Gradient eigendecomposition invariance biogeography-based optimization for mobile robot path planning

Xiaodong Na · Jiaqian Wang · Min Han · Decai Li

Abstract

The path planning for mobile robots has attracted extensive attention, and evolutionary algorithms have been applied to this problem increasingly. In this paper, we propose a novel gradient eigendecomposition invariance biogeography-based optimization (GEI-BBO) for mobile robot path planning, which has the merits of high rotation invariance and excellent search performance. In GEI-BBO, we design an eigendecomposition mechanism for migration operation, which can reduce the dependency of biogeography-based optimization (BBO) on the coordinate system, improve the rotation invariance and share the information between eigensolutions more effectively. Meanwhile, to find the local optimal solution better, gradient descent is added, and the system search strategy can reduce the occurrence of local trapping phenomenon. In addition, combining the GEI-BBO with cubic spline interpolation will solve the problem of mobile robot path planning through a defined coding method and fitness function. A series of experiments are implemented on benchmark functions, whose results indicated that the optimization performance of GEI-BBO is superior to other algorithms. And the successful application of GEI-BBO for path planning in different environments confirms its effectiveness and practicability.

Keywords
Mobile robot path planning · Biogeography-based optimization · Eigendecomposition · Gradient decent strategy · System search strategy

1 Introduction

In recent decades, mobile robot path planning (MRPP) has become an indispensable aspect of artificial intelligence in robotics and has been widely studied and discussed (Manikas et al. 2007). The goal of path planning is to search out an optimal or nearly optimal collision free path from the initial state to the target state according to a certain performance index (such as time and distance) (Rasekhipour et al. 2017). According to different environmental information, path planning can be divided into global path planning and local path planning (Sucan et al. 2012). Global path planning is carried out in the known environment. By modeling the current environmental information, mobile robots can use the existing mature path planning algorithm or improved algorithm to plan an optimal path from the start to the end in the established environmental model (Borenstein and Koren 1991). In the local path planning problem, the environmental information is usually unknown, focusing on considering the current local environmental information of the mobile robot to make the robot have good obstacle avoidance ability. The working environment of the mobile robot is detected by sensors to
obtain information such as the location and several properties of obstacles (Zeng et al. 2016). However, without complete environmental information, the results of local path planning may not be best or even incorrect. Therefore, global path planning, which can be regarded as an optimization problem, has attracted more attention and became a hot topic in research of path planning (Alam and Rafique 2015).

Researchers have developed many methods for global MRPP. Traditional path planning methods include grid method (Borenstein and Koren 1991), artificial potential field method (Chen et al. 2016), visual graph method (Sucan et al. 2012), neural network method (Kim and Lee 2015), etc. However, these methods partly have some problems such as complex calculation, low efficiency and easy to fall into local optimization, which usually leads to certain inconveniences in solving the problem of global MRPP. Meanwhile, evolutionary algorithms for MRPP show a growing number of advantages and attract increasing attention (Lim et al. 2008).

Using the evolutionary algorithm for global MRPP, which can put the problem of path planning down to the problem of finding the optimal path with the minimum cost (Kennedy and Eberhart 1995). In addition, many evolutionary algorithms, such as genetic algorithm (GA) (Bakdi et al. 2017), particle swarm optimization algorithm (PSO) (Tharwat et al. 2019) and ant colony algorithm (ACO) (Kumar et al. 2018), have strong computing capability and robustness, which can achieve optimization results well. At present, many evolutionary algorithms have been applied to the problem of MRPP (Castillo et al. 2006). For example, considering the slow convergence speed of ACO, Liu et al. (2017) proposed an improved ACO to improve its convergence speed and applied it to the MRPP in the grid environment. Marco et al. (2015) combined the artificial bee colony algorithm with the evolutionary planning algorithm to solve the problem of MRPP, through a set of local processes to refine the feasible path. The comparison results of the method on a set of benchmark problems and the experimental results on a real mobile robot show that the method has good performance. Hong et al. (2013) proposed a coevolutionary improved GA for global MRPP, which puts forward an effective fitness function and modifies the genetic operator of traditional GA. For the problem of global smooth MRPP, Song et al. (2017) advanced a new multimodal delayed PSO (MDPSO). The test results based on benchmark function show that the performance of MDPSO is better than other five famous PSO algorithms. Finally, the application of MDPSO in the global smooth MRPP further proves that it has better performance than the global smooth path generated by GA in previous studies. These evolutionary algorithms have been used to solve the problem of MRPP and have achieved remarkable results, but in general, these methods still have some shortcomings in computational complexity, local optimization and adaptability.

Biogeography-based optimization (BBO) (Simon 2008) is an evolutionary algorithm based on the concept of biogeography. Based on a mathematical model, the algorithm describes the migration of species between habitats, that is, from unsuitable habitats to suitable habitats (Simon et al. 2010). BBO is an efficient bionic search algorithm, which has the advantages of relatively simple principle, less adjustment parameters, easier implementation of the algorithm and high operation efficiency. Many researchers have improved BBO and applied it to many fields (Mirjalili et al. 2014), including MRPP. For example, Zhu et al. (2014) proposed a method based on chaotic predator–prey BBO (CPPBBO) to solve the path planning problem of unmanned aerial vehicle, and the simulation results show that CPPBBO is more effective than other algorithms. Mo et al. (2015) advanced a biogeography PSO algorithm to plan the path of mobile robot by combining the BBO and PSO, and then using the BPSO algorithm, the optimal path based on the approved voronoi boundary network is found out. Yang et al. (2017) put forward an improved optimization algorithm based on biogeography to solve the problem of global MRPP in the static environment. It can be seen from these articles that BBO has been effectively applied to MRPP. However, due to the strong dependence of BBO on the coordinate system, the performance of the algorithm is inadequate when solving high-dimensional problems, and the ability of BBO to mine the global optimal solution is also slightly insufficient.

In order to solve these problems, we proposed a novel gradient eigendecomposition invariance biogeography-based optimization algorithm (GEI-BBO). The innovation of this paper is as follows:

1. An eigendecomposition-based migration strategy is proposed, which has the advantages of reducing the dependence on coordinates, improving the rotation invariance of BBO and sharing the information between eigensolutions more effectively.
2. On the basis of the eigendecomposition-based migration strategy, the gradient descent strategy is added to search the neighborhood of the best individual, which can mine the global optimal solution, so as to improve the local search ability of BBO.
3. The system search strategy is put forward to ensure that the algorithm with gradient descent strategy finds the global optimal solution, which can carry out a range search for each dimension, so as to cover the whole search range well and avoid falling into local optimum.
4. GEI-BBO is combined with the cubic spline interpolation method to solve the problem of MRPP. This paper defines the coding method based on the path node and constructs fitness function which aims at avoiding the obstacle and finding the shortest path.
The rest of the paper is organized as follows: Section 2 introduces the basic BBO. Section 3 introduces the gradient eigendecomposition invariance biogeography-based optimization. Section 4 shows the method for path planning, and Section 5 shows the simulation results and analysis.

