RESEARCH ARTICLE

Mobile robot path planning with reformative bat algorithm

Gongfeng Xin[1], **Lei Shi**[2], **Guanxu Long**[1], **Weigang Pan**[2]*, **Yiming Li**[1], **Jicun Xu**[2]

1 Innovation Research Institute, Shandong Hi-speed Group Co. LTD, Jinan, Shandong, China, **2** School of Information Science and Electrical Engineering (School of Artificial Intelligence), Shandong Jiaotong University, Jinan, Shandong, China

* panweigang1980@163.com

Citation: Xin G, Shi L, Long G, Pan W, Li Y, Xu J (2022) Mobile robot path planning with reformative bat algorithm. PLoS ONE 17(11): e0276577. https://doi.org/10.1371/journal.pone.0276577

Editor: Vincent Yu, National Taiwan University of Science and Technology, TAIWAN

Received: March 31, 2022

Accepted: October 10, 2022

Published: November 4, 2022

Data Availability Statement: All relevant data are within the paper and its Supporting information files.

Funding: This study was supported by the Major Science and Technology Innovation Project of Shandong Province (2020CXGC010110), the Shandong Provincial Transportation Science and Technology Plan (2021B83), the Key Science and Technology Projects in Transportation Industry (2021-MS6-146), and the Shandong Provincial Natural Science Foundation (ZR2022MF345) through grants awarded to WP. The funders had no role in study design, data collection and

Abstract

Mobile robot path planning has attracted much attention as a key technology in robotics research. In this paper, a reformative bat algorithm (RBA) for mobile robot path planning is proposed, which is employed as the control mechanism of robots. The Doppler effect is applied to frequency update to ameliorate RBA. When the robot is in motion, the Doppler effect can be adaptively compensated to prevent the robot from prematurely converging. In the velocity update and position update, chaotic map and dynamic disturbance coefficient are introduced respectively to enrich the population diversity and weaken the limitation of local optimum. Furthermore, Q-learning is incorporated into RBA to reasonably choose the loudness attenuation coefficient and the pulse emission enhancement coefficient to reconcile the trade-off between exploration and exploitation, while improving the local search capability of RBA. The simulation experiments are carried out in two different environments, where the success rate of RBA is 93.33% and 90%, respectively. Moreover, in terms of the results of success rate, path length and number of iterations, RBA has better robustness and can plan the optimal path in a relatively short time compared with other algorithms in this field, thus illustrating its validity and reliability. Eventually, by the aid of the Robot Operating System (ROS), the experimental results of real-world robot navigation indicate that RBA has satisfactory real-time performance and path planning effect, which can be considered as a crucial choice for dealing with path planning problems.

Introduction

As the representative of high-end intelligent equipment and high-tech, mobile robot technology is changing with each passing day, which has been widely applied in family services, rescue and relief, warehousing and logistics, and other practical application fields. In order to achieve the shortest collision-free movement of the mobile robot from the starting point to the target point, the path planning of the mobile robot has become a hot spot of current research, and has attracted close attention of relevant scholars. To date, a variety of effective methods have been developed to deal with path planning problems, such as visibility graph [1, 2], artificial potential field (APF) [3, 4], rapidly-exploring random tree (RRT) [5], reinforcement learning

analysis, decision to publish, or preparation of the manuscript.

Competing interests: The authors have declared that no competing interests exist.

(RL) [6], ReinforcedRimJump (RRJ) [7, 8], nonlinear control [9], etc. Nevertheless, with the increase of environment complexity and task difficulty, the above path planning methods are hard to achieve desired effects. The path drawn by the visibility graph or the RRT is composed of multiple straight lines, resulting in the path is not smooth enough. The APF is easy to get trapped into local optima, moreover the phenomenon that the target point is unreachable may occur. For the RL, it is difficult to use less resources to address the path planning problem in complex environments. As an emerging algorithm, RRJ can achieve the shortest path planning, but it is only suitable for static environment, which seriously affects its practical application value.

Since the establishment of swarm intelligence (SI) [10], it has become a research field of great concern, bringing hope to solve complex optimization problems. The inspiration of SI mainly comes from the collective behavior patterns of ants, bees, bats, and other biological groups. All of these creatures search their targets through common wisdom and experience. The SI-based optimization algorithms simulate the behavioral attributes of biological populations, including particle swarm optimization (PSO) [11, 12], teaching-learning-based optimization (TLBO) [13, 14], artificial bee colony (ABC) optimization [15–17], ant colony optimization (ACO) [18–23], firefly algorithm (FA) [24, 25], bat algorithm (BA) [26–31], whale optimization algorithm (WOA) [32], etc. As a classic swarm intelligence optimization algorithm, PSO is frequently utilized to handle mobile robot path planning problems due to its simple structure, high search efficiency, and easy improvement. So far, many valuable research results have emerged [33–36]. Mo and Xu [33] proposed a novel approach for global path planning in a static environment that hybridizes biogeography-based optimization (BBO) and PSO. Tang et al. [34] introduced a hybrid PSO that combines PSO and differential evolution (DE) algorithm. Mac et al. [35] conducted a more in-depth study on the path planning problem of mobile robots in complex environments and presented a constrained multi-objective PSO. However, the above PSO variants are prone to fall into local optima, making it difficult to efficiently complete optimal path planning. Li and Chou [36] came up with a SLPSO algorithm and comprehensively considered constraints such as path length, collision risk degree and smoothness to generate a feasible collision-free path. Nevertheless, the robustness of the algorithm is not satisfactory, that is, as the complexity of the environment increases, the path planning effect of the algorithm declines.

The BA, first introduced in 2010, is similar to PSO, however, it has better convergence and can balance exploration and exploitation well when searching for the global optimum. Consequently, BA is increasingly favored by researchers, and a number of well-known BA variants have been advanced one after another. Liu et al. [37] put forward a modified BA, called PTRBA, to process the global path planning problem of single-robot or multi-robots. In order to improve the optimization performance of BA, the dynamic perturbation coefficient is introduced into the position update in the global search stage, and the tangent random exploration mechanism is integrated into the local search stage. Eventually, the PTRBA and cubic spline interpolation are combined to form a smooth and feasible path. In reference [38], an adaptive robotic bat algorithm (ARBA) was put forth to handle the multirobot target searching problem. The adaptive inertial weight strategy is added to the velocity update to improve the diversity of ARBA. Furthermore, the Doppler effect and multi-swarm strategy are introduced into ARBA to assist robots to better accomplish target searching. Based on the above description, BA variants have many merits, but there are still some challenges to be solved. For instance, loudness attenuation coefficient and pulse emission enhancement coefficient are the key elements that influence the balance between exploration and exploitation of BA. If the above two parameters are not properly coordinated, the optimization performance of BA will be affected, making it hard to guarantee the path planning effect. However, the BA variants described above do not take this factor into account. Hence, there is still plenty of room for improvement in their performance.

In order to further improve BA and better complete the path planning task in static environments, this paper puts forward a reformative BA, named RBA, in which all robots are regarded as bats, and one robot represents one bat. Moreover, RBA is employed as the robots' control mechanism to realize the robots' search for the target, thereby accomplishing the path planning task. The main contributions of RBA are highlighted in the following aspects: (1) The Doppler effect is applied to the frequency update to ameliorate RBA. When the robot is in motion, the Doppler effect can be adaptively compensated to prevent the robot from prematurely converging. (2) In the velocity update and position update, chaotic map and dynamic disturbance coefficient are introduced respectively to enrich the population diversity and weaken the limitation of local optimum. (3) Q-learning is adopted to make reasonable choices for the loudness attenuation coefficient and the pulse emission enhancement coefficient to coordinate the trade-off between exploration and exploitation, while improving the local search capability of RBA. To verify the validity and reliability of RBA, simulation experiments are carried out in two different environments. To begin with, the original RBA is compared with five classical swarm intelligence optimization algorithms, including PSO, BA, FA, TLBO and WOA. The experimental results demonstrate that RBA has good comprehensive performance and can effectively and reliably implement the optimal path planning. Subsequently, RBA is compared with four PSO variants, namely BPSO [33], PSO-DE [34], CMOPSO [35] and SLPSO [36]. Experimental results show that contrasted with PSO variants, RBA has superior search performance and stronger robustness. Finally, the proposed RBA is compared with three other state-of-the-art BA variants, i.e. EBat [28], PTRBA [37] and ARBA [38]. Experimental results indicate that RBA can give consideration to optimization effect and computational efficiency, and has excellent robustness. With the help of ROS, real-world robot navigation experiments are also carried out. The related results reveal that RBA has satisfactory real-time performance and path planning effect, and can be considered as a crucial choice for dealing with path planning problems.

The remainder of this paper is organized as follows. In 'Bat algorithm' and 'Q-learning', we review the knowledge of BA and Q-learning, respectively. The proposed RBA is described in detail in 'Reformative bat algorithm (RBA)'. To evaluate the proposed approach, simulation experiments are conducted in 'Simulation testing' and real-world robot navigation experiments are finished in 'Real-world case'. In the end, conclusions are drawn and future work is provided in 'Conclusions and future work'.

Bat algorithm

BA was first introduced in 2010, inspired by bats' echolocation behavior in search of prey. In nature, bats emit ultrasonic pulses and analyze reflected ultrasonic waves to determine the information of prey. Besides, bats can search for prey by changing their ultrasonic frequency, velocity and position. In the process of approaching prey, bats will increase the emissivity of ultrasonic pulses and weaken the loudness. The implementation of BA is based on the following assumptions. (1) All bats use echolocation to sense distance, and they can accurately distinguish between prey and obstacles. (2) Bats can automatically adjust the frequency and emissivity of the pulses according to the proximity of the target. (3) It is assumed that the loudness changes from a maximum value to a fixed minimum value.

