to multiobjective or constrained optimization problems to be suitable for more complex situations.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Authors' Contributions

Yisheng Ji was responsible for novelty construction and verification, simulation design, and manuscript writing. Xinchao Zhao was responsible for research motivation, simulation design, and analysis. Junling Hao read through the manuscript and gave some suggestions on structure and representation.

Acknowledgments

The authors are grateful to Prof. Xingquan Zuo from School of Computer Science, Beijing University of Posts and Telecommunications, for his suggestions on research motivation and simulation. This research was supported by the Beijing Natural Science Foundation (1202020) and National Natural Science Foundation of China (61973042).

References

[1] I. K. Nikolos, K. P. Valavanis, N. C. Tsourveloudis, and A. N. Kostaras, "Evolutionary algorithm based offline/online path planner for uav navigation," *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, vol. 33, no. 6, pp. 898–912, 2003.

[2] Y. Fu, M. Ding, and C. Zhou, "Phase Angle-Encoded and Quantum-Behaved Particle Swarm Optimization Applied to Three-Dimensional Route Planning for UAV," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 42, no. 2, pp. 511–526, 2012.

[3] Y. V. Pehlivanoglu, "A new vibrational genetic algorithm enhanced with a Voronoi diagram for path planning of autonomous UAV," *Aerospace Science and Technology*, vol. 16, no. 1, pp. 47–55, 2012.

[4] X. Zhang and H. Duan, "An improved constrained differential evolution algorithm for unmanned aerial vehicle global route planning," *Applied Soft Computing*, vol. 26, pp. 270–284, 2015.

[5] E. Besada-Portas, L. de la Torre, A. Moreno, and J. L. Risco-Martín, "On the performance comparison of multi-objective evolutionary UAV path planners," *Information Sciences*, vol. 238, pp. 111–125, 2013.

[6] B. Zhang and H. Duan, "Three-Dimensional Path Planning for Uninhabited Combat Aerial Vehicle Based on Predator-Prey Pigeon-Inspired Optimization in Dynamic Environment," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 14, no. 1, pp. 97–107, 2017.

[7] YB Chen, YS Mei, and JQ Yu, "Three-dimensional unmanned aerial vehicle path planning using modified wolf pack search algorithm[J]," *Neurocomputing*, vol. 266, pp. 445–457, 2017.

[8] Z. Sun, J. Wu, J. Yang, Y. Huang, C. Li, and D. Li, "Path Planning for GEO-UAV Bistatic SAR Using Constrained Adaptive Multiobjective Differential Evolution," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 11, pp. 6444–6457, 2016.

[9] V. Roberge, M. Tarbouchi, and G. Labonte, "Comparison of Parallel Genetic Algorithm and Particle Swarm Optimization for Real-Time UAV Path Planning," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 132–141, 2013.

[10] RC Eberchart and J Kennedy, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks IEEE*, pp. 1942–1948, WA, Australia, December 1995.

[11] ZH Zhi-Hui Zhan, J Jun Zhang, Y Yun Li, and H. S.-H. Chung, "Adaptive Particle Swarm Optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 6, pp. 1362–1381, 2009.

[12] A. Nickabadi, M. M. Ebadzadeh, and R. Safabakhsh, "A novel particle swarm optimization algorithm with adaptive inertia weight," *Applied Soft Computing*, vol. 11, no. 4, pp. 3658–3670, 2011.

[13] XD Xiaodong Li and X Xin Yao, "Cooperatively Coevolving Particle Swarms for Large Scale Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 2, pp. 210–224, 2012.

[14] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

[15] X. Zhao, Y. Ji, and J. Hao, "A Novel Biogeography-based Optimization Algorithm with Momentum Migration and Taxonomic Mutation," in *The Eleventh International Conference on Swarm Intelligence (ICSI 2020)*, vol. 12145, pp. 83–93, Belgrade, Serbia, 2020.

[16] X. Zhao, W. Lin, J. Hao, X. Zuo, and J. Yuan, "Clustering and pattern search for enhancing particle swarm optimization with Euclidean spatial neighborhood search," *Neurocomputing*, vol. 171, pp. 966–981, 2016.

[17] S.-W. Lin, K.-C. Ying, S.-C. Chen, and Z.-J. Lee, "Particle swarm optimization for parameter determination and feature selection of support vector machines," *Expert Systems with Applications*, vol. 35, no. 4, pp. 1817–1824, 2008.

[18] R. K. Yadav and Anubhav, "PSO-GA based hybrid with Adam Optimization for ANN training with application in Medical Diagnosis," *Cognitive Systems Research*, vol. 64, pp. 191–199, 2020.

[19] R. Vafashoar and M. R. Meybodi, "Cellular learning automata based bare bones PSO with maximum likelihood rotated mutations," *Swarm and Evolutionary Computation*, vol. 44, pp. 680–694, 2019.

[20] G. Li and X. Jiao, "Synthesis and validation of finite time servo control with PSO identification for automotive electronic throttle," *Nonlinear Dynamics*, vol. 90, no. 2, pp. 1165–1177, 2017.

[21] Z Xiang, D Ji, H Zhang, W Hongrun, and L Yuanxiang, "A simple PID-based strategy for particle swarm optimization algorithm," *Information Sciences*, vol. 502, 2019.

[22] D. Simon, "Biogeography-Based Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702–713, 2008.

[23] X. Chen, H. Tianfield, C. Mei, W. Du, and G. Liu, "Biogeography-based learning particle swarm optimization," *Soft Computing*, vol. 21, no. 24, pp. 7519–7541, 2017.

[24] W. Gong, Z. Cai, and C. X. Ling, "DE/BBO: a hybrid differential evolution with biogeography-based optimization for global numerical optimization," *Soft Computing*, vol. 15, no. 4, pp. 645–665, 2010.

[25] JJ Liang, BY Qu, PN Suganthan, and Q Chen, *Problem Definitions and Evaluation Criteria for the CEC 2015 Competition on Learning-based Real-Parameter Single Objective Optimization*, Zhengzhou University (China) and Nanyang Technological University, Singapore), 2014.