Hindawi
Journal of Sensors
Volume 2022, Article ID 2559955, 16 pages
https://doi.org/10.1155/2022/2559955

Research Article

Robot Path Planning Based on an Improved Salp Swarm Algorithm

Xianbao Cheng,[1] Liucun Zhu,[2] Huihui Lu,[1] Jinzhan Wei,[2] and Ning Wu[2]

[1]*School of Electronics and Information Engineering, Beibu Gulf University, No. 12, Binhai Avenue, Qinzhou, Guangxi, China*
[2]*Advanced Science and Technology Research Institute, Beibu Gulf University, No. 12, Binhai Avenue, Qinzhou, Guangxi, China*

Correspondence should be addressed to Liucun Zhu; lczhu@bbgu.edu.cn and Ning Wu; n.wu@bbgu.edu.cn

Received 2 April 2022; Accepted 6 June 2022; Published 25 June 2022

Academic Editor: Haidong Shao

This paper demonstrates an improved version of the Salp Swarm Algorithm (SSA) to solve the problems of slow convergence and local minima of the original version. In the population initialization of this scheme, ten chaotic searches with dynamic inertia coefficients are introduced to increase the diversity so that the probability of being trapped in local minima is reduced. Genetic algorithms are then applied to improve the global search ability and convergence speed. The experiments with 12 test functions show that the improved version achieves better accuracy and convergence speed over the original SSA. In the test with robot path planning problem, the proposed algorithm shows improved performance in the average number of iterations, path length, and average number of turns by 69.2%, 19.1%, and 43%, respectively, compared with the original SSA.

1. Introduction

Traditional mathematical methods are effective to solve linear and differentiable optimization problems, but for more complicated problems like nondifferentiable functions, more intelligent algorithms are needed. Intelligent algorithms solve optimization problems by imitating natural phenomena, for example, Particle Swarm Optimization (PSO) [1] simulates bird foraging behavior, Gray Wolf Algorithm (GWA) [2] focuses on wolf predation behavior, and artificial bee colony algorithm (ABC) [3] simulates bee foraging behavior.

The Salp Swarm Algorithm (SSA) [4] is a relatively new swarm intelligence algorithm to simulate the foraging behavior of the sea swarm slap. As a new heuristic optimization algorithm, the SSA has the advantages of less parameter requirements and effectiveness for both continuous and discrete problems.

Salp is one kind of Salpidae with a transparent barrel-shaped body similar to jellyfishes with a length of about 1~10 cm. Salps do not behave active locomotion, and the movement is performed by pumping water through the body as propulsion to go forward. The individual salp does not forage very well, and they live in groups to get more feeding. When salps prey in groups, multiple of them are lined up to form a chain structure (salp chain). The first salp in the chain is called the leader, and the rest is called the follower. The leader guides the whole chain, and the followers are mobile following each other [4]. The leader leads the followers to move towards the food source for global search, while the followers go accordingly for a local search. In the SSA, the position update of each follower will only be affected by the position of the previous follower, and the leader's position update is only affected by the food source position. The hierarchical system of the SSA makes the followers cooperate closely with each other to increase the optimization efficiency and reduce the chance of being trapped in a local optimum. The SSA has been widely used in many industrial applications such as variable speed wind turbine [5], industrial design [6], extreme learning machine [7], feature selection [8], neural network [9], image segmentation [10], and biomedicine [11–13].

Path planning is a key topic for the mobility of the robot to navigate the robot automatically from one position to another [14, 15]. Robots often face uncertain and complex

operating environment, and in the meantime time, an efficient path connecting one position to another in this environment is required to be found quickly and accurately [16, 17]. Depending on the operation task of the robot, the optimal path of an environment is generally evaluated based on the shortest distance or time, the minimum energy consumption, or the highest safety rate. The path planning algorithm with superior performance can plan the most efficient path in the uncertain and complex environment, to increase the working efficiency of the robot and reduce the wear and tear of robot. Since one of the key technologies of mobile robot is to look for the optimal path solution for a task, path planning algorithm has become a research hotspot in recent years [18].