The frequency, velocity and position values of each bat can be calculated as

$$f_i = f_{min} + (f_{max} - f_{min}) \cdot \beta, \tag{1}$$

$$v_i^t = v_i^{t-1} + (x_i^t - x^*) \cdot f_i, \tag{2}$$

$$x_i^t = x_i^{t-1} + v_i^t, \tag{3}$$

where f_{max} and f_{min} are the maximum and minimum values of the search pulse frequency, respectively; $\beta \in [0, 1]$ is a uniformly distributed random number; x^* indicates the optimal position of all current bats.

For the local search stage, a new result is performed in accordance with the following:

$$x_{new} = x_{old} + \epsilon A^t, \tag{4}$$

where x_{old} is the current best solution, x_{new} is the new solution generated after the local search; $\epsilon \in [-1, 1]$ is a random number; A^t is the average loudness of all bats at iteration t.

The iterative equations for loudness A_i and pulse emissivity r_i are expressed as follows:

$$A_i^{t+1} = \alpha A_i^t, \tag{5}$$

$$r_i^{t+1} = r_i^0 \cdot [1 - \exp(-\gamma t)], \tag{6}$$

where α and γ are constants; r_i^0 is the initial pulse emissivity. For any $0 < \alpha < 1$ and $\gamma > 0$, we have $A_i^t \rightarrow 0, r_i^t \rightarrow r_i^0$, as $t \rightarrow +\infty$.

The pseudo code of BA is listed in Algorithm 1. As can be seen from Algorithm 1, the pulse emissivity r_i controls whether BA can perform local search, and the loudness A_i determines the local search performance of BA. Furthermore, according to Eqs (5) and (6), it is distinct that the loudness attenuation coefficient α and the pulse emission enhancement coefficient γ play a vital role in the iterative process of loudness and pulse emissivity, respectively. Therefore, in order to effectively coordinate the balance between exploration and exploitation and improve the local search capability of BA, it is necessary to reasonably choose the loudness attenuation coefficient and the pulse emission enhancement coefficient. In this paper, Q-learning is employed to tackle this issue. The details will be given in 'Parameters preselection'.

Q-learning

Q-learning is a trial and error learning method, whose purpose is to learn optimal strategies to accumulate rewards, so as to maximize the Q-value. The Q-value is updated as follows:

$$Q(s_t, a_t) \leftarrow (1-\mu)Q(s_t, a_t) + \mu[re(s_t, a_t) + \eta \max_{a_{t+1}} Q(s_{t+1}, a_{t+1})], \tag{7}$$

where $re(s_t, a_t)$ is an immediate reward; η is a discount factor; μ is the learning rate, which controls the learning speed. Within a certain range of values, the larger the μ, the faster the convergence.

In this paper, greedy strategy is chosen as action selection strategy. The greedy strategy, as the name implies, aims to select the action that maximizes the Q-value. The relevant equation is expressed as

$$a_t = \arg\max_{a_t} Q(s_t, a_t). \tag{8}$$

Reformative bat algorithm (RBA)

As mentioned in 'Introduction', BA has both advantages and challenges. Thus, in this section, the RBA is proposed to address the corresponding challenges and significantly improve the BA. On the one hand, the Doppler effect, chaotic map and dynamic disturbance coefficient are utilized to assist RBA to avoid premature convergence and weaken the limitation of local optimum. On the other hand, by means of Q-learning, RBA can effectively solve the challenges of BA caused by the poor coordination between loudness attenuation coefficient and pulse emission enhancement coefficient.

Algorithm 1 Pseudo code of BA.

```
Determine the fitness function $fit(x)$, $x = (x_1, x_2, \cdots, x_d)^T$;
Generate bat population $x_i$ $(i = 1, 2, \cdots, m)$ and initial velocity $v_i$
($i = 1, 2, \cdots, m$);
Define pulse frequency $f_i$ at $x_i$;
Initialize values for pulse emissivity $r_i$ and loudness $A_i$;
while $t \leq T_{max}$ do
  Adjust frequency by Eq (1);
  Update velocities by Eq (2);
  Update positions by Eq (3);
  if $rand > r_i$ then
    Select a best position;
    Generate a local position by Eq (4);
  end
  if $rand < A_i$ and $fit(x_i) > fit(x^*)$ then
    Accept the new position;
    Update $A_i$ by Eq (5);
    Update $r_i$ by Eq (6);
  end
  Find the current best $x^*$;
  if target is reached or stop condition is met then
    break;
  end
end
Show the results
```

Doppler effect

According to Eq (1), we can intuitively see that the frequency update of BA has a strong randomness, resulting in the planned path is not smooth enough, and premature convergence may occur. Consequently, the Doppler effect is introduced to ameliorate the frequency update of BA. The improved frequency calculation formula is expressed as

$$f_i = f_{min} + (f_{max} - f_{min}) \cdot \xi_i, \tag{9}$$

$$\xi_i = \left(\frac{v \pm v_i^t}{v \mp v_s} \right) \cdot \xi_0, \tag{10}$$

where ξ_i is the observation frequency, ξ_0 is the original emission frequency of the emission source (target); v is the velocity of wave propagation; v_i^t is the movement velocity of the observer (robot), if the observer is close to the emission source, the operator in front is "+", otherwise it is "-"; v_s is the movement velocity of the emission source, if the emission source is close to the observer, the operator in front is "-", otherwise it is "+".

In the light of Eq (10), we can discover that in the Doppler effect, the frequency will change as the distance between the robot and the target changes. Hence, the robot can adaptively compensate for the Doppler effect during the movement, and then regulate the velocity by adaptively adjusting the frequency, thereby avoiding premature convergence.

Improved model for velocity and position

In RBA, the velocity and position values can be updated as

$$v_i^t = v_i^{t-1} + \sigma \cdot (x_i^t - x^*) \cdot f_i, \tag{11}$$

$$\sigma = \sigma_t = \zeta \cdot \sin(\pi \cdot \sigma_{t-1}), \;\; \sigma_t \in (0,1), \;\; t = 1,2,\cdots,T_{max}, \tag{12}$$

$$x_i^t = \omega \cdot x_i^{t-1} + v_i^t, \tag{13}$$

$$\omega = 1 - \sin\left(\frac{\pi t}{2 \cdot T_{max}}\right) + \tau \cdot betarnd(). \tag{14}$$

The standard BA uses Eq (3) to update the position, in which the calculation of v_i^t is inseparable from $x_i^t - x^*$. Hence, when conducting the global search, BA is directly constrained by $x_i^t - x^*$, and it is easy to fall into local optima. In response to this problem, the attenuation coefficient σ is introduced in Eq (12). Since chaotic map has the merits of ergodicity, non-repeatability and sensitivity, we select chaotic map to update σ, where $\zeta \in (0, 1)$ is a constant and t represents the current iteration number. Based on Eq (12), it is evident that the value range of σ always belongs to (0, 1). Therefore, the limitation of local optimum is reduced. In addition, the dynamic disturbance coefficient ω is put forward as shown in Eq (14), where τ is the disturbance deviation factor and $betarnd()$ is a random number obeying the beta distribution. The dynamic disturbance coefficient ω decreases adaptively with the increase of the number of iterations. Consequently, in the early stage, the dynamic disturbance coefficient ω has a large disturbance to the position update, which is conducive to expanding the search scope of bats. In the later stage, the dynamic disturbance coefficient ω reduces the disturbance to the position update, which is beneficial to the stability of the algorithm. Through many experiments, the constant ζ and the disturbance deviation factor τ are set to 0.5 and 0.1, respectively.

Parameters preselection

In BA, the quality of optimization results is determined by loudness attenuation coefficient α and pulse emission enhancement coefficient γ. If the above parameters are not properly coordinated, the convergence speed of BA will be affected, making it difficult to ensure the path planning effect. Therefore, in the local search phase, Q-learning is applied to preselect the optimal combinations of the above parameters to ameliorate the optimization effect of BA. The relevant idea is displayed in Fig 1.

In Fig 1, $< \alpha, \gamma >$ set is composed of the loudness attenuation coefficient α and the pulse emission enhancement coefficient γ, and a $< \alpha, \gamma >$ combination corresponds to an action in Q-learning. $X_i(t)$ is defined as the position of the ith bat at iteration t. Moreover, $R_i(t)$ is the fitness function value of the bat at position $X_i(t)$, which is defined as the state of Q-learning. The combination of BA and Q-learning can be described as selecting the optimal combination $< \alpha', \gamma' >$ from the $< \alpha, \gamma >$ set according to Eq (8) when the state is $R_i(t)$. In BA, the optimal combination $< \alpha', \gamma' >$ is utilized to obtain the next position $X_i(t+1)$ of the bat, and then the Q-value of the next state $R_i(t+1)$ is estimated. On the other hand, when the optimal action $< \alpha', \gamma' >$ acts on the environment, the corresponding immediate reward $re(R_i(t), < \alpha', \gamma' >)$ will be generated. The immediate reward is set to the difference between the fitness function

