A Random Opposition-Based Sparrow Search Algorithm for Path Planning Problem

Guangjian Zhang and Enhao Zhang(✉)

Chongqing University of Technology, Chongqing, China
52192325122@2019.cqut.edu.cn

Abstract. In this paper, an improved intelligence algorithm is proposed for path planning problem. The algorithm is based on Sparrow Search Algorithm and is combined with Random Opposition-based Learning and linear decreasing strategy, named ROSSA. The mobile robot path planning problem can be mathematically transformed into an optimization problem, which can be solved by intelligent optimization algorithms. With this consideration, an SSA-based optimization algorithm is proposed. Random opposition-based learning increases the diversity of the population and enhances the exploration ability of the algorithm; the linear decreasing strategy balances the ability of the algorithm to explore globally and exploit locally by adjusting the algorithm parameters. Meanwhile, the Bezier curve satisfies the requirement of path smoothness for the robot path planning problem. The superiority of the proposed algorithm is verified by conducting experiments with three standard algorithms for 11 benchmark test functions, and some comparison experiments on the path planning problem with PSO and SSA to confirm that the proposed algorithm can find a safe and optimal path in the mobile robot path planning problem.

Keywords: Path planning · Sparrow search algorithm · Opposition-based learning · Bezier curve

1 Introduction

Robot path planning is a very important part in the field of robotics, because it gives robots the ability to move, so that it can handle a variety of tasks that need to move between two points [1].

Given the start and goal position for robot in a 2D environment with static obstacles, the goal of path planning is to search for an optimal or suboptimal collision-free path so that robots can move from the start point to the target point without collision with obstacles [2]. Based on the mastery of the environment, path planning can be divided into global path planning and local path planning.

There has been lots of research on motion planning since the pioneering work presented by N. J. Nilsson in late 1960 s [3]. Thus far, various motion planning

L. Fang et al. (Eds.): CICAI 2021, LNAI 13070, pp. 408–418, 2021.
https://doi.org/10.1007/978-3-030-93049-3_34

algorithms have been presented such as Probabilistic Roadmaps [4,5], Rapidly Exploring Random Trees [6,7], and Potential Fields [8,9], etc. These algorithms can be divided into deterministic and undeterministic algorithms. Deterministic algorithms must find the optimal solution when the problem has an optimal solution, otherwise they return information that there is no optimal solution. However, as the size of the problem becomes more complex, the complexity of modeling the problem and the amount of computation required by the algorithm grows exponentially. Besides, since practical engineering problems usually have many locally optimal solutions, it is difficult for these deterministic algorithms to cope with increasingly difficult problems. Unlike deterministic algorithms, meta-heuristic algorithms can find an approximate solution in case the exact solution cannot be found. This can significantly reduce the amount of computation. Also, meta-heuristic algorithms introduce stochasticity, which gives it the ability to get rid of the local optimum problem. These advantages provide important implications for metaheuristic algorithms to solve global optimization problems. In the past decades, researchers have proposed various Swarm intelligence algorithms, including: Particle Swarm Optimization Algorithm [10], Krill Herd Optimization Algorithm [11], Beetle Antenna Search Algorithm [12], etc. The sparrow search algorithm [13] is a novel metaheuristic optimization algorithm recently proposed with faster convergence, fewer control parameters, and simpler computation, but like other swarm intelligence algorithms, it tends to converge early when solving complex optimization problems, thus falling into the local optima.

In this paper, a novel SSA-based path planning algorithm is proposed. The algorithm incorporates random opposition-based learning strategy and linear decreasing mechanism and is utilized to optimize the control points of Bezier curve, which is used to generate an optimal feasible path. The Bezier curve requires only a few control points to generate a smooth curve, which makes the dimension of the path planning problem not increase exponentially with the complexity of the environment and greatly reduces the complexity of the path planning problem. The superiority of the proposed algorithm is verified by benchmark function experiments, and the smooth optimal path of the robot is designed more stably in the contrast experiments of the path planning problem.