2 Biogeography-based optimization

BBO is a new population-based optimization algorithm to solve global optimization problems, based on the concept of biogeography. In biogeography, each habitat is regarded as an individual, and the index to measure its quality of life is called habitat suitability index (HSI) (Mo and Xu 2015). Good habitats for species have high HSI, and bad habitats have low HSI. A habitat has many characteristics, such as area, temperature, rainfall and so on. These characteristics will affect the habitat, so they are referred to as the suitability index variables (SIVs). In BBO, a candidate solution to the problem is considered as a habitat. Each candidate solution is associated with a fitness value that is similar to the HSI of the habitat. High HSI habitats represent better solutions, and low HSI habitats represent worse solutions. Immigration rates and emigration rates for each candidate solution are used to share features probabilistically between habitats.

In BBO, the rate of immigration $\lambda$ and the rate of emigration $\mu$ determine the dynamic movement between habitats, depending on the number of species in the habitat (Simon 2011). The equation for the rate of immigration $\lambda_k$ and the rate of emigration $\mu_k$ of $k$ species can be written as follows (Fig. 1):

$$\lambda_k = I \left( 1 - \frac{k}{n} \right)$$  \hspace{1cm} (1)

$$\mu_k = E \left( \frac{k}{n} \right)$$  \hspace{1cm} (2)

where $I$ and $E$ are the maximum immigration rate and the maximum emigration rate, respectively. $n = s_{max}$ is the maximum number of species in the habitat. If $I = E$, then $\lambda_k + \mu_k = I$.

BBO is different from other evolutionary algorithms because of its migration strategy. The migration strategy is based on the migration model, that is, the immigration rate and the emigration rate equation. The migration model generally adopts the linear model. When a habitat $X_i$ needs to be relocated according to the migration rate, some methods, such as roulette, will be used to select a source habitat $X_j$ probabilistically according to the migration rate, and then a SIV will be selected randomly in the source habitat to replace directly or modify $X_i$. BBO can improve the solution through the migration operation.

The mutation operation of BBO is to randomly mutate the habitat according to the mutation rate $m(s)$, that is, changing the SIVs of the habitat. $m(s)$ is determined by the following equation:

$$m(s) = m_{\max} \left( 1 - \frac{P_s}{P_{\max}} \right)$$  \hspace{1cm} (3)

where $m(s)$ is the mutation rate of habitat, $m_{\max}$ is the maximum mutation rate, and $P_{\max}$ is the maximum probability. BBO can increase the diversity of species through mutation operation (Du et al. 2009).

3 Gradient eigendecomposition invariance biogeography-based optimization

In this section, we introduce a method to solve the problem of two-dimensional static global MRPP. We introduce three innovative points of the proposed algorithm, namely the eigendecomposition-based migration, gradient descent and system search strategy. Finally, the algorithm is described.

3.1 Eigendecomposition-based migration

In order to improve the rotation invariance of BBO algorithm, the eigendecomposition-based migration is proposed in this paper. The core of the eigendecomposition-based migration is to rotate the original coordinate system into the eigenvector-based coordinate system, in which habitats can share their information more effectively. The proposed method illustrates that the migration of BBO can be carried out more effectively in the eigenvector-based coordinate system. The eigendecomposition-based migration is to carry out the eigendecomposition of population and migrate the decomposed population, which will make BBO run more effectively (Fig. 2).
Firstly, suppose a population $H$

\[
H = (H^G_i, \ldots, H^G_1, \ldots, H^G_n)^T
\]

\[
H^G_i = (H^G_{(i,1)}, \ldots, H^G_{(i,j)}, \ldots, H^G_{(i,D)})
\]

where $i = 1, 2, \ldots, n, j = 1, 2, \ldots, D, n$ is the population size, $D$ is the number of independent variables, $G$ is the number of iterations, and $H^G_i$ is the $i$th population in the $G$th iteration.

$H$ is not a square matrix of order $n$; it cannot be eigendecomposed directly, so we introduce the concept of covariance matrix.

Make:

\[
H^G_{n \times D} = \begin{bmatrix}
H^G_{(1,1)} & H^G_{(1,2)} & \cdots & H^G_{(1,D)} \\
H^G_{(2,1)} & H^G_{(2,2)} & \cdots & H^G_{(2,D)} \\
\vdots & \vdots & \ddots & \vdots \\
H^G_{(n,1)} & H^G_{(n,1)} & \cdots & H^G_{(n,D)}
\end{bmatrix} = \begin{bmatrix} c_1 & c_2 & \cdots & c_D \end{bmatrix}
\]

Then, the covariance matrix is

\[
\text{cov}(H) = \frac{1}{n-1} \begin{bmatrix}
\text{cov}(c_1, c_1) & \text{cov}(c_1, c_2) & \cdots & \text{cov}(c_1, c_D) \\
\text{cov}(c_2, c_1) & \text{cov}(c_2, c_2) & \cdots & \text{cov}(c_2, c_D) \\
\vdots & \vdots & \ddots & \vdots \\
\text{cov}(c_D, c_1) & \text{cov}(c_D, c_2) & \cdots & \text{cov}(c_D, c_D)
\end{bmatrix}
\]

where $\text{cov}(c_i, c_j)$ is the covariance between the $i$th independent variable and the $j$th independent variable in $G$th generation, which is defined as follows:

\[
\text{cov}(c_i, c_j) = E[(c_i - \xi_i)(c_j - \xi_j)]
\]

where $\xi_i$ and $\xi_j$ represent the mean values of the $i$th and $j$th independent variables, respectively.