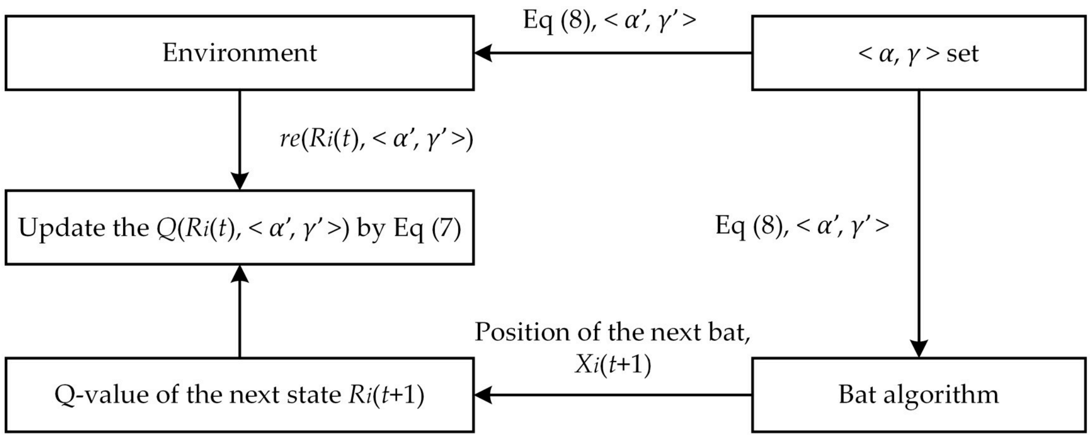

Environment
Eq (8), < α′, γ′ >
< α, γ > set
re(Ri(t), < α′, γ′ >)
Update the Q(Ri(t), < α′, γ′ >) by Eq (7)
Eq (8), < α′, γ′ >
Position of the next bat, Xi(t+1)
Q-value of the next state Ri(t+1)
Bat algorithm

Fig 1. The combination of bat algorithm and Q-learning.

https://doi.org/10.1371/journal.pone.0276577.g001

values of the bats in successive iterations. The related equation is executed as follows:

$$re(R_i(t), < \alpha', \gamma' >) = fit(X_i(t+1)) - fit(X_i(t)) = R_i(t+1) - R_i(t). \tag{15}$$

Finally, $Q(R_i(t), <\alpha', \gamma'>)$ is updated in accordance with Eq (7).

Owing to the application of the Q-learning, in the local search phase, each bat position has its corresponding optimal $< \alpha', \gamma' >$ combination, and all the information is saved in the Q-table. In the implementation stage, RBA can directly select the optimal $< \alpha', \gamma' >$ combinations from the Q-table, thus overcoming the defects of the standard BA due to the parameters are not well coordinated.

Fitness function

In this paper, the fitness function is designed in the light of the following evaluation criteria. (1) No collision with obstacles. (2) Achieve the shortest path length. The corresponding fitness function is expressed as

$$fit = \frac{1}{L \cdot (1 + \bar{p} \cdot \lambda)}, \tag{16}$$

where L is the path length of the mobile robot from the starting point to the target point, which conforms to Eq (17),

$$L = \sum_{i=1}^{n} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}. \tag{17}$$

$\bar{p}$ is the penalty term used to exclude paths that collide with obstacles. The value of $\bar{p}$ is set to 100. λ is the flag variable with an initial value of 0. The update process of λ is as follows:

for k = 1: *nobs*

$$d_k = \sqrt{(xx - xobs_k)^2 + (yy - yobs_k)^2}, \tag{18}$$

$$\theta_k = \max\left(1 - \frac{d_k}{robs_k}, 0\right), \tag{19}$$

$$\lambda = \lambda + mean(\theta_k). \tag{20}$$

end

Given that the robot has a certain volume, the obstacles are expanded to prevent the robot from hitting the obstacles. *nobs* is the total number of obstacles. ($xobs_k$, $yobs_k$) and $robs_k$ are the center coordinate and maximum influence radius of the *k*th expanded obstacle, respectively. d_k is the distance from the point on the path to the center coordinate of the obstacle. For λ, if there is no collision between the robot and the obstacle, then λ = 0. However, if the robot collides with the obstacle, λ is a positive number greater than 0. Hence, when the fitness function *fit* reaches the maximum value, the shortest collision-free path can be obtained.

Implementation of RBA

After model improvement and parameters preselection, RBA will be implemented into path planning. In the global search stage, the Doppler effect, attenuation coefficient and dynamic disturbance coefficient are added to the RBA. Consequently, unlike standard BA, the frequency, velocity and position values in RBA are updated according to Eqs (9)–(14). In the local search stage, RBA can directly select the corresponding optimal $<\alpha', \gamma'>$ combination from the Q-table on the basis of the current position of the bat, which can significantly improve the optimization performance of the algorithm.

The pseudo code of RBA is given in Algorithm 2.

Algorithm 2: Pseudo code of RBA.

```
Determine the fitness function $fit(x)$, $x = (x_1, x_2, \cdots, x_d)^T$;
Generate bat population $x_i$ $(i = 1, 2, \cdots, m)$ and initial velocity $v_i$
$(i = 1, 2, \cdots, m)$;
Define pulse frequency $f_i$ at $x_i$;
Initialize values for pulse emissivity $r_i$ and loudness $A_i$;
while $t \leq T_{max}$ do
  Adjust frequency by Eqs (9) and (10);
  Update velocities by Eqs (11) and (12);
  Update positions by Eqs (13) and (14);
  if $rand > r_i$ then
    Select a best position;
    Select the optimal combination $< \alpha', \gamma' >$ from the Q-table;
    Generate a local position by Eq (4);
  end
  if $rand < A_i$ and $fit(x_i) > fit(x^*)$ then
    Accept the new position;
    Update $A_i$ by Eq (5);
    Update $r_i$ by Eq (6);
  end
  Find the current best $x^*$;
  if target is reached or stop condition is met then
    break;
  end
end
Show the results
```

Complexity analysis of RBA

The time computational complexity of the proposed RBA can be expressed as $O((T + M_1) \times (N + M_2) \times D)$, where T is the number of iterations, N is the population size, D is the dimension of the path planning problem to be addressed, M_1 is the computation time of the attenuation coefficient σ and dynamic disturbance coefficient ω, and M_2 indicates the computation time for the observation frequency ξ_i. For the original BA, its time computational complexity can be described as $O(T \times N \times D)$. From Eqs (10), (12) and (14), it is apparent that only simple numerical operations are involved in M_1 and M_2. Hence, the time computational complexity of the proposed RBA is only slightly increased compared to that of BA.

Simulation testing

Experimental setup

In order to verify the validity and feasibility of the proposed RBA, five classical swarm intelligence optimization algorithms (PSO, BA, FA, TLBO and WOA), four PSO variants (BPSO, PSO-DE, CMOPSO and SLPSO), and three BA variants (EBat, PTRBA and ARBA) are selected to compare with RBA. To ensure the objectivity and fairness of the algorithm comparison, all experiments are conducted in Windows 10 environment, using Intel(R) Core(TM) i7–8750H 2.2GHz CPU and 8GB RAM, and all algorithms are implemented in MATLAB R2018b. Two static maps of different complexity are constructed, in which the number of obstacles is 9 and 13, respectively, as shown in Figs 5(a) and 9(a). The scale of both maps is 10×10, where the yellow square and green star are the starting point and the target point, respectively. We run the proposed RBA and baseline algorithms 30 times on each map and calculate the mean and standard deviation of the experimental results for comparison. The experimental results include the path length planned by each algorithm and the iteration number required by each algorithm to complete the path planning.

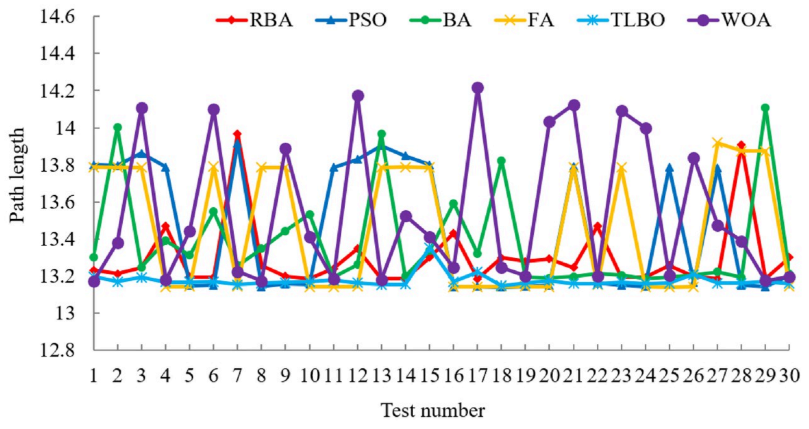
Based on previous researches, the key parameters of the above algorithms are fairly chosen as follows. The population size and the maximum number of iterations are set to $npop = 100$ and $T_{max} = 100$, respectively. For PSO, BPSO, PSO-DE, CMOPSO and SLPSO, inertia weight and acceleration coefficients are selected as $\omega = 1$ and $c_1 = c_2 = 1.5$, respectively. Especially, in PSO-DE, scaling factor $F = 0.5$ and crossover rate $CR = 0.5$. For FA, randomization parameter $\alpha = 0.5$, light absorption coefficient $\gamma = 1$, highest attractiveness $\beta_0 = 2$ and constant $m = 2$. In BA, EBat, PTRBA and ARBA, the numerical settings of the loudness attenuation coefficient and the pulse emission enhancement coefficient are the same, i.e. $\alpha = \gamma = 0.9$. Different from BA, EBat, PTRBA and ARBA, RBA selects the optimal α and γ values from the Q-table.