The remaining of the article is arranged as follows: Sect. 2 explains basic SSA algorithm. Section 3 presents the proposed algorithm ROSSA. The description of robot path planning problem and Bezier curve are discussed and contrast experiments with PSO and SSA are conducted in Sect. 4. Finally, Sect. 5 gives the conclusion.

2 Sparrow Search Algorithm(SSA)

SSA is a novel swarm intelligence-based optimization algorithm inspired by the foraging and anti-predatory behaviors of a sparrow population. It has three phases: producer phase, scrounger phase and scouter phase. The key steps of the SSA algorithm are following:

Initialization: First of all, SSA initializes all the parameters and random population of sparrow as follows:

$$X_{i,j} = rand \times (UB_j - LB_j) \tag{1}$$

where $i = 1, 2, ..., pop$, $j = 1, 2, ..., dim$. LB_j and UB_j are lower and upper bounds of search spaceseparately. $rand \in (0, 1)$ is a random number.

Producer phase: After initialization, sparrows in the top 10%–20% fitness values (producers) start to search for a better solution in the search space. In this phase, the location of the sparrow is updated by

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^{t} \cdot \exp\left(\frac{-i}{\alpha \cdot T}\right) & ifR_2 \leq ST \\ X_{i,j}^{t} + Q \cdot L & ifR_2 \geq ST \end{cases} \tag{2}$$

where t represents the current iteration. α is a random number in the range $(0, 1)$. T is the max iteration. Q obeys normal distribution. $R_2 \in [0, 1]$, $ST \in [0.5, 1]$ are the alarm value and the safety threshold respectively.

Scrounger phase: The rest of population are called scroungers. The movement of scrounger individuals can be defined as:

$$X_{i,j}^{t+1} = \begin{cases} Q \cdot \exp\left(\frac{x_{worst}^{t} - X_{i,j}^{t}}{i^2}\right) & i > n/2 \\ X_p^t + \left|X_{ij}^t - X_p^t\right| \cdot A^+ \cdot L & otherwise \end{cases} \tag{3}$$

where X_p is the best position occupied by the producer. X_{worst} denotes the current global worst location. A represents a matrix of $1 \times d$ for which each element inside is randomly assigned 1 or –1, and $A^+ = A^T \left(\mathrm{A}A^T \right)^{-1}$.

Scout phase: Randomly select 10%–20% of population as scout. The update formula of scout is described as follows:

$$X_{i,j}^{t+1} = \begin{cases} X_{best}^{t} + \beta \cdot \left|X_{i,j}^{t} - X_{best}^{t}\right| & f_i > f_g \\ X_{i,j}^{t} + K \cdot \left(\frac{|x_{l,j}^{t} - X_{worst}^{t}}{(f_i - f_{worst}) + \varepsilon}\right) & f_i = f_g \end{cases} \tag{4}$$

where X_{best} is the current global optimal location. b is the step control parameter that obeys a normal distribution. $K \in [-1, 1]$ is a random number. f_i represents the fitness value of sparrow i. f_g and f_w denote the current global best and worst fitness values, respectively. ε is a constant used to avoid the denominator being 0.

3 Improvement

3.1 Opposition-based Learning(OBL)

OBL was first proposed by Tizhoosh [14], and a large amount of variants of opposition-based learning were proposed, such as quasi-opposition [15], quasi-reflection [16], comprehensive opposition [17], etc. Studies showed that considering both random outcomes and their opposite results is more advantageous than considering only random results [18]. The concept of opposition-based learning is based on opposite numbers. It is expressed as follows: Let $x \in [a, b]$ be a real number. Then its opposite number, $\breve{x}$, is given by following equation:

$$\breve{x} = a + b - x \tag{5}$$

In higher dimensional space, the extended definition is defined as follows:

Let $x\,(x_1, \ldots, x_d)$ be a point in d-dimensional space and $x_i \in [a_i, b_i]$, $i = 1, 2, \ldots, d$. The opposite point of x, $\breve{x}\,(\breve{x}_1, \ldots, \breve{x}_d)$, can be expressed as:

$$\breve{x}_i = a_i + b_i - x_i \tag{6}$$

Different from basic opposite point, this paper uses a variant strategy called random opposite point [19], which is defined by:

$$\breve{x}_i = \begin{cases} a_i + b_i - x_i & rand \geq R \\ x_i & otherwise \end{cases} \tag{7}$$

It is reported that by this reverse strategy there are more possible positions than the base reverse strategy, further increasing the diversity of the population.