Thus, it can be seen that covariance matrix $\text{cov}(H)$ is a symmetric matrix of order $n$, and the following standard form can be obtained by eigendecomposition:

\[
\text{cov}(H) = Q_H \Lambda_H Q_H^T
\]

where $Q_H$ is a $D \times D$ eigenmatrix composed of feature vector of $\text{cov}(H)$ and $\Lambda_H$ is a diagonal matrix composed of all the eigenvalues of $\text{cov}(H)$. After eigendecomposition, the eigenvector is obtained. We also need to rotate the matrix $H$ with the eigenvectors. That is:

\[
\begin{bmatrix}
\text{eig} H^G_1, \text{eig} H^G_2, \ldots, \text{eig} H^G_n
\end{bmatrix}
\]

where $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, D$. Migration of the eig$H^G_i$ obtained by the rotation operation resulting in the generation of eig$H^G_{i+1}$. Because of the eigendecomposition and rotation operation, the migration can run more efficiently. After the migration operation, $H$ should be rotated back and used as the population of the new generation. The following operation is enough.

\[
H^G_{i+1} = \text{eig} H^G_{i+1} \times Q_H
\]

In summary, the steps of the migration algorithm based on eigendecomposition are as follows:

\begin{algorithm}
\textbf{Algorithm 1} Eigendecomposition-based migration
\begin{enumerate}
\item \textbf{Require:} $H^G_i$
\item \textbf{Ensure:} $H^G_{i+1}$
\item \textbf{for} $i = 1$ to $n$ \textbf{do}
\item \hspace{1em} Compute the matrix $\text{cov}(H)$
\item \hspace{1em} Apply eigendecomposition according to Equation 8
\item \hspace{1em} Rotate according to Equation ??
\item \textbf{for} $k = 1$ to $D$ \textbf{do}
\item \hspace{1em} \textbf{if} rand $< \lambda_i$ \textbf{then}
\item \hspace{2em} select $\text{eig} H^G_j$ with probability $\mu_j$
\item \hspace{2em} $\text{eig} H^G_{(i,k)} \leftarrow \text{eig} H^G_{(i,k)} + \alpha \cdot \text{eig} H^G_j$
\item \hspace{2em} $\text{eig} H^G_{(i,k)} \leftarrow \text{eig} H^G_{(i,k)}$
\item \hspace{1em} \textbf{end if}
\item \textbf{end for}
\item \textbf{end for}
\item \textbf{Get} $H^G_{i+1}$ according to Equation 10
\item \textbf{end for}
\end{enumerate}
\end{algorithm}
3.2 Gradient decent strategy

BBO algorithm is effective in searching the global optimal region, but it is difficult in mining the global optimal region. The local search performance of BBO can be improved by adding gradient descent into BBO to search the best individual domain.

The following conditions are required to activate the local search of gradient descent:

1. \( N_G \) is a positive integer.
2. \( G_{flag} \) is equal to \( G_A \), where \( G_A \) is a predefined value.

If both of these conditions are met, then we apply gradient descent strategy to \( N_G \) optimal individuals. \( G_{flag} \) starts at zero and increases by 1 when the following conditions are met.

\[
R_I = \frac{f_{\min}(it) - f_{\min}(it + 1)}{f_{\min}(it)} \leq \varepsilon_1
\]  

where \( R_I \) is the improvement ratio, which is the relative improvement in the minimum generation value from the \( i(t) \) iteration to the \( i(t + 1) \) iteration. \( \varepsilon_1 \) is a predefined threshold. Moreover, by adding \( G_A \) through \( G_{INC} \), the incremental \( G_A \) delays the next call to gradient descent local search. The reason why the gradient is called infrequently is that when the algorithm falls into a local optimal state, multiple local searches may not be helpful, and the function evaluation times will be wasted.

In short, gradient descent is activated when \( R_I \) is continuously smaller than \( \varepsilon_1 \). We use the \( f_{\min} \) function in MATLAB to achieve the gradient descent of the best individual. The gradient descent algorithm is shown as follows:

Algorithm 2 Gradient descent strategy

Require: \( H_i^{G}, i = 1: N_G \)
Ensure: \( H_{i+1}^{G}, i = 1: N_G \)
1: Compute improvement ratio \( R_I < \frac{(f_{\min}(it) - f_{\min}(it + 1))}{f_{\min}(it)} \)
2: if \( R_I < \varepsilon_1 \) then
3: \( G_{flag} = G_{flag} + 1 \)
4: else
5: \( G_{flag} = 0 \)
6: end if
7: if \( N_G > 0 \) and \( G_{flag} = G_A \) then
8: Apply gradient descent to the \( N_G \) best individuals in the population
9: \( G_{flag} = 0 \)
10: \( G_A = G_{flag} + G_I \)
11: end if

3.3 System search strategy

In order to ensure that the algorithm with gradient descent strategy can find the global optimal solution, we add the system search strategy, which can cover the whole search range well and avoid falling into the local optimal.

The conditions for activating the system search strategy are similar to those for gradient decent in 3.2, and the improved ratio \( R_I \) is also used. When the following formula is satisfied, the system search strategy will be activated.

\[
R_I = \frac{f_{\min}(it) - f_{\min}(it + 1)}{f_{\min}(it)} \leq \varepsilon
\]  

where \( \varepsilon \) is the computer precision, that is, when there is no improvement in the minimum cost value from the \( i(t) \) iteration to the \( i(t + 1) \) iteration, a global search will be conducted for \( N_S \) optimal individuals to execute the system search strategy. The steps of the system search strategy are given as Algorithm 3.

Algorithm 3 System search strategy

Require: \( H_i^{G}, i = 1: N_S \)
Ensure: \( H_{i+1}^{G}, i = 1: N_S \)
1: if \( R_I < \varepsilon \) then
2: \( \Delta = a_0(UB - LB) \)
3: for \( i = 1 \) to \( N_S \) do
4: for \( k = 1 \) to \( D \) do
5: \( temp_{i,k} = pop_{i,k} \)
6: while \( \Delta \geq LB \) do
7: \( temp_{i,k} = temp_{i,k} - \Delta \)
8: if \( f(temp_{i,k}) < f(pop_{i,k}) \) then
9: \( pop_{i,k} = temp_{i,k} \)
10: end if
11: end while
12: \( temp_{i,k} = pop_{i,k} \)
13: while \( \Delta \leq UB \) do
14: \( temp_{i,k} = temp_{i,k} + \Delta \)
15: if \( f(temp_{i,k}) < f(pop_{i,k}) \) then
16: \( pop_{i,k} = temp_{i,k} \)
17: end if
18: end while
19: end for
20: end for
21: end if

As shown in the table, we increment or decrement the independent variables in accordance with \( \Delta \) in the search space, where \( a_0 \) is a specific score that can be evaluated as \( a_0 = 0.1 \). The system search strategy decreases the value of the given dimension \( pop_{i,k} \) by an increment \( \Delta \), equal to 10% of the size of the search space, one increment at a time, until the value reaches the lower bound of the search space. Then, the system search strategy increases the value of the given dimension, one increment at a time, until it reaches the upper limit of the search space. By performing this process...
for each dimension, the system search strategy realizes the global search and replaces the original individual with the best one found by it to avoid falling into the local optimization.