Test case 1

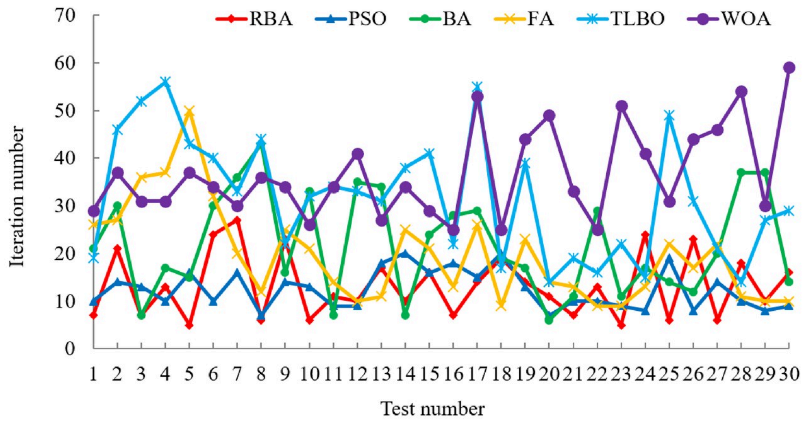
The map used in test case 1 contains nine obstacles. The shortest collision-free path on this map is shown in Fig 5(a), where (0, 0) is the starting point, (8, 10) is the target point, and the optimal path length is approximately 13.1716.

Comparison with classical algorithms. Five classical optimization algorithms are compared with our approach to demonstrate the superiority of the proposed RBA. In order to objectively analyze the performance of the algorithms and avoid contingency, each algorithm runs 30 times on the map. After 180 experiments, the experimental results are shown in Fig 2. In the 30 experiments of each algorithm, RBA realizes the optimal path 28 times, PSO realizes the optimal path 17 times, BA realizes the optimal path 26 times, FA realizes the optimal path 16 times, TLBO realizes the optimal path 30 times, and WOA realizes the optimal path 20 times. Consequently, the success rates of the above algorithms are 93.33%, 56.67%, 86.67%, 53.33%, 100%, and 66.67%, respectively.

For an in-depth understanding of the distribution of the experimental data in Fig 2, the mean and standard deviation of the relevant data are listed in Table 1. According to Fig 2(a) and Table 1, the path length data curve of TLBO is very stable. This is because in 30 experiments, TLBO has planned the optimal path every time, which also confirms that TLBO has the



(a)



(b)

Fig 2. Experimental results of six algorithms in test case 1. (a) Path length. (b) Iteration number.

https://doi.org/10.1371/journal.pone.0276577.g002

best ability to search for the global optimum. Excluding TLBO algorithm, among the remaining algorithms, it is clear that RBA has better performance compared with other algorithms, and its path length data curve shows relatively small fluctuations. In the light of Table 1, it is obvious that the average path length of RBA is 13.3019, and the standard deviation of the path length is only 0.1913. In terms of the number of iterations, as can be seen intuitively from Fig 2(b) and Table 1, the average number of iterations required for PSO to accomplish the path planning is the least, which is 12.43. The second is RBA, with an average number of iterations of 13.2. Although PSO can fulfill the path planning quickly, it has the defect that it is easy to fall into the local optima and cannot plan the optimal path effectively. This can be verified

Table 1. Performance comparison between RBA and classical optimization algorithms in test case 1.

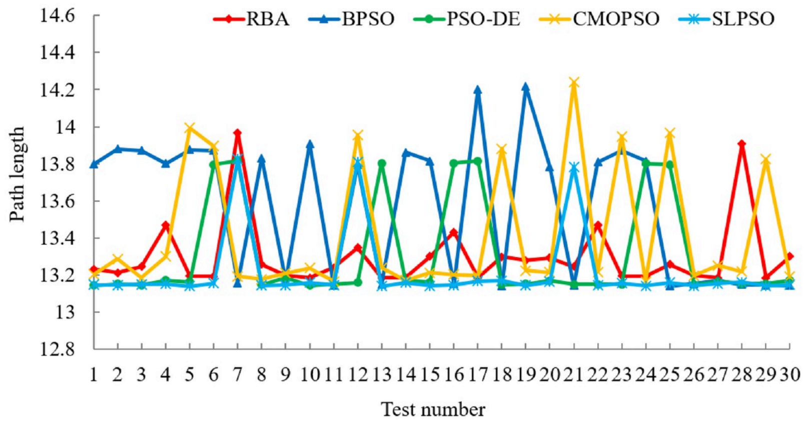
Algorithm	RBA	PSO	BA	FA	TLBO	WOA
Average Path Length	13.3019	13.4432	13.3805	13.4543	13.1762	13.5402
Standard Deviation of Path Length	0.1913	0.3391	0.2654	0.3386	0.0373	0.3905
Average Iteration Number	13.2	12.43	21.87	19.6	31.8	36.67
Standard Deviation of Iteration Number	6.6974	4.0402	10.7663	9.9848	12.7344	9.4989

https://doi.org/10.1371/journal.pone.0276577.t001

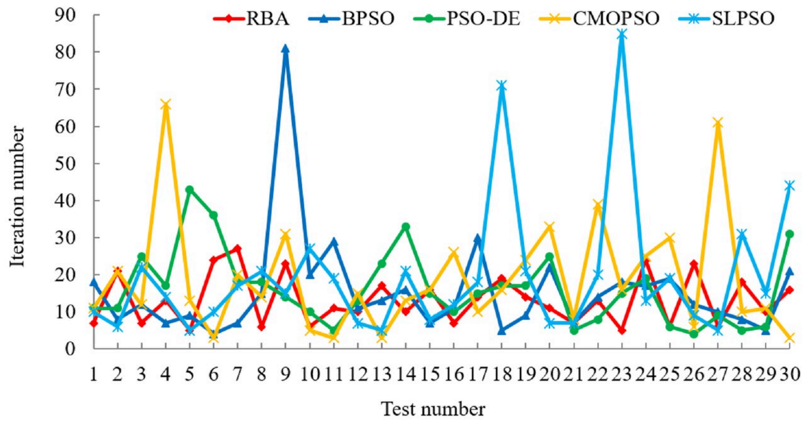
from the optimal path planning success rate, Fig 2(a) and Table 1. Relative to the excellent performance in path length, TLBO does not perform satisfactorily in the number of iterations. In 30 experiments, the average number of iterations of TLBO is 31.8, and the standard deviation of the number of iterations is as high as 12.7344.

Furthermore, under the same experiment, the path planning results of the six algorithms are displayed in Fig 5(b). It is obvious that PSO, BA and FA are all trapped in local optima, and only RBA, TLBO and WOA plan the shortest path. Among them, in order to complete the optimal path planning, TLBO requires 8 iterations, WOA requires 60 iterations, while RBA only requires 5 iterations. Therefore, contrasted with the classical optimization algorithms, RBA can achieve the optimal path planning in a relatively short time, and the success rate can reach 93.33%, which demonstrates that RBA has the merits of rapid optimization speed and good optimization effect.

Comparison with PSO variants. In order to compare the path planning effects of RBA and PSO variants, 150 experiments are fulfilled, and the related experimental results are exhibited in Fig 3. In the 30 experiments of each algorithm, RBA realizes the optimal path 28 times, BPSO realizes the optimal path 13 times, PSO-DE realizes the optimal path 23 times, CMOPSO realizes the optimal path 22 times, and SLPSO realizes the optimal path 27 times.



(a)



(b)

Fig 3. Experimental results of five algorithms in test case 1. (a) Path length. (b) Iteration number.

https://doi.org/10.1371/journal.pone.0276577.g003

Table 2. Performance comparison between RBA and PSO variants in test case 1.

Algorithm	RBA	BPSO	PSO-DE	CMOPSO	SLPSO
Average Path Length	13.3019	13.5662	13.3084	13.413	13.2171
Standard Deviation of Path Length	0.1913	0.3808	0.2787	0.3445	0.2003
Average Iteration Number	13.2	15.5	16.1333	18.8	19.4667
Standard Deviation of Iteration Number	6.6974	14.0755	9.8811	15.4661	18.2204

https://doi.org/10.1371/journal.pone.0276577.t002

Consequently, the success rates of the above algorithms are 93.33%, 43.33%, 76.67%, 73.33%, and 90%, respectively. On the basis of the success rate of each algorithm, it is obvious that in addition to SLPSO, other PSO variants have relatively poor performance and are prone to fall into local optimum, making it difficult to achieve optimal path planning.

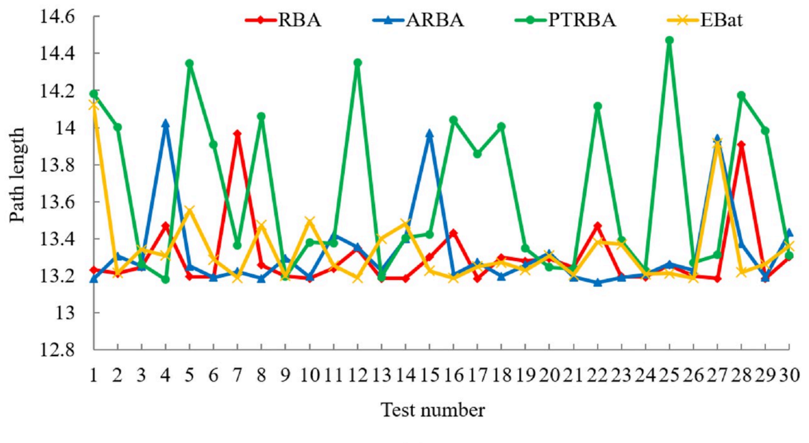
To clearly analyze the experimental data in Fig 3, the mean and standard deviation of the corresponding data are shown in Table 2. It can be seen intuitively from Table 2 that the average path length of the five algorithms is roughly the same, while RBA has the smallest standard deviation of path length, which indicates that RBA has a more stable path planning effect. In terms of the number of iterations, RBA achieves the smallest mean and standard deviation values, which means that RBA can accomplish optimal path planning faster than PSO variants. Moreover, under the same experiment, the path planning results of the five algorithms are exhibited in Fig 5(c). Obviously, except BPSO, other algorithms plan the shortest path, among which, PSO-DE requires 9 iterations, CMOPSO requires 49 iterations, SLPSO requires 16 iterations, while RBA only requires 4 iterations.