3.2 Random Opposition-based Sparrow Search Algorithm(ROSSA)

SSA has the disadvantage that when the search is close to the global optimum, the population diversity decreases and it is easy to fall into the local optimum solution [20]. This paper uses random opposition-based strategy to improve SSA. First, a random OBL strategy is used to generate the opposite initial solution when initializing the population, and an elite strategy is used to select better individuals from the initial population and the opposite initial population to form the final initial population. This gives the algorithm an advantage at the beginning. Meanwhile, the producers in SSA searches the whole search space, and random opposition-based strategy can effectively increase the population diversity and optimize the global search ability.

Both producers and scouters in SSA can enhance the global exploration ability of the algorithm, but their proportion is fixed, which does not balance well between global exploration and local exploitation in the first and second stages of SSA algorithm. Therefore, this paper adopts a linear decreasing strategy to control the number of both producers and scouters, which is beneficial to the convergence of the algorithm. The decreasing formula is as follows.

$$p = p_{\max} - (p_{\max} - p_{\min}) \cdot \frac{t}{T} \tag{8}$$

where p is the proportion of producers and scouters, p_{max} and p_{min} denote the maximum and minimum number of p. In this paper, the maximum and minimum values of both are taken as 0.4 and 0.1, respectively.

The main flow of ROSSA is shown in Algorithm 1.

Algorithm 1: Framework of ROSSA

```
input  : $T$: the maximum iterations,
         $pop$: the size of population,
         $num_p$: the number of producers,
         $num_s$: the number of scouters,
         $ST$: the threshold of alert value
output: $X_{best}$, $f_{best}$
Initialize the population and opposition population;
Sort the population by fitness and retain the pop individuals with better
fitness values;
while $t < T$ do
    Calculate $num_p$ with equation 8;
    for $i = 1 : num_p$ do
        update the producers' location with Equation 7;
    end
    for $i = (num_p + 1) : pop$ do
        update the scroungers' location with Equation 3;
    end
    Calculate $num_s$ with equation 8;
    for $i = 1 : num_s$ do
        update the scouters' location with Equation 4;
    end
    t=t+1;
end
return $X_{best}$, $f_{best}$;
```

3.3 Benchmark Test

To verify the advancedness of the proposed algorithm, PSO, KH, SSA and ROSSA are used to solve these test functions, which are shown in Table 1. In the tests, the population size is set to 30, the total number of iterations is set to 500, and the dimension of each test function are 30. The other properties of the functions are shown in the following table. 30 simulation experiments were conducted for each test function separately, and the mean and variance obtained from 30 experiments were counted as shown in Table 2.

Among them, F1–F6 are unimodal test functions, which are mainly used to test the exploitation ability of the algorithm. The results of these 6 unimodal functions show that ROSSA has the best effect of finding the best solution for unimodal functions, and can obtain the global optimal solution to all of these 6 functions, and the stability of ROSSA is better than other 3 algorithms.

F7–F11 are multimodal test functions, which have multiple local optimal solutions, and the intelligent optimization algorithm is easy to fall into the local optimum when solving, so the multimodal test functions are mainly used to test the exploration ability of the algorithm. In solving F7, all four algorithms have unsatisfactory results for this function on average, but SSA and ROSSA can