3.4 Description of the GEI-BBO algorithm

The GEI-BBO algorithm proposed in this paper includes the contents mentioned in Sects. 3.1, 3.2 and 3.3. Before the number of iterations reaches the maximum number of iterations \( m \), \( n \) habitats are selected to migrate according to the proportional parameter \( P \) in each iteration. If the random number is less than \( P \), the migration is based on the eigendecomposition, and if the random number is greater than \( P \), the migration is based on the standard BBO. After migration, the mutation was carried out. Then, the gradient descent strategy is implemented on the \( N_G \) best individuals and the system search strategy is implemented on the \( N_s \) best individuals. \( N_G \) and \( N_s \) are preset parameters. The specific algorithm flow is as follows:

4 Method for path planning

Combining the improved algorithm with cubic spline interpolation method, the coding method based on the path node is defined, and the method and fitness function aiming at solving the obstacle avoidance and shortest path of mobile robot are constructed to solve the problem of MRPP. Figure 3 shows a conceptual map of habitat migration in MRPP using GEI-BBO to understand how the proposed approach works.

In the figure, habitat 1 has the highest HSI, representing the most suitable habitat, followed by habitat 2, habitat 3 and habitat 4. The higher the HSI, the more suitable for species growth, the less need to be changed, so the lower the immigration rate, the higher the emigration rate. It can be seen that habitat 1 has the highest emigration rate, while habitat 4 has the highest emigration rate. Residents represent the path nodes. So the fourth habitat accepts many residents (path
nodes) from other habitats, as shown in different colors. Purple nodes and links also describe mutations that occur in all habitats, regardless of their HSI values. This example shows how path planning evolved using the proposed approach. The specific coding method and fitness function are shown below.

4.1 Cubic spline interpolation

Cubic spline interpolation is a kind of piecewise interpolation method to form a smooth curve through a train of interpolation point intervals based on cubic polynomials. The curve of moving path of mobile robot fitted by cubic spline interpolation method is smoother than the curve fitted by straight line and circular arc. In this paper, the cubic spline interpolation method is integrated into the improved GEI-BBO algorithm to solve the problem of static global MRPP.

On the interval \([a, b]\), taking \(n + 1\) nodes \(a = x_0 < x_1 < \cdots < x_n = b\), if \(s(x)\) satisfies the following conditions:

1. \(s(x) \in C^2[a, b]\).
2. On each small interval \([x_i, x_{i+1}]\), \(s(x)\) is a cubic polynomial.
3. On node \(x_i\), given the function value \(f_i = f(x_i), i = 0, 1, \ldots, n\) and \(s(x_i) = f_i, i = 0, 1, \ldots, n\). Then, \(s(x)\) is a cubic spline interpolation function.

The cubic spline interpolation function is a piecewise cubic polynomial, on each small interval \([x_i, x_{i+1}]\), which can be written as: \(s(x) = a_i x^3 + b_i x^2 + c_i x + d_i, i = 0, 1, \ldots, n - 1\), where \(a_i, b_i, c_i, d_i\) is the undetermined coefficient, so \(s(x)\) has 4 undetermined coefficients. To solve for \(s\) of \(x\), we need \(4n\) conditions. From \(s(x_i) = f_i, i = 0, 1, \ldots, n, n + 1\) interpolation conditions can be obtained. And from \(s(x) \in C^2[a, b]\), \(s(x)\) is second differentiable on the interval \([a, b]\), so the first order is differentiable and \(s(x)\) is continuous. Therefore, the following conditions can be obtained:

\[
\begin{align*}
 s"(x_i) &= s"_+(x_i), i = 0, 1, \ldots, n - 1 \\
 s'(x_i) &= s'_+(x_i), i = 0, 1, \ldots, n - 1 \\
 s(x_i) &= s_+(x_i), i = 0, 1, \ldots, n - 1 \\
\end{align*}
\]

So far, there are \(4n\) conditions. In the actual calculation, two boundary conditions need to be introduced to calculate \(s(x)\). Commonly used boundary conditions are:

1. The value of the first derivative at the two endpoints is given.
2. The value of the second derivative at the two endpoints is given.
3. \(s(x)\) is a function of period \(b - a\).

4.2 Encoding

It can be seen from the above that cubic spline interpolation is a piecewise interpolation method, and the junction between segments is called the path node. The splines between segments are different, and the whole spline curve is continuous in the first order and is continuous in the second order at the path node. Therefore, the directions between segmented splines may be different, and the path node represents the maximum turning times of the entire path. According to this, this paper takes all nodes on a path as the code of a habitat individual, that is, a habitat individual represents all nodes on the corresponding path. The \(x\)-coordinate set of all \(m\) path nodes on a path constitutes the \(m\)-dimensional \(x\)-coordinate of habitat individuals, and correspondingly, the \(y\)-coordinate set of all \(m\) path nodes on a path constitutes the \(m\)-dimensional \(y\)-coordinate of habitat individuals.

Suppose we know the coordinates of \(m\) path node \((x_{m1}, y_{m1}), (x_{m2}, y_{m2}), \ldots, (x_{mm}, y_{mm})\), starting point coordinates \((x_s, y_s)\) and terminal coordinates \((x_t, y_t)\), on the interval \((x_1, x_{m1}, x_{m2}, \ldots, x_{mm}, x_t)\) and \((y_1, y_{m1}, y_{m2}, \ldots, y_{mm}, y_t)\), the abscissa and ordinate of \(n\) interpolation points are obtained by cubic spline interpolation. In this way, we get \(n\) interpolation points, and the line between the path node and the interpolation point and the starting point and the ending point is the path of the mobile robot that we want.