Comparison with BA variants. To finish the performance comparison between RBA and other novel BA variants, we collate 120 experimental data in Fig 4 and present the mean and standard deviation of the relevant data in Table 3. In the 30 experiments of each algorithm, RBA realizes the optimal path 28 times, ARBA realizes the optimal path 27 times, PTRBA realizes the optimal path 17 times, and EBat realizes the optimal path 28 times. Consequently, the success rates of the above algorithms are 93.33%, 90%, 56.67%, and 93.33%, respectively. Based on Fig 4 and Table 3, it is distinct that the path planning effect of PTRBA is relatively poor. In our opinion, this is because the tangent random exploration mechanism is applied in the local search phase of PTRBA, which replaces ϵ in Eq (4). The tangent random exploration mechanism is represented as $\tan(\pi \cdot (\xi - 0.5))$, where ξ is a random number belonging to [0, 1]. When the value of ξ approaches 0 or 1, the value of the tangent function approaches infinity. Therefore, in the iterative process of PTRBA, the phenomenon that the value of the tangent function is too large may occur, which influences the stability of the algorithm and reduces the path planning effect. For ARBA and EBat, their optimization performance is roughly the same and better than that of PTRBA. Compared with the aforementioned BA variants, RBA has more excellent path planning effects, not only in the path length but also in the number of iterations, thus verifying the superiority of RBA.

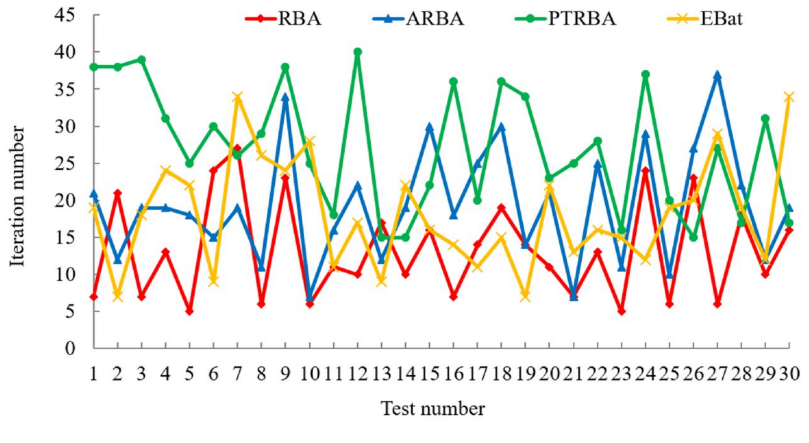
Besides, under the same experiment, the path planning results of the four algorithms are displayed in Fig 5(d). As the above analysis of the shortcomings of PTRBA, although PTRBA converges quickly, it plans a relatively long path and has poor optimization effect. In contrast with PTRBA, other algorithms plan the shortest path, among which, ARBA requires 40 iterations, EBat requires 50 iterations, while RBA only requires 9 iterations.

Test case 2

In order to further demonstrate the superiority of RBA, a more complex map is used in test case 2, which contains thirteen obstacles. The shortest collision-free path on this map is drawn



(a)



(b)

Fig 4. Experimental results of four algorithms in test case 1. (a) Path length. (b) Iteration number.

https://doi.org/10.1371/journal.pone.0276577.g004

in Fig 9(a), where (0, 0) is the starting point, (8, 10) is the target point, and the optimal path length is approximately 13.1966.

Comparison with classical algorithms. In the comparison experiment between RBA and five classical optimization algorithms, after 180 experiments, the experimental results are shown in Fig 6. In addition, the mean and standard deviation of the related experimental results are listed in Table 4. In the 30 experiments of each algorithm, RBA realizes the optimal path 27 times, PSO realizes the optimal path 16 times, BA realizes the optimal path 24 times, FA realizes the optimal path 16 times, TLBO realizes the optimal path 29 times, and WOA realizes the optimal path 18 times. Consequently, the success rates of the above algorithms are

Table 3. Performance comparison between RBA and BA variants in test case 1.

Algorithm	RBA	ARBA	PTRBA	EBat
Average Path Length	13.3019	13.3313	13.6544	13.3442
Standard Deviation of Path Length	0.1913	0.2325	0.4308	0.2124
Average Iteration Number	13.2	19.37	27.03	18.13
Standard Deviation of Iteration Number	6.6974	7.757	8.3438	7.3472

https://doi.org/10.1371/journal.pone.0276577.t003

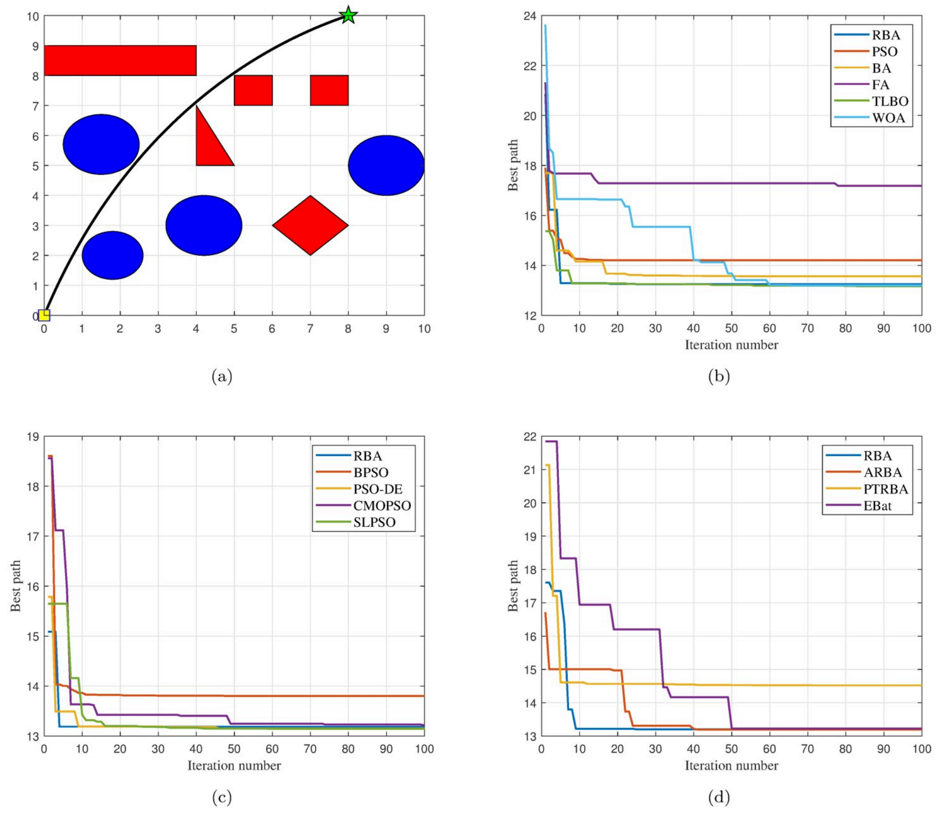


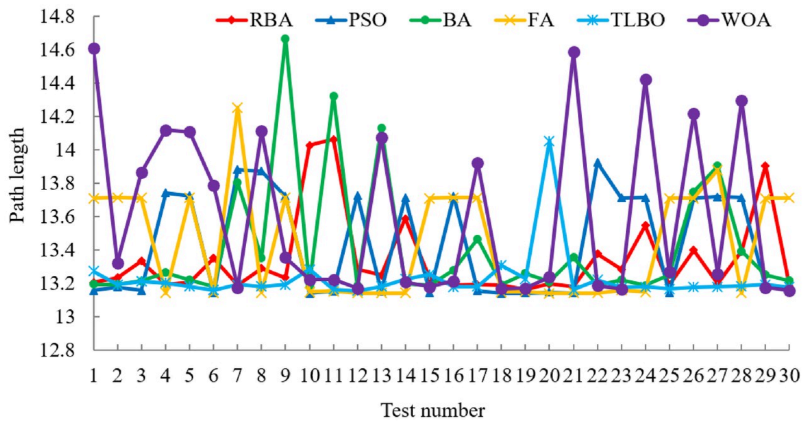
Fig 5. The path planning results in the test case 1 and the number of iterations of all algorithms when the path is implemented. (a) Optimal path. (b) Iteration curves of RBA and classical algorithms. (c) Iteration curves of RBA and PSO variants. (d) Iteration curves of RBA and BA variants.

https://doi.org/10.1371/journal.pone.0276577.g005

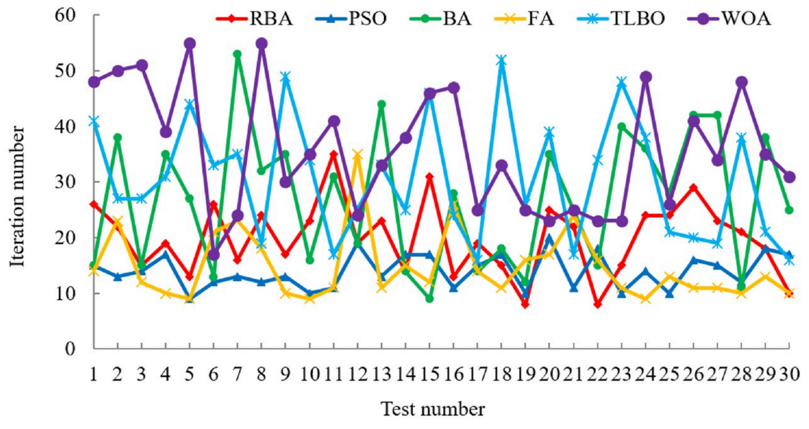
90%, 53.33%, 80%, 53.33%, 96.67%, and 60%, respectively. For RBA, compared to the results of test case 1, the performance is slightly decreased. The probability of realizing the optimal path is reduced by 3.33%.