Traditional algorithms to solve the path planning problem in known environments include artificial potential field method [19–21], A∗ algorithm [22–24], Dijkstra algorithm [25–27], and rapidly-exploring random tree (RRT) [28–30]. However, the exploration performance of these algorithms is generally poor, and it is difficult for them to find the optimal path in an unknown environment. For this reason, a swarm intelligence algorithm was introduced to make use of the exploration and optimization performance to find the optimal path, such as the Particle Swarm Optimization (PSO) [31], the ant colony (AC) [32], the whale algorithm for UAV path planning [33], and the water wave algorithm for the path planning of underwater vehicles [34]. However, the performance of the SSA for robot path planning has rarely been reported in the literature.

This paper focuses on demonstrating an improved version of the SSA and its application to path planning. The problems with the original SSA such as the locomotion and slow convergence will be overcome. The initialization of the population of the SSA will be improved, and a set of dynamic inertia weight coefficients are defined to maintain the diversity of population. Genetic algorithm (GA) is then used to assist with the globalization of search. In the experiment, the improved SSA method will be tested on the 12 most popular test functions and compared with five other evolution methods. The optimization of robot path planning problem will also be tested with the proposed method, and the comparison with other methods will be shown. This paper is organized in six sections. The second section reviews the related works for SSA and path planning. The third section demonstrates the theory of the proposed algorithm, and the fourth section tests the performance of the improved SSA. The fifth section gives examples of the application of the improved SSA to path planning, and a conclusion is made in Section 6.

2. Related Works

2.1. The Original Salp Swarm Algorithm (SSA). Similar to other swarm intelligence algorithms, the SSA initializes the population in an n-dimensional search space, and the fitness function is regarded as the food source. The salp chain is always trying to approach the food source and finally reach the valuable food source in the search area, which is hopefully the global optimum. The procedure of the SSA can be given as follows:

(1) Initialize the population according to the upper and lower limits of each of the n dimensions, and it can be written as

$$x_m^i = lb(m) + \text{rand}\,(N, D) * (ub(m) - lb(m)), \quad (1)$$

where X_m^i represents the ith salp of m-dimensional space, $i = 1, 2, \cdots, N$ $m = 1, 2, \cdots, D$, N is the total number of salps in the chain, and D is the dimension of the objective function. rand (N, D) represents a random number matrix of N rows and D columns with elements evenly distributed between 0 and 1. $lb(m)$ represents the lower limit, and $ub(m)$ represents the upper limit. The initialization according to Equation (1) will generate an x_D^N matrix such that

$$X = \begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_D^1 \\ x_1^2 & x_2^2 & \cdots & x_D^2 \\ \vdots & \vdots & \cdots & \vdots \\ x_1^N & x_2^N & \cdots & x_D^N \end{bmatrix} \quad (2)$$

(2) The fitness value of each individual is calculated based on the fitness function

(3) Determine the initial location of the selected food source according to the salp with the best fitness value

(4) Identify the leader and followers: the first scalp in the chain is the leader, and the rest are followers

(5) Update the position of the leader according to Equation (3) such that

$$x_i^1 = \begin{cases} F_i + c_1((ub_i - lb_i)c_2 + lb_i), c_3 \geq 0.5, \\ F_i - c_1((ub_i - lb_i)c_2 + lb_i), c_3 \geq 0.5, \end{cases} \quad (3)$$

where x_i^1 is the ith component of the leader and F_i represents ith element of the food source. c_2 and c_3 are random numbers generated in the interval of [0, 1], which represent the length and direction of the movement, respectively. c_1 is a coefficient for adjusting the exploration and exploitation of the salp chain and can be written as $c_1 = 2e^{-(4t/T_{\max})^2}$, where t is the current iteration and $T_{\max}$ is the maximum number of iteration

(6) Update the position of the followers:

Since the salp chain moves in the direction of the food source during foraging, the update of followers depends on the initial speed, iteration, and acceleration like the Newton's law of motion [4], such that

$$x_j^i(t) = \frac{1}{2}\left[x_j^i(t-1) + x_j^{i-1}(t-1)\right], \quad (4)$$

Table 1: The test functions used in this experiment.