4.3 Fitness function

There are two conditions for global MRPP:

1. Do not collide with obstacles;
2. The path length should be as short as possible. In this paper, the evaluation criterion of fitness function is the shortest path length satisfying the above conditions. In this paper, the constructed fitness function is

\[
\begin{align*}
 f &= L \cdot (1 + \beta \cdot v) \\
\end{align*}
\]

where \(\beta\) is a very large number, and it is used to impose a very large cost value to exclude paths that have collisions, and it can take on a value of 1000. \(L\) is the distance from the starting point to the terminal point; the calculation formula is

\[
\begin{align*}
 L &= \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \\
\end{align*}
\]

where \(v\) is a flag variable. \(v = 0\) means the path is a non-collision path; otherwise, there is a collision path. The initial value of \(v\) is set to 0. In the rectangular obstacle environment, the following equation is used to determine
whether the obstacle is encountered:

\[
d = (x_{obs_k} - \text{space}) < xx_j \land xx_j < (x_{obs_k} + l_{obs_k} + \text{space}) \land (y_{obs_k} - \text{space}) < yy_j \land yy_j < (y_{obs_k} + w_{obs_k} + \text{space})
\] (18)

where \( j \) is the number of nodes, including path nodes and interpolation points. \( xx_j \) and \( yy_j \) represent the abscissa and ordinate of the node. \((x_{obs_k}, y_{obs_k})\) is the starting point of the \( k \)th rectangle obstacle, that is the bottom left corner of the rectangle. \( l_{obs_k}, w_{obs_k} \) is the length and width of the rectangular obstacle. \( \text{space} \) is a small positive number to avoid hitting the edge of the rectangle.

At each node, judge whether it collides with the obstacle according to formula 18. When it encounters the obstacle, \( d \) will generate a nonzero change value. When \( d \) changes, \( v \) will be added with one. If the path does not pass through any obstacle, then \( v = 0 \); otherwise, \( v \) is equal to a number that is not zero.

4.4 Path planning process

We combine GEI-BBO with cubic spline interpolation to design the path planning steps; the flowchart of path planning is shown in Fig. 6.

- **Step 1** Determine the number of path nodes \( m \) and initialize the algorithm.
- **Step 2** The coordinates of \( n \) interpolation points are obtained by cubic spline interpolation.
- **Step 3** The path length \( L \) and the flag variable \( v \) are calculated, respectively, to obtain the value of fitness function.
- **Step 4** Using GEI-BBO to update the ordered habitat individuals.
- **Step 5** Determine whether the maximum number of iterations is reached. If yes, output the optimal result directly. If not, return to the step 2 for circulation (Fig. 4).

5 Simulation results and analysis

In order to prove the effectiveness and practicability of the proposed method, we first use 23 benchmark test functions to compare and test the GEI-BBO and then use it in the simulation experiment of path planning and conduct in-depth analysis and summary of the experimental results.
### Table 1  Benchmark test functions

| ID  | Function name                                | Separability | Dimension | Search space         | Optima  |
|-----|----------------------------------------------|--------------|-----------|----------------------|---------|
| F1  | Sphere model                                | Separable    | 30        | $[-100, 100]^D$      | 0       |
| F2  | Schwefel’s problem 2.22                     | Non-separable| 30        | $[-10, 10]^D$        | 0       |
| F3  | Schwefel’s problem 1.2                      | Non-separable| 30        | $[-100, 100]^D$      | 0       |
| F4  | Schwefel’s problem 2.21                     | Non-separable| 30        | $[-100, 100]^D$      | 0       |
| F5  | Generalized Rosenbrocks functions            | Non-separable| 30        | $[30, 30]^D$         | 0       |
| F6  | Step function                               | Separable    | 30        | $[-100, 100]^D$      | 0       |
| F7  | Quartic function                             | Separable    | 30        | $[-1.28, 1.28]^D$    | 0       |
| F8  | Generalized Schwefel’s problem 2.26         | Separable    | 30        | $[-500, 500]^D$      | $-12569.5$|
| F9  | Generalized Rastrigin’s function             | Separable    | 30        | $[-5.12, 5.12]^D$    | 0       |
| F10 | Ackley’s function                            | Separable    | 30        | $[-32, 32]^D$        | 0       |
| F11 | Generalized Griewank function               | Separable    | 30        | $[-600, 600]^D$      | 0       |
| F12 | Generalized penalized function 1            | Non-separable| 30        | $[-50, 50]^D$        | 0       |
| F13 | Generalized penalized function 2            | Non-separable| 30        | $[-50, 50]^D$        | 0       |
| F14 | Shekel’s Foxholes function                  | Non-separable| 2         | $[-65.536, 65.536]^D$| 0.9980  |
| F15 | Kowalik’s function                           | Non-separable| 4         | $[-5, 5]^D$          | 0.00031 |
| F16 | Six-hump camel-back function                | Non-separable| 2         | $[-5, 5]^D$          | $-1.03162$|
| F17 | Branin function                              | Non-separable| 2         | $[-5,10] \times [0,15]$| 0.39788 |
| F18 | Goldstein–Price function                    | Non-separable| 2         | $[-2, 2]^D$          | 2.99999 |
| F19 | Hartmann’s function 1                       | Non-separable| 3         | $[0, 1]^D$           | $-3.86278$|
| F20 | Hartmann’s function 2                       | Non-separable| 6         | $[0, 1]^D$           | $-3.32199$|
| F21 | Shekel’s function 1                         | Non-separable| 4         | $[0, 10]^D$          | $-10.1531$|
| F22 | Shekel’s function 2                         | Non-separable| 4         | $[0, 10]^D$          | $-10.4029$|
| F23 | Shekel’s function 3                         | Non-separable| 4         | $[0, 10]^D$          | $-10.5364$|

![Fig. 5 Three-dimensional graph of the partial test function](image-url)
| ID | Best | Mean  |
|----|------|-------|
| F1 | 5.14E+01 | 7.11E−15 |
| F2 | 3.03E−03 | 1.90E−12 |
| F3 | 1.39E+01 | 5.33E+00 |
| F4 | 3.60E+01 | 6.01E−10 |
| F5 | 8.61E+01 | 1.75E+00 |
| F6 | 3.63E+02 | 2.38E+00 |
| F7 | 1.13E+01 | 1.26E+03 |
| F8 | 6.75E+03 | 1.81E+01 |
| F9 | 2.08E+01 | 3.85E+00 |
| F10 | 4.08E+00 | 2.00E−08 |
| F11 | 6.42E+00 | 1.34E−07 |
| F12 | 4.86E+00 | 2.07E−02 |
| F13 | 2.08E+01 | 6.63E−12 |
| F14 | 3.57E+01 | 4.16E−03 |
| F15 | 5.03E+00 | 3.51E+00 |
| F16 | 3.00E+00 | 3.80E+00 |
| F17 | 1.33E+00 | 1.00E+00 |
| F18 | 3.00E+00 | 3.00E+00 |
| F19 | 3.00E+00 | 3.86E+00 |
| F20 | 3.00E+00 | 3.86E+00 |
| F21 | 3.00E+00 | 3.86E+00 |
| F22 | 3.00E+00 | 3.86E+00 |
| F23 | 3.00E+00 | 3.86E+00 |
Table 3 Path planning results obtained by different algorithms (part 2)