On the basis of Table 4, it is clear that the average path length of RBA is 13.3436, and the standard deviation of the path length is 0.2469. Excluding TLBO algorithm, the RBA performs better than other algorithms. In terms of the number of iterations, PSO and FA can accomplish the path planning task faster than RBA. However, they are difficult to achieve the optimal path planning, and the success rate of optimal path planning is relatively low. Moreover, under the same experiment, the path planning results of the six algorithms are depicted in Fig 9(b). It is distinct that only RBA, BA, and TLBO fulfill the shortest path planning. Among them, in order to implement the optimal path, BA requires 10 iterations, TLBO requires 27 iterations, while RBA only requires 8 iterations. Thus, contrasted with classical optimization algorithms, RBA has good overall performance, not only achieves good path planning effect, but also has satisfactory robustness.

Comparison with PSO variants. To compare the optimization performance of RBA and PSO variants, 120 experiments are conducted and the experimental data are presented in Fig 7. In the 30 experiments of each algorithm, RBA realizes the optimal path 27 times, BPSO realizes the optimal path 12 times, PSO-DE realizes the optimal path 18 times, CMOPSO realizes the optimal path 18 times, and SLPSO realizes the optimal path 20 times. Consequently, the success rates of the above algorithms are 90%, 40%, 60%, 60%, and 66.67%, respectively. For



(a)



(b)

Fig 6. Experimental results of six algorithms in test case 2. (a) Path length. (b) Iteration number.

https://doi.org/10.1371/journal.pone.0276577.g006

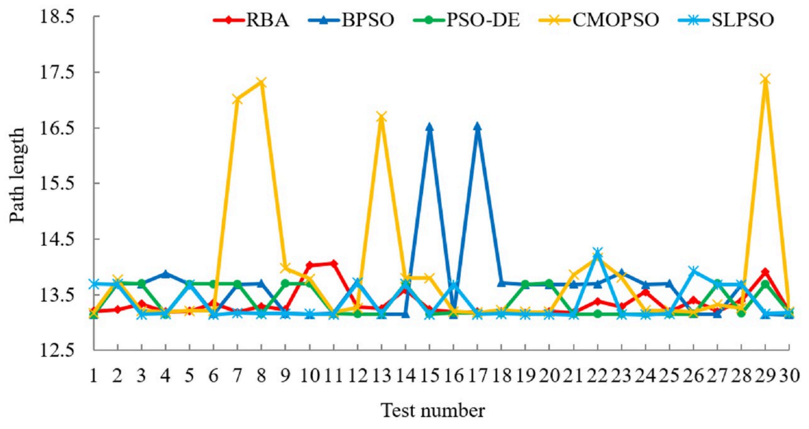
PSO-DE, CMOPSO and SLPSO, the performance is significantly degraded compared to the results in test case 1. The success rates of the three algorithms are reduced by 16.67%, 13.33%, and 23.33%, respectively.

The mean and standard deviation of the experimental results in Fig 7 are exhibited in Table 5. It can be seen from Fig 7 and Table 5 that in terms of the number of iterations, in addition to CMOPSO, BPSO, PSO-DE and SLPSO can complete the path planning faster than RBA. Nevertheless, in the light of the success rate and path length results, PSO variants, especially BPSO and CMOPSO, have unsatisfactory global optimization performance and are prone to fall into local optima. Besides, under the same experiment, the path planning results

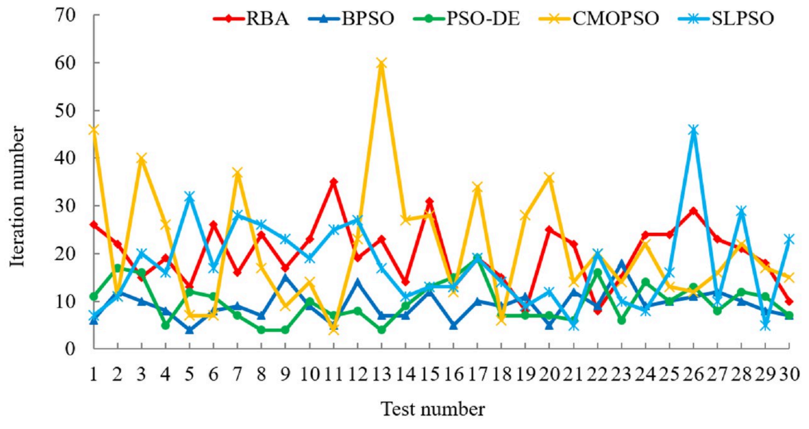
Table 4. Performance comparison between RBA and classical optimization algorithms in test case 2.

Algorithm	RBA	PSO	BA	FA	TLBO	WOA
Average Path Length	13.3436	13.436	13.4159	13.4352	13.2287	13.6005
Standard Deviation of Path Length	0.2469	0.3111	0.3819	0.3281	0.1603	0.5077
Average Iteration Number	19.9	13.97	26.83	14.83	30.5	35.8
Standard Deviation of Iteration Number	6.5303	3.0904	12.0146	6.1928	10.7855	11.1213

https://doi.org/10.1371/journal.pone.0276577.t004



(a)



(b)

Fig 7. Experimental results of five algorithms in test case 2. (a) Path length. (b) Iteration number.

https://doi.org/10.1371/journal.pone.0276577.g007

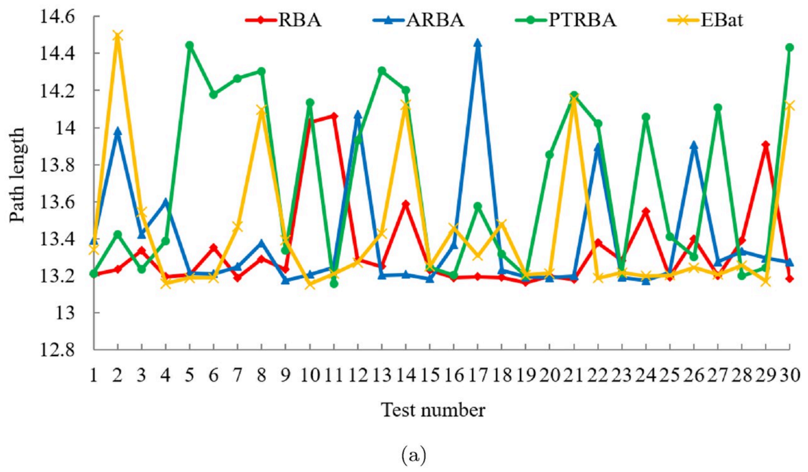
of the five algorithms are depicted in Fig 9(c). It is clear that only RBA and SLPSO accomplish the shortest path, among which, SLPSO requires 14 iterations, while RBA only requires 7 iterations.

Comparison with BA variants. After 120 experiments, we collate the experimental results of RBA and other state-of-the-art BA variants in Fig 8. Meanwhile, the mean and standard deviation of the relevant experimental data are listed in Table 6. In the 30 experiments of each algorithm, RBA realizes the optimal path 27 times, ARBA realizes the optimal path 25 times, PTRBA realizes the optimal path 16 times, and EBat realizes the optimal path 25 times.

Table 5. Performance comparison between RBA and PSO variants in test case 2.

Algorithm	RBA	BPSO	PSO-DE	CMOPSO	SLPSO
Average Path Length	13.3436	13.6786	13.3749	13.9078	13.3586
Standard Deviation of Path Length	0.2469	0.8235	0.2692	1.3125	0.313
Average Iteration Number	19.9	9.3	9.87	21.23	17.7
Standard Deviation of Iteration Number	6.5303	3.153	4.1501	13.0481	9.1544

https://doi.org/10.1371/journal.pone.0276577.t005



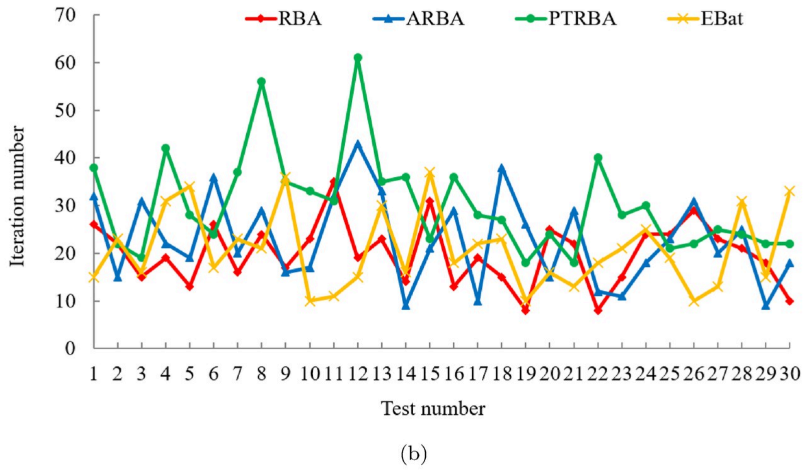


Fig 8. Experimental results of four algorithms in test case 2. (a) Path length. (b) Iteration number.

https://doi.org/10.1371/journal.pone.0276577.g008

Consequently, the success rates of the above algorithms are 90%, 83.33%, 53.33%, and 83.33%, respectively.