Table 1. benchmark functions

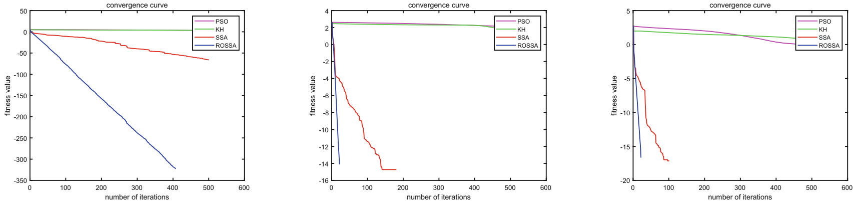
Unimodal functions	Range	fmin
Sphere	[–100,100]	0
Schwefel 2.22	[–10,10]	0
Schwefel 1.2	[–100,100]	0
Schwefel 2.21	[–100,100]	0
Rosenbrock	[–30,30]	0
Step	[–100,100]	0
Multimodal functions	Range	fmin
Schwefel 2.26	[–500,500]	0
Rastrigin	[–5.12,5.12]	0
Ackley	[–32,32]	0
Griewank	[–600,600]	0
Penalty	[–50,50]	0

Table 2. Test results

Function	Values	PSO	KH	SSA	ROSSA
Sphere	Ave	1.139900e+03	1.834809e+03	6.655620e-66	0
	Best	7.770424e+01	1.366952e+03	0	0
	Var	6.449311e+06	7.730924e+04	7.504269e-130	0
Schwefel 2.22	Ave	6.683219e+01	1.610389e+10	4.019652e-38	0
	Best	3.229506e+01	7.338242e+06	0	0
	Var	4.282651e+02	2.609927e+21	4.826645e-74	0
Schwefel 1.2	Ave	1.989746e+04	5.264767e+03	3.469262e-41	0
	Best	9.569614e+03	2.534085e+03	0	0
	Var	2.610611e+07	2.632828e+06	3.610733e-80	0
Schwefel 2.21	Ave	2.519220e+01	1.506538e+00	2.556319e-27	0
	Best	1.836057e+01	1.066817e+00	0	0
	Var	1.893842e+01	8.679998e-02	1.960410e-52	0
Rosenbrock	Ave	9.918774e+03	5.557089e+05	1.722730e-05	1.349896e-09
	Best	2.237782e+02	2.341405e+05	0	0
	Var	6.563432e+08	6.284255e+10	4.127078e-09	5.466655e-17
Step	Ave	1.876022e+00	4.683979e+02	7.875731e-08	2.208193e-11
	Best	3.994384e-01	3.481912e+02	0	0
	Var	4.533699e+00	3.435012e+03	3.706890e-14	1.462835e-20
Schwefel 2.26	Ave	2.143348e+03	4.225624e+03	3.038711e+03	1.908453e+03
	Best	9.521574e+02	2.625807e+03	3.818270e-04	3.818270e-04
	Var	3.464213e+05	6.830527e+05	9.286784e+06	3.112237e+06
Rastrigin	Ave	1.185528e+02	3.808714e+01	0	0
	Best	6.077383e+01	2.178187e+01	0	0
	Var	1.086724e+03	1.342092e+02	0	0
Ackley	Ave	1.406729e+00	1.891097e+00	8.881784e-16	8.881784e-16
	Best	2.128831e-01	1.439818e+00	8.881784e-16	8.881784e-16
	Var	5.047473e-01	4.622237e-02	0	0
Griewank	Ave	8.646091e-01	5.004274e+00	0	0
	Best	3.867774e-01	3.705478e+00	0	0
	Var	4.332038e-02	5.003298e-01	0	0
Penalty	Ave	5.049397e+00	5.059365e+04	3.214013e-09	6.214156e-10
	Best	1.200169e+00	2.353369e+03	1.570545e-32	1.570545e-32
	Var	7.984803e+00	3.192327e+09	7.707496e-17	1.124535e-17

explore better positions; in solving F8, F9 and F10, SSA and ROSSA outperform the other two algorithms; in solving F11, both SSA and ROSSA have the ability to find excellent solutions, but ROSSA has a slight advantage. In summary, ROSSA performs better than the other three algorithms in the benchmark function experiments.

Figure 1 shows the convergence curves of the partial functions of each algorithm. The horizontal axis represents the number of update generations and the vertical axis represents the log of the fitness value. It can be seen that, ROSSA has better convergence speed, accuracy and stability.