| ID | Fitness | SSA         | BBO         | IWO/BBO     | LBBO        | GEI-BBO     |
|----|---------|-------------|-------------|-------------|-------------|-------------|
| F1 | Best    | 5.97E−08    | 3.57E−01    | 4.42E−17    | 0.00E+00    | 8.95E−39    |
|    | Mean    | 2.99E−07    | 5.88E−01    | 6.61E−17    | 0.00E+00    | 2.74E−24    |
| F2 | Best    | 7.22E−06    | 1.92E−01    | 7.10E−09    | 6.86E−04    | 1.13E−33    |
|    | Mean    | 3.14E−03    | 2.35E−01    | 2.02E−08    | 4.56E−03    | 5.25E−17    |
| F3 | Best    | 1.92E−08    | 6.20E+01    | 1.23E−14    | 6.65E−12    | 6.25E−21    |
|    | Mean    | 5.60E−07    | 8.67E+01    | 1.71E−14    | 2.30E−09    | 3.93E−19    |
| F4 | Best    | 1.49E−05    | 6.53E−01    | 2.46E−11    | 3.40E−08    | 5.64E−17    |
|    | Mean    | 2.95E−05    | 7.87E−01    | 4.54E−08    | 1.31E−07    | 9.50E−17    |
| F5 | Best    | 6.41E+00    | 3.56E+01    | 8.49E−04    | 8.50E−03    | 0.00E+00    |
|    | Mean    | 1.82E+02    | 1.17E+02    | 5.67E+01    | 2.34E+01    | 0.00E+00    |
| F6 | Best    | 5.66E−10    | 0.00E+00    | 0.00E+00    | 2.84E+02    | 0.00E+00    |
|    | Mean    | 8.97E−10    | 0.00E+00    | 0.00E+00    | 3.74E+02    | 0.00E+00    |
| F7 | Best    | 5.46E−03    | 1.79E−03    | 3.11E−05    | 2.88E−02    | 1.85E−05    |
|    | Mean    | 1.29E−02    | 4.16E−03    | 3.19E−05    | 3.69E−02    | 6.75E−05    |
| F8 | Best    | −3.14E+03   | −9.80E+03   | −4.38E+03   | −6.74E+03   | −9.79E+03   |
|    | Mean    | −2.79E+03   | −9.04E+03   | −3.59E+03   | −4.79E+03   | −8.72E+03   |
| F9 | Best    | 3.98E+00    | 2.23E+01    | 1.78E−14    | 6.87E−11    | 0.00E+00    |
|    | Mean    | 1.47E+01    | 3.40E+01    | 3.28E−05    | 1.67E−03    | 2.23E−08    |
| F10| Best    | 1.00E−05    | 1.98E−01    | 2.27E−03    | 2.32E+00    | 2.11E+00    |
|    | Mean    | 8.78E−01    | 2.39E−01    | 3.68E−03    | 2.45E+00    | 2.12E+00    |
| F11| Best    | 5.90E−02    | 3.71E−01    | 2.39E−13    | 1.16E−05    | 1.11E−16    |
|    | Mean    | 2.01E−01    | 5.78E−01    | 1.06E−02    | 9.53E−03    | 5.88E−16    |
| F12| Best    | 8.74E−12    | 8.64E−04    | 7.35E−17    | 1.03E−05    | 2.73E−32    |
|    | Mean    | 8.48E−01    | 1.37E−03    | 5.47E−15    | 7.19E−04    | 3.86E−30    |
| F13| Best    | 5.50E−11    | 1.02E−02    | 1.87E−18    | 7.90E−10    | 1.10E−02    |
|    | Mean    | 1.10E−03    | 1.96E−02    | 3.61E−17    | 6.08E−04    | 9.85E−02    |
| F14| Best    | 9.98E−01    | 9.98E−01    | 9.98E−01    | 9.98E−01    | 9.98E−01    |
|    | Mean    | 1.10E+00    | 1.20E+00    | 9.98E−01    | 9.98E−01    | 2.08E+00    |
| F15| Best    | 6.04E−04    | 3.61E−04    | 3.08E−04    | 3.08E−04    | 3.08E−04    |
|    | Mean    | 9.06E−04    | 5.38E−04    | 3.08E−04    | 3.08E−04    | 4.11E−04    |
| F16| Best    | −1.03E+00   | −1.03E+00   | −1.03E+00   | −1.02E+00   | −1.03E+00   |
|    | Mean    | −1.03E+00   | −1.03E+00   | −9.30E−01   | −1.01E+00   | −1.03E+00   |
| F17| Best    | 3.98E−01    | 3.98E−01    | 3.98E−01    | 3.98E−01    | 3.98E−01    |
|    | Mean    | 3.98E−01    | 3.98E−01    | 3.98E−01    | 3.98E−01    | 3.98E−01    |
| F18| Best    | 3.00E+00    | 3.00E+00    | 3.00E+00    | 3.00E+00    | 3.00E+00    |
|    | Mean    | 3.00E+00    | 3.00E+00    | 3.00E+00    | 3.00E+00    | 3.00E+00    |
| F19| Best    | −3.86E+00   | −3.86E+00   | −3.86E+00   | −3.86E+00   | −3.86E+00   |
|    | Mean    | −3.86E+00   | −3.86E+00   | −3.86E+00   | −3.86E+00   | −3.86E+00   |
| F20| Best    | −3.32E+00   | −3.32E+00   | −2.37E+00   | −3.32E+00   | −3.32E+00   |
|    | Mean    | −3.32E+00   | −3.32E+00   | −2.32E+00   | −3.31E+00   | −3.32E+00   |
| F21| Best    | −1.02E+01   | −1.02E+05   | −2.05E+00   | −1.97E+00   | −1.02E+01   |
|    | Mean    | −8.65E+00   | −1.02E+01   | −1.30E+00   | −1.30E+00   | −1.02E+01   |
| F22| Best    | −1.04E+01   | −1.04E+01   | −3.92E+00   | −5.10E+00   | −1.04E+01   |
|    | Mean    | −1.04E+01   | −7.73E+00   | −1.96E+00   | −5.04E+00   | −1.04E+01   |
| F23| Best    | −1.05E+01   | −1.05E+01   | −2.30E+00   | −5.34E+00   | −1.05E+01   |
|    | Mean    | −6.93E+00   | −9.33E+00   | −1.80E+00   | −5.25E+00   | −1.05E+01   |
Algorithm (Mirjalili and Lewis 2016), sine cosine algorithm (Mirjalili 2016), salp swarm algorithm (Mirjalili et al. 2017), hybrid invasive weed/biogeography-based optimization (IWO/BBO) (Khademi et al. 2017) and linearized biogeography-based optimization (LBBO) (Simon et al. 2014), and Tables 2 and 3 show the test results of comparative algorithms.