On the basis of Table 6, we can get that compared with the novel BA variants, RBA has better path planning effect, not only in the path length but also in the number of iterations, which further proves the superiority of RBA. Furthermore, under the same experiment, the path planning results of the four algorithms are displayed in Fig 9(d). It is evident that only RBA and EBat fulfill the shortest path planning, among which, EBat requires 71 iterations, while RBA only requires 12 iterations.

Table 6. Performance comparison between RBA and BA variants in test case 2.

Algorithm	RBA	ARBA	PTRBA	EBat
Average Path Length	13.3436	13.3984	13.703	13.4314
Standard Deviation of Path Length	0.2469	0.3278	0.4682	0.3699
Average Iteration Number	19.9	22.97	30.17	20.73
Standard Deviation of Iteration Number	6.5303	9.1594	10.2959	8.1238

https://doi.org/10.1371/journal.pone.0276577.t006

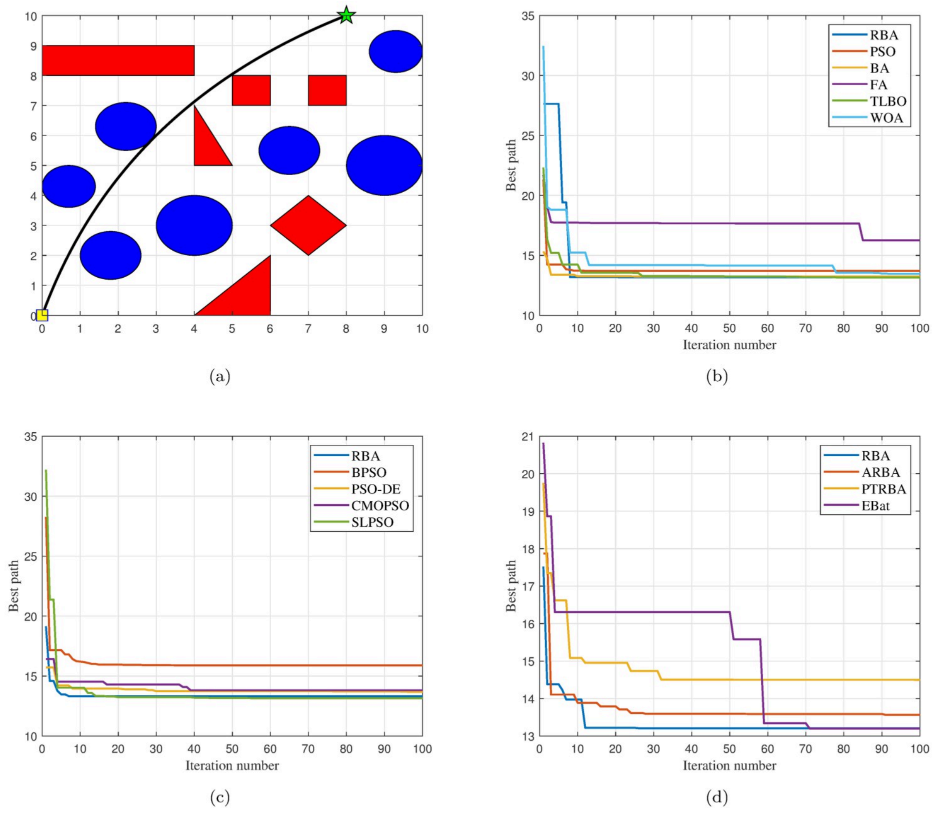


Fig 9. The path planning results in the test case 2 and the number of iterations of all algorithms when the path is implemented. (a) Optimal path. (b) Iteration curves of RBA and classical algorithms. (c) Iteration curves of RBA and PSO variants. (d) Iteration curves of RBA and BA variants.

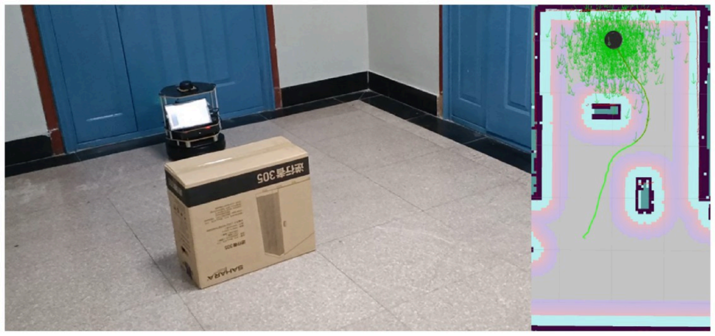
https://doi.org/10.1371/journal.pone.0276577.g009

After the above tests, the validity and superiority of RBA have been verified. With the increase of environment complexity, the path planning effect of RBA is basically not influenced, and the optimal path can be realized in a relatively short time. Simultaneously, the robustness of RBA has also been proven, and it has good adaptability in complex environments.

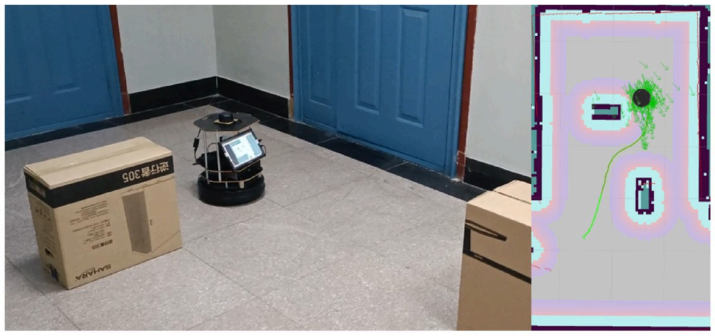
Real-world case

Except for the above simulation experiments, real-world experiments are carried out to verify the real-time performance and effectiveness of our algorithm. The TurtleBot 2 mobile robot equipped with SLAMTEC RPLIDAR A3 is adopted as the experimental platform. In addition, the motion commands of the robot are generated by an IRU-K10 minicomputer with Ubuntu 16.04 and ROS Kinetic installed. The experimental environment map, robot localization and robot path planning are implemented by ROS packages *gmapping*, *amcl* and *move_base*, respectively.

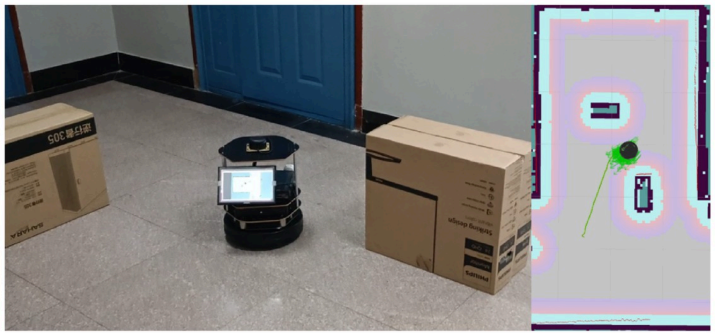
The real-world experimental results are depicted in Fig 10, where the black circle indicates the robot, the green line is the global path of the robot planned by the proposed algorithm RBA, and the green arrow cluster is the particle cloud, representing the robot pose estimated by *amcl*. It is evident from Fig 10 that RBA can plan the global optimal path for the robot in real time. Additionally, along this path, the robot can reach the target point safely and efficiently, thus confirming the feasibility and validity of the proposed algorithm.



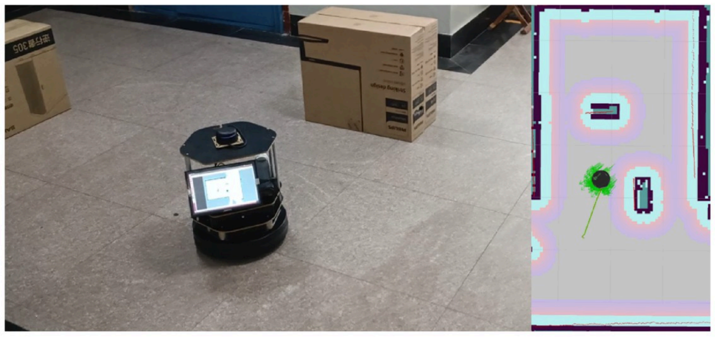
(a)



(b)



(c)



(d)

Fig 10. Real-world robot navigation.

https://doi.org/10.1371/journal.pone.0276577.g010

Conclusions and future work

In this study, a reformative BA is put forth and effectively addresses the mobile robot path planning problem, mainly relying on the following contributions. First, the Doppler effect is applied to frequency update to ameliorate RBA. When the robot is in motion, the Doppler effect can be adaptively compensated to prevent the robot from prematurely converging. Second, the chaotic map and dynamic disturbance coefficient are adopted in the velocity update and position update respectively to weaken the limitation of local optimum and expand the scope of global exploration. Third, Q-learning is integrated into RBA to make reasonable choices for the loudness attenuation coefficient and the pulse emission enhancement coefficient to optimize the algorithm performance and improve the local exploitation capability. Various simulation results verify the effectiveness and superiority of RBA. Compared with other algorithms, RBA has good comprehensive performance in path planning tasks. Furthermore, as the complexity of the environment increases, RBA exhibits superior robustness, and has the merits of fewer iterations, high success rate and high efficiency. Ultimately, real-world experimental results demonstrate that RBA can accomplish the global optimal path planning in real time, and along the optimal path, the robot can reach the target safely and efficiently.

Nevertheless, our work encompasses the following limitations. On the one hand, only the path planning problem in static scenes is considered, while the influence of dynamic obstacles is ignored. On the other hand, only the single-objective optimization problem is solved, while the multi-objective optimization situation is not comprehensively taken into account. Therefore, based on the above defects, in future work, we will comprehensively consider constraints such as path length, collision risk degree and path smoothness to address the path planning issue of mobile robots in static and dynamic environments.