(a) Convergence graph of F1.
(b) Convergence graph of F8.
(c) Convergence graph of F11.

Fig. 1. Convergence curves of partial functions: (a) F1, (b) F8, (c) F11.

4 ROSSA for Path Planning Problems

4.1 Problem Description

In this paper, ROSSA is used to solve the robot path planning problem. The target environment is a two-dimensional plane with static obstacles. Each individual in the algorithm denotes a path, represented by N control points as $p[p_1, p_2, \ldots, p_N]$, where p1 is the starting point and p_N is the end point. In the implementation, the SSA individuals are represented as $[x_2, y_2, x_3, y_3, \ldots, x_{(N-1)}, y_{(N-1)}]$ for coding. In the actual environment, obstacles have various shapes. In this paper, for the simplification of the environment model, the circumcircle of the obstacle is used to simplify modeling.

4.2 Bezier Curve

Bezier curve was first proposed by engineer P.E. Bezier [21] and is widely used in practices such as computer graphics and mechanical design [22]. Bezier curve is generated by a series of control points and these points are not on the curve except for the start and end points. Given a set of control points $P_0, ..., P_n$, the corresponding Bezier curve can be expressed as

$$P(t) = \sum_{i=0}^{n} P_i B_{i,n}(t), \quad t \in [0, 1] \tag{9}$$

where t is the normalized time variable, $B_{i,n}$ is the Bernstein basis polynomials, which represents the base function in the expression of a Bezier curve:

$$B_{i,n}(t) = C_n^i t^i (1-t)^{n-i} \quad i = 0, 1, \ldots n \tag{10}$$

In this way, a smooth curve can be created with only a small number of control points.

4.3 Fitness Function

The purpose of this paper is to find an optimal path for the robot that satisfies the constraints, where the constraints include (1) feasibility, (2) optimality, and (3) safety.

1. Feasibility

   Feasibility is the most important goal of path planning. If the path collides with an obstacle, the fitness should be large, which is set here to 10000:

$$f_{feasible} = \begin{cases} 0 & if \quad feasible \\ 10000 & otherwise \end{cases} \tag{11}$$

2. Shortest distance

   The second target is to minimize the length of the solution generated by the algorithm. For simplicity, we choose 100 points on the path and calculate the Euclidean distance between two adjacent points:

$$f_{length} = \sum_{i=1}^{n-1} \|p_{i+1} - p_i\| \tag{12}$$

3. Safety
   An excellent path should be as far away from obstacles as possible. If the distance between the path and the obstacle is less than the safe distance, d_{safe}, it will be penalized:

$$f_{safe}(o_j) = \begin{cases} \left(1 - \frac{d_{\min}(o_j)}{d_{safe}(o_j)}\right)^2 & if \quad d_{\min}(o_j) \le d_{safe}(o_j) \\ 0 & otherwise \end{cases} \tag{13}$$

$$f_{risk} = \max(f_{safe}) \tag{14}$$

   where $d_{min}(o_j)$ means the minimum distance of the path from the obstacle j. $d_{safe}(o_j)$ can be expressed as follow:

$$d_{safe}(o_j) = k r_{o_j} \tag{15}$$

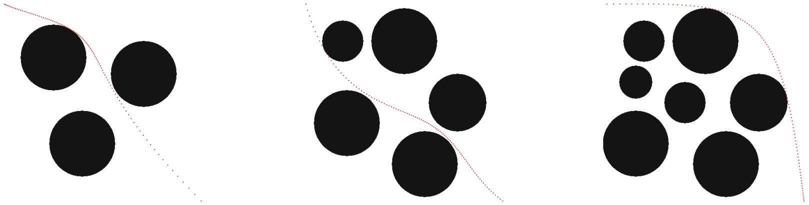
   where k indicates the scale factor and r_{o_j} denotes the radius of obstacle j.