In Tables 2 and 3, MEAN is the average of ten test results, BEST represents the best result of 10, and the bold part is the best one among 11 algorithms. Our proposed GEI-BBO obtains 16 optimal BEST and 14 optimal MEAN solutions in the optimization process of 23 functions, and its performance is better than the other 10 comparison algorithms. The experimental results verify the effectiveness of the optimization process of GEI-BBO on the benchmark function.

### 5.2 Simulation on path planning

To verify the effectiveness and feasibility of the proposed algorithm in solving the problem of MRPP, we compare the proposed GEI-BBO algorithm with GA (Bakdi et al. 2017), PSO (Kennedy and Eberhart 1995), ACO (Mavrovouniotis and Yang 2011), BBO (Simon 2011) and IWO/BBO (Khademi et al. 2017) algorithms, so as to comprehensively test the optimization performance of different algorithms. Specifically, we try to verify the path planning performance of different algorithms in three environments with different complexity. Circular obstacles, rectangular obstacles and labyrinth obstacles are designed in turn in the three environments, and the best path planning results of GEI-BBO in the three environments are shown in Fig. 6.

| Table 4 Path planning results of different algorithms |
|------------------------------------------------------|
| **Environment** | **Algorithm** | **Best** | **Worst** | **Mean** | **SD** |
|-----------------|--------------|---------|-----------|----------|--------|
| Environment 1   | GA           | 9.2584  | 11.2608   | 9.8816   | 0.4425 |
|                 | PSO          | 7.5602  | 9.0484    | 8.7848   | 0.5801 |
|                 | ACO          | 8.9149  | 8.9149    | 8.9149   | 0      |
|                 | BBO          | 9.3092  | 9.3092    | 9.3092   | 0      |
|                 | IWO/BBO      | 9.2580  | 9.2580    | 9.2580   | 0      |
|                 | GEI-BBO      | 7.5465  | 7.5465    | 7.5465   | 0      |
| Environment 2   | GA           | 29.5561 | 32.3786   | 30.7575  | 1.0769 |
|                 | PSO          | 28.9670 | 31.6581   | 29.7025  | 1.2418 |
|                 | ACO          | 31.4247 | 31.5697   | 31.4411  | 0.0021 |
|                 | BBO          | 28.9453 | 29.4278   | 29.1776  | 0.0462 |
|                 | IWO/BBO      | 28.8787 | 29.3112   | 29.0374  | 0.0309 |
|                 | GEI-BBO      | 28.8721 | 28.9298   | 28.8917  | 0.0009 |
| Environment 3   | GA           | 64.3348 | 107.7602  | 79.1527  | 201.8650 |
|                 | PSO          | 64.1951 | 79.2291   | 66.8617  | 21.7340 |
|                 | ACO          | 85.9004 | 104.1118  | 95.5918  | 35.6040 |
|                 | BBO          | 64.6743 | 69.2475   | 65.9671  | 2.1830 |
|                 | IWO/BBO      | 64.2394 | 67.4213   | 65.5749  | 0.8040 |
|                 | GEI-BBO      | 64.1673 | 64.1673   | 64.1673  | 0      |
Table 5  Success rate of path planning in different algorithms

| Environment | GA (%) | PSO (%) | ACO (%) | BBO (%) | IWO/BBO (%) | GEI-BBO (%) |
|-------------|--------|---------|---------|---------|-------------|-------------|
| Environment 1 | 100    | 90      | 100     | 100     | 100         | 100         |
| Environment 2 | 60     | 80      | 70      | 90      | 90          | 100         |
| Environment 3 | 20     | 50      | 60      | 60      | 80          | 100         |

Fig. 7  Box plots for the results of path planning in different environments

It can be seen intuitively from Fig. 6 that the path of the mobile robot planned by the proposed GEI-BBO algorithm rarely has unnecessary twists and turns. Specifically, in environment 1, the starting point coordinate is (0,0) and the ending point coordinate is (4,6). The obstacle consists of three circular obstacles, and the optimal path length is 7.5465. There are 12 rectangular obstacles in environment 2, the starting point coordinate is (0,0), the ending point coordinate is (20,20), and the optimal path length is 28.8721. Environment 3 is composed of more complex rectangular obstacles, the starting point coordinate is (0,0), the ending point coordinate is (45,45), and the optimal path length is 64.1673. Table 4 shows the path results of each algorithm running independently for 10 times, including best, worst, mean and standard deviation (SD) of path length. Meanwhile, the success rate of each algorithm in the 10 times algorithm is shown in Table 5.

Combining Tables 4 and 5, we can see that GEI-BBO has good performance in finding the optimal path. In environment 1, each algorithm can find its own optimal path, but compared with GA, PSO, ACO, BBO and IWO/BBO, GEI-BBO has the shortest path length, and the standard deviation is 0. In environment 2, except GEI-BBO, the success rate of other algorithms is less than 100%, and the optimal value and mean value obtained by GEI-BBO are optimal. In environment 3, GEI-BBO has the shortest path length of 64.1673, and the standard deviation is 0. Although other algorithms can also find the approximate optimal path with GEI-BBO, they are not very stable, and the success rate is low.

To make the experimental results more intuitive, we further add box plots as shown in Fig. 7. In environment 1, it can be seen that the variance of GA and PSO data is relatively large. Although the variance of ACO, BBO and IWO/BBO is 0, GEI-BBO has a smaller path length. In environment 2, the variance of GA and PSO is still larger, followed by BBO and IWO/BBO. The variance of ACO is similar to GEI-BBO, but its optimal path length is larger. In environment 3, except for ACO, the optimal path length of the other four algorithms is similar to GEI-BBO, but the variance is greater than GEI-BBO, especially GA. Generally, the path planned by the proposed GEI-BBO algorithm is better than the other five comparison algorithms. This is because the strategies of eigendecomposition-based migration, gradient descent and system search are introduced, which balances the capabilities of global search and local mining. Simulation experiments verify the feasibility and reliability of the combination of GEI-BBO and cubic spline interpolation method to solve the problem of MRPP.