Supporting information

S1 Data. Code and data. This file provides the relevant code for Test case 1 and Test case 2, as well as the experimental data for Figs 2–4 and 6–8.
(ZIP)

Acknowledgments

The authors would like to thank the editor and the anonymous reviewers for their valuable comments.

Author Contributions

Conceptualization: Gongfeng Xin, Lei Shi.

Data curation: Gongfeng Xin.

Formal analysis: Gongfeng Xin.

Funding acquisition: Weigang Pan.

Investigation: Gongfeng Xin.

Methodology: Gongfeng Xin, Lei Shi.

Project administration: Gongfeng Xin, Weigang Pan.

Resources: Gongfeng Xin.

Software: Gongfeng Xin, Lei Shi.

Supervision: Weigang Pan.

Validation: Gongfeng Xin, Guanxu Long, Yiming Li.

Visualization: Jicun Xu.

Writing – original draft: Gongfeng Xin.

Writing – review & editing: Guanxu Long, Yiming Li, Jicun Xu.

References

1. Lozano-Perez T, Wesley MA. An algorithm for planning collision-free paths among polyhedral obstacles. Commun ACM. 1979; 22:560–570. https://doi.org/10.1145/359156.359164
2. Toan TQ, Sorokin AA, Trang VTH. Using modification of visibility-graph in solving the problem of finding shortest path for robot. In: 2017 International Siberian Conference on Control and Communications. IEEE; 2017. p. 1–6.
3. Zhou ZY, Wang JJ, Zhu ZF, Yang DH, Wu J. Tangent navigated robot path planning strategy using particle swarm optimized artificial potential field. Optik. 2018; 158:639–651. https://doi.org/10.1016/j.ijleo.2017.12.169
4. Orozco-Rosas U, Montiel O, Sepulveda R. Mobile robot path planning using membrane evolutionary artificial potential field. Appl Soft Comput. 2019; 77:236–251. https://doi.org/10.1016/j.asoc.2019.01.036
5. Wei K, Ren BY. A method on dynamic path planning for robotic manipulator autonomous obstacle avoidance based on an improved RRT algorithm. Sensors. 2018; 18:571. https://doi.org/10.3390/s18020571
6. Low ES, Ong P, Cheah KC. Solving the optimal path planning of a mobile robot using improved Q-learning. Robot Auton Syst. 2019; 115:143–161. https://doi.org/10.1016/j.robot.2019.02.013
7. Yao Z, Zhang WM, Shi YL, Li MZ, Liang ZS, Li FX, et al. RimJump: Edge-based shortest path planning for a 2D map. Robotica. 2019; 37:641–655. https://doi.org/10.1017/S0263574718001236
8. Yao Z, Zhang WM, Shi YL, Li MZ, Liang ZS, Huang Q. ReinforcedRimJump: Tangent-based shortest-path planning for two-dimensional maps. IEEE Trans Ind Inform. 2020; 16:949–958. https://doi.org/10.1109/TII.2019.2918589
9. Ibraheem GAR, Azar AT, Ibraheem IK, Humaidi AJ. A novel design of a neural network-based fractional PID controller for mobile robots using hybridized fruit fly and particle swarm optimization. Complexity. 2020; 2020:3067024. https://doi.org/10.1155/2020/3067024
10. Blum C, Li XD. Swarm Intelligence in Optimization. In: Swarm Intelligence. Springer; 2008. p. 43–85.
11. Kennedy J. The particle swarm: social adaptation of knowledge. In: 1997 International Conference on Evolutionary Computation. IEEE; 1997. p. 303–308.
12. Chen K, Zhou FY, Yuan XF. Hybrid particle swarm optimization with spiral-shaped mechanism for feature selection. Expert Syst Appl. 2019; 128:140–156. https://doi.org/10.1016/j.eswa.2019.03.039
13. Rao RV, Savsani VJ, Vakharia DP. Teaching-learning-based optimization: An optimization method for continuous non-linear large scale problems. Inform Sciences. 2012; 183:1–15. https://doi.org/10.1016/j.ins.2011.08.006
14. Wang BC, Li HX, Feng Y. An improved teaching-learning-based optimization for constrained evolutionary optimization. Inform Sciences. 2018; 456:131–144. https://doi.org/10.1016/j.ins.2018.04.083
15. Karaboga D, Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Global Optim. 2007; 39:459–471. https://doi.org/10.1007/s10898-007-9149-x
16. Contreras-Cruz MA, Ayala-Ramirez V, Hernandez-Belmonte UH. Mobile robot path planning using artificial bee colony and evolutionary programming. Appl Soft Comput. 2015; 30:319–328. https://doi.org/10.1016/j.asoc.2015.01.067
17. Xue Y, Jiang JM, Zhao BP, Ma TH. A self-adaptive artificial bee colony algorithm based on global best for global optimization. Soft Comput. 2018; 22:2935–2952. https://doi.org/10.1007/s00500-017-2547-1
18. Dorigo M, Caro GD. Ant colony optimization: a new meta-heuristic. In: 1999 International Conference on Evolutionary Computation. IEEE; 1999. p. 1470–1477.
19. Ibraheem IK, Ajeil FH. Path planning of an autonomous mobile robot using swarm based optimization techniques. Al-Khwarizmi Engineering Journal. 2016; 12:12–25. https://doi.org/10.22153/kej.2016.08.002

20. Rajput U, Kumari M. Mobile robot path planning with modified ant colony optimization. Int J Bio-inspir Com. 2017; 9:106. https://doi.org/10.1504/IJBIC.2017.083133

21. Deng W, Xu JJ, Zhao HM. An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem. IEEE Access. 2019; 7:20281–20292. https://doi.org/10.1109/ACCESS.2019.2897580

22. Wang HJ, Guo F, Yao HF, He SS, Xu X. Collision avoidance planning method of USV based on improved ant colony optimization algorithm. IEEE Access. 2019; 7:52964–52975. https://doi.org/10.1109/ACCESS.2019.2907783

23. Ajeil FH, Ibraheem IK, Azar AT, Humaidi AJ. Grid-based mobile robot path planning using aging-based ant colony optimization algorithm in static and dynamic environments. Sensors. 2020; 20:1880. https://doi.org/10.3390/s20071880 PMID: 32231091

24. Sadhu AK, Konar A, Bhattacharjee T, Das S. Synergism of firefly algorithm and Q-learning for robot arm path planning. Swarm Evol Comput. 2018; 43:50–68. https://doi.org/10.1016/j.swevo.2018.03.014

25. Patle BK, Pandey A, Jagadeesh A, Parhi DR. Path planning in uncertain environment by using firefly algorithm. Def Technol. 2018; 14:691–701. https://doi.org/10.1016/j.dt.2018.06.004

26. Yang XS. A new metaheuristic bat-inspired algorithm. In: Nature Inspired Cooperative Strategies for Optimization. Springer; 2010. p. 65–74.

27. Meng XB, Gao XZ, Liu Y, Zhang HZ. A novel bat algorithm with habitat selection and Doppler effect in echoes for optimization. Expert Syst Appl. 2015; 42:6350–6364. https://doi.org/10.1016/j.eswa.2015.04.026

28. Ghanem WAHM, Jantan A. An enhanced bat algorithm with mutation operator for numerical optimization problems. Neural Comput Appl. 2019; 31:617–651. https://doi.org/10.1007/s00521-017-3021-9

29. Yildizdan G, Baykan OK. A novel modified bat algorithm hybridizing by differential evolution algorithm. Expert Syst Appl. 2020; 141:112949. https://doi.org/10.1016/j.eswa.2019.112949

30. Ajeil FH, Ibraheem IK, Azar AT, Humaidi AJ. Autonomous navigation and obstacle avoidance of an omnidirectional mobile robot using swarm optimization and sensors deployment. Int J Adv Robot Syst. 2020; 17:1–15. https://doi.org/10.1177/1729881420929498

31. Ajeil FH, Ibraheem IK, Sahib MA, Humaidi AJ. Multi-objective path planning of an autonomous mobile robot using hybrid PSO-MFB optimization algorithm. Appl Soft Comput. 2020; 89:106076. https://doi.org/10.1016/j.asoc.2020.106076

32. Mirjalili S, Lewis A. The whale optimization algorithm. Adv Eng Softw. 2016; 95:51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

33. Mo HW, Xu LF. Research of biogeography particle swarm optimization for robot path planning. Neurocomputing. 2015; 148:91–99. https://doi.org/10.1016/j.neucom.2012.07.060

34. Tang BW, Zhu ZX, Luo JJ. Hybridizing particle swarm optimization and differential evolution for the mobile robot global path planning. Int J Adv Robot Syst. 2016; 13:86. https://doi.org/10.5772/63812

35. Mac TT, Copot C, Tran DT, Keyser RD. A hierarchical global path planning approach for mobile robots based on multi-objective particle swarm optimization. Appl Soft Comput. 2017; 59:68–76. https://doi.org/10.1016/j.asoc.2017.05.012

36. Li GS, Chou WS. Path planning for mobile robot using self-adaptive learning particle swarm optimization. Sci China Inform Sci. 2018; 61:052204. https://doi.org/10.1007/s11432-016-9115-2

37. Liu JS, Ji HY, Li Y. Robot path planning based on improved bat algorithm and cubic spline interpolation. Acta Automatica Sinica. 2021; 47:1710–1719.

38. Tang HW, Sun W, Yu HS, Lin AP, Xue M. A multirobot target searching method based on bat algorithm in unknown environments. Expert Syst Appl. 2020; 141:112945. https://doi.org/10.1016/j.eswa.2019.112945