Considering the above factors, the fitness function of the robot path planning problem can be expressed as:

$$f = f_{feasible} + w_1 * f_{length} + w_2 * f_{risk} \tag{16}$$

4.4 Comparision

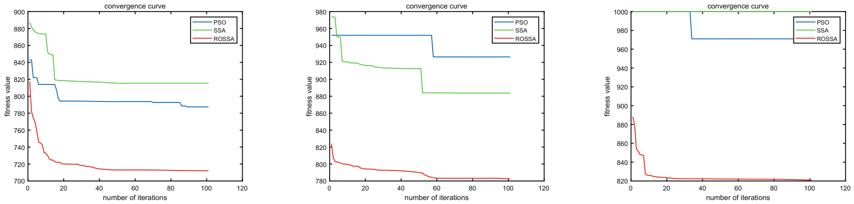
The parameters of the path planning model are set as follows: the map is 500 × 500, as shown in Fig. 2. The number of control points is 5, the robot moves from (10,10) to (490,490).



(a) Environment 1. (b) Environment 2. (c) Environment 3.

Fig. 2. Environments.

The objective function of this path planning model is solved using PSO, SSA and ROSSA respectively to obtain the desired paths. The population size is set to 30, each individual is a path, the maximum number of iterations is 500, and 30 simulation experiments are conducted. Figure 3 shows the convergence curves of the three algorithms and related data are shown in Table 3.



(a) (b) (c)

Fig. 3. Convergence curve of each environment.

The comparison in Fig. 3 and Table 3 show that the ROSSA algorithm outperforms the PSO and SSA algorithms for the path planning problem. As can be seen from Fig. 3, due to the opposition-based initialization of ROSSA, the initial solution of ROSSA is in a more optimal position. Meanwhile, the rapid convergence to the better position and the continuous approximation to the optimum can stabilize the convergence to the optimal value. Table 3 shows that the average and minimum fitness values obtained by the ROSSA algorithm are lower than those of the PSO and SSA algorithms, and that it is able to solve the path planning problem stably, resulting in a safe and feasible trajectory that is optimal and satisfies the constraints.

Table 3. Performance comparison of three algorithms

(a) Environment 1.

Algorithm	Ave	Best	Var	Success rate
PSO	7.679612e+02	7.160134e+02	7.450565e+03	90%
SSA	8.636940e+02	7.040057e+02	2.140079e+04	50%
ROSSA	7.145130e+02	7.023720e+02	9.443526e+01	100%

(b) Environment 2.

Algorithm	Ave	Best	Var	Success rate
PSO	8.929376e+02	7.090549e+02	1.959787e+04	40%
SSA	9.395579e+02	6.977466e+02	1.623664e+04	30%
ROSSA	8.003072e+02	7.034846e+02	2.640106e+03	100%

(c) Environment 3.

Algorithm	Ave	Best	Var	Success rate
PSO	INF	INF	0	0%
SSA	9.714638e+02	9.146378e+02	8.143158e+03	10%
ROSSA	8.216849e+02	8.152322e+02	5.115674e+01	100%

5 Conclusions

In this paper, an improved SSA is used to solve the path planning problem. The random OBL strategy and a linear decreasing strategy are introduced into the basic SSA. These strategies are used to increase the population diversity, balance the local exploitation and global exploration ability of the algorithm, and avoid the algorithm from falling into local optimum. The results of benchmark function test show that the proposed algorithm has a significant improvement in the performance of convergence speed, accuracy and stability. The path planning simulation results show that the path planning based on ROSSA can effectively find the optimal path and steadily plan a feasible and efficient path.