6 Conclusion

In this paper, we proposed a new improved algorithm of BBO for MRPP, which combines the eigendecomposition-based migration, gradient descent and system search strategy. This method can effectively reduce the dependence of BBO on coordinate system and improve the local search ability. The comparison of 23 benchmark functions has proven the effectiveness of GEI-BBO. Combining GEI-BBO with cubic spline interpolation function, a method and fitness function for solving the obstacle avoidance and shortest path of mobile robot are constructed to solve the problem of MRPP. The simulation results also show that the proposed method has higher accuracy and success rate, which can prove the feasibility of the algorithm.
In the future, we will study how to use the new approaches to further enhance the performance of BBO, and apply the proposed algorithm to the more complex global path planning for mobile robots.

**Funding** This work was supported by the State Key Laboratory of Robotics (2019-O18) and the Fundamental Research Funds for the Central Universities (DUT20LJAB114, DUT2018TB06).

**Data availability** Enquiries about data availability should be directed to the authors.

**Declarations**

**Conflict of interest** The authors have not disclosed any competing interests.

**References**

Alam MS, Rafique MU (2015) Mobile robot path planning in environments cluttered with non-convex obstacles using particle swarm optimization. In: 2015 International conference on control, automation and robotics, pp. 32–36

Bakdi A, Bentout A, Boutami H, Maoudj A, Hachour O, Bouzouia B (2017) Optimal path planning and execution for mobile robots using genetic algorithm and adaptive fuzzy-logic control. Robot Auton Syst 89(89):95–109

Borenstein J, Koren Y (1991) Histogramic in-motion mapping for mobile robot obstacle avoidance. Int Conf Robot Autom 7(4):535–539

Castillo O, Trujillo L, Melin P (2006) Multiple objective genetic algorithms for path-planning optimization in autonomous mobile robots. Soft Comput A Fusion Found Method Appl Fuzzy-Neural Comput Robot Arch 11:269–279

Chen Yb, Luo Gc, Mei Ys, Yu Jq, Su Xl (2016) UAV path planning using artificial potential field method updated by optimal control theory. Int J Syst Sci 47(6):1407–1420

Contreras-Cruz MA, Ayala-Ramirez V, Hernandez-Belmonte UH (2015) Mobile robot path planning using artificial bee colony and evolutionary programming. Appl Soft Comput Arch 30:319–328

Du D, Simon D, Ergezer M (2009) Biogeography-based optimization combined with evolutionary strategy and immigration refusal. In: 2009 IEEE International conference on systems, man and cybernetics.

Hansen N, Ostermeier A (2001) Completely derandomized self-adaptation in evolution strategies. Evol Comput 9(2):159–195

Kennedy J, Eberhart R (1995) Particle swarm optimization. Proc ICNN’95-Int Conf Neural Netw 4:1942–1948

Khadem G, Mohammadi H, Simon D (2017) Hybrid invasive weed/biogeography-based optimization. Eng Appl Artif Intell 64:213–231

Kim J-J, Lee J-J (2015) Trajectory optimization with particle swarm optimization for manipulator motion planning. IEEE Trans Ind Inf 11(3):620–631

Kumar PB, Sahu C, Parhi DR (2018) A hybridized regression-adaptive ant colony optimization approach for navigation of humanoids in a cluttered environment. Appl Soft Comput 68:565–585

Lim KK, Ong YS, Lim MH, Chen X, Agarwal A (2008) Hybrid ant colony algorithms for path planning in sparse graphs. Soft Comput A Fusion Found Method Appl Arch 12:981–994

Liu J, Yang J, Liu H, X Tian, Gao M (2017) An improved ant colony algorithm for robot path planning. Soft Comput A Fusion Found Method Appl Arch 21:5829–5839

Manikas TW, Ashenayi K, Wainwright RL (2007) Genetic algorithms for autonomous robot navigation. IEEE Instrum Measurement Mag 10(6):26–31

Mavrovouniotis M, Yang S (2011) A memetic ant colony optimization algorithm for the dynamic travelling salesman problem. Soft Comput A Fusion Found Method Appl Arch 15:1405–1425

Mirjalili S (2016) Sca: a sine cosine algorithm for solving optimization problems. Knowl Based Syst 96(96):120–133

Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95(95):51–67

Mirjalili S, Mirjalili SM, Lewis A (2014) Let a biogeography-based optimizer train your multi-layer perceptron. Inf Sci 269(269):188–209

Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp swarm algorithm. Adv Eng Softw 114:163–191

Mo H, Xu L (2015) Research of biogeography particle swarm optimization for robot path planning. Neurocomputing 148:91–99

Qu H, Xing K, Alexander T (2013) An improved genetic algorithm with co-evolutionary strategy for global path planning of multiple mobile robots. Neurocomputing 120:509–517

Rasekhipour Y, Khajepour A, Chen S-K, Litkouhi B (2017) A potential field-based model predictive path-planning controller for autonomous road vehicles. IEEE Trans Intell Transp Syst 18(5):1255–1267

Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(6):702–713

Simon D (2011) A probabilistic analysis of a simplified biogeography-based optimization algorithm. Evol Comput 19(2):167–188

Simon D, Ergezer M, Du D, Ratric R (2010) Markov models for biogeography-based optimization. IEEE Trans Syst Man Cybern Part B 41(1):299–306

Simon D, Omran MGH, Clerc M (2014) Linearized biogeography-based optimization with re-initialization and local search. Inf Sci 267:140–157

Song B, Wang Z, Zou L (2017) On global smooth path planning for mobile robots using a novel multimodal delayed psd algorithm. Cogn Comput 9(1):5–17

Sucan IA, Moll M, Kavraki LE (2012) The open motion planning library. IEEE Robot Autom Mag 19(4):72–82

Tharwat A, Elhoseny M, Hassanien AE, Gabel T, Kumar A (2019) Intelligent bezier curve-based path planning model using chaotic particle swarm optimization algorithm. Clust Comput 22(2):4745–4766

Yang J, Li L (2017) Improved biogeography-based optimization algorithm for mobile robot path planning. Chin Intell Syst Conf. https://doi.org/10.1007/978-981-10-6499-9_22

Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3(2):82–102

Zeng N, Zhang H, Chen Y, Chen B, Liu Y (2016) Path planning for intelligent robot based on switching local evolutionary psd algorithm. Assem Autom 36(2):120–126

Zhu W, Duan H (2014) Chaotic predatorprey biogeography-based optimization approach for uav path planning. Aerosp Sci Technol 32(1):153–161

**Publisher’s Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.