References

1. Reif, J.H.: Complexity of the mover's problem and generalizations. In: 20th Annual Symposium on Foundations of Computer Science (SFCS 1979), pp. 421–427. IEEE (1979)
2. Li, G., Chou, W.: Path planning for mobile robot using self-adaptive learning particle swarm optimization. Sci. China Inf. Sci. **61**(5), 1–18 (2017). https://doi.org/10.1007/s11432-016-9115-2
3. Nilsson, N.J.: A mobile automaton: An application of artificial intelligence techniques. Technical Report, Sri International Menlo Park Ca Artificial Intelligence Center (1969)
4. Kavraki, L.E., Svestka, P., Latombe, J.C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Autom. **12**(4), 566–580 (1996)

5. Yan, F., Liu, Y.S., Xiao, J.Z.: Path planning in complex 3d environments using a probabilistic roadmap method. Int. J. Autom. Comput. **10**(6), 525–533 (2013)
6. LaValle, S.M.: Rapidly-exploring random trees: A new tool for path planning (1998)
7. Yuan, C., Liu, G., Zhang, W., Pan, X.: An efficient RRT cache method in dynamic environments for path planning. Robot. Auton. Syst. **131**, 103595 (2020)
8. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. In: Cox I.J., Wilfong G.T. (eds.) Autonomous robot vehicles, pp. 396–404. Springer, New York (1986). https://doi.org/10.1007/978-1-4613-8997-2_29
9. Sabudin, E.N., et al.: Improved potential field method for robot path planning with path pruning. In: Md Zain, Z., et al. (eds.) Proceedings of the 11th National Technical Seminar on Unmanned System Technology 2019. LNEE, vol. 666, pp. 113–127. Springer, Singapore (2021). https://doi.org/10.1007/978-981-15-5281-6_9
10. Li, X., Wu, D., He, J., Bashir, M., Liping, M.: An improved method of particle swarm optimization for path planning of mobile robot. J. Control Sci. Eng. **2020** (2020)
11. Rao, D.C., Kabat, M.R., Das, P.K., Jena, P.K.: Cooperative navigation planning of multiple mobile robots using improved krill herd. Arab. J. Sci. Eng. **43**(12), 7869–7891 (2018)
12. Zhang, B., Duan, Y., Zhang, Y., Wang, Y.: Particle swarm optimization algorithm based on beetle antennae search algorithm to solve path planning problem. In: 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), vol. 1, pp. 1586–1589. IEEE (2020)
13. Xue, J., Shen, B.: A novel swarm intelligence optimization approach: sparrow search algorithm. Syst. Sci. Control Eng. **8**(1), 22–34 (2020)
14. Tizhoosh, H.R.: Opposition-based learning: a new scheme for machine intelligence. In: International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC 2006), vol. 1, pp. 695–701. IEEE (2005)
15. Rahnamayan, S., Tizhoosh, H.R., Salama, M.M.: Quasi-oppositional differential evolution. In: 2007 IEEE Congress on Evolutionary Computation, pp. 2229–2236. IEEE (2007)
16. Ergezer, M., Simon, D., Du, D.: Oppositional biogeography-based optimization. In: 2009 IEEE International Conference on Systems, Man and Cybernetics, pp. 1009–1014. IEEE (2009)
17. Seif, Z., Ahmadi, M.B.: An opposition-based algorithm for function optimization. Eng. Appl. Artif. Intell. **37**, 293–306 (2015)
18. Rahnamayan, S., Tizhoosh, H.R., Salama, M.M.: Opposition-based differential evolution. IEEE Trans. Evol. Comput. **12**(1), 64–79 (2008)
19. Bairathi, D., Gopalani, D.: Random-opposition-based learning for computational intelligence. In: Tuba, M., Akashe, S., Joshi, A. (eds.) Information and Communication Technology for Sustainable Development. AISC, vol. 933, pp. 111–120. Springer, Singapore (2020). https://doi.org/10.1007/978-981-13-7166-0_11
20. Qinghua, M., Qiang, Z.: Improved sparrow algorithm combining cauchy mutation and opposition-based learning. J. Front. Comput. Sci. Technol. **15**, 1–12 (2020)
21. Farin, G.: Curves and Surfaces for Computer-Aided Geometric Design: a Practical Guide. Elsevier, Amsterdam (2014)
22. Song, B., Wang, Z., Zou, L.: An improved PSO algorithm for smooth path planning of mobile robots using continuous high-degree Bezier curve. Appl. Soft Comput. **100**, 106960 (2021)