No.	Test functions	Dimension	Search range	Minimum value
1	$f_1(x) = \sum_{i=1}^{n} x_i^2$	50	[-100, 100]	0
2	$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	50	[-10, 10]	0
3	$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	50	[-100, 100]	0
4	$f_4(x) = \max \{ x_i \lvert : 1 \le i \le n \}$	50	[-100, 100]	0
5	$f_5(x) = \sum_{i=1}^{n-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right]$	50	[-100, 100]	0
6	$f_6(x) = \sum_{i=1}^{n} (\lvert x_i + 0.5 \rvert)^2$	50	[-100, 100]	0
7	$f_7(x) = \sum_{i=1}^{n} i x_i^4$	50	[-1.28, 1.28]	0
8	$f_8(x) = \sum_{i=1}^{n} \left(-x_i \sin \left(\sqrt{\lvert x_i \rvert} \right) \right)$	50	[-500, 500]	0
9	$f_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10 \cos (2\pi x_i) + 10 \right]^2$	50	[-5.12, 5.12]	0
10	$f_{10}(x) = -20 \exp \left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2} - \exp \left(\frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i) \right) \right) + e + 20$	50	[-32, 32]	0
11	$f_{11}(x) = \frac{1}{4000} \sum_{i=1}^{n} (x_i^2) - \prod_{i=1}^{n} \cos \left(\frac{x_i}{\sqrt{i}} \right) + 1$	50	[-600, 600]	0
12	$f_{12}(x) = 0.1 \left\{ \sin^2(3\pi x_i) + \sum_{i=1}^{n} (x_i - 1)^2 \left[1 + \sin^2(3\pi x_i + 1) \right] + (x_n - 1)^2 \left[1 + \sin^2(2\pi x_n) \right] \right\} + \sum_{i=1}^{n} \mu(x_i, 5, 100, 4)$	50	[-50, 50]	0

Table 2: Test results for the 12 functions listed in Table 1.

Test function	Value types	SSA	WOA	PSO	ABC	CSSA	Proposed SSA
	Minimum	7.27*e*-29	9.27*e*-34	8.23*e*-24	2.39*e*-17	4.26*e*-34	**7.23*e*-38**
f_1	Average	9.45*e*-29	4.26*e*-34	7.92*e*-23	8.93*e*-17	1.28*e*-34	**1.58*e*-38**
	Standard variance	4.66*e*-29	2.78*e*-34	1.25*e*-23	2.45*e*-17	7.21*e*-34	7.23*e*-38
	Minimum	4.98*e*-38	5.44*e*-38	7.21*e*-12	3.63*e*-2	1.77*e*-99	**2.63*e*-148**
f_2	Average	8.64*e*-38	9.48*e*-38	6.74*e*-12	5.67*e*-2	7.62*e*-99	**9.33*e*-148**
	Standard variance	7.58*e*-38	6.55*e*-38	7.25*e*-12	4.23*e*-2	5.38*e*-99	8.47*e*-148
	Minimum	4.29*e*-23	5.11*e*-68	9.27*e*-21	3.47*e*-10	9.26-70	**2.46*e*-115**
f_3	Average	3.24*e*-23	1.27*e*-68	1.31*e*-21	6.26*e*-10	1.86*e*-70	**8.77*e*-115**
	Standard variance	9.26*e*-23	6.67*e*-68	7.25*e*-21	3.77*e*-10	4.61*e*-70	6.55*e*-115
	Minimum	4.21*e*-25	5.58*e*-168	4.25*e*-149	1.70*e*-13	1.41*e*-140	**9.25*e*-169**
f_4	Average	9.93*e*-25	6.45*e*-168	4.33*e*-149	8.67*e*-18	8.92*e*-140	**2.80*e*-169**
	Standard variance	8.45*e*-25	7.22*e*-168	9.75*e*-149	5.49*e*-18	7.20*e*-140	5.39*e*-169
	Minimum	6.72*e*-26	9.76*e*-68	2.43*e*-25	5.12*e*-12	5.80*e*-70	**9.24*e*-112**
f_5	Average	7.28*e*-26	2.73*e*-67	1.73*e*-24	7.48*e*-11	2.62*e*-69	**1.50*e*-111**
	Standard variance	9.30*e*-26	7.77*e*-67	5.76*e*-25	2.66*e*-12	6.67*e*-70	6.90*e*-111
	Minimum	4.37*e*-07	1.57*e*-08	4.96*e*-08	3.24*e*-07	1.46*e*-08	**8.24*e*-11**
f_6	Average	7.61*e*-06	8.26*e*-08	6.44*e*-07	5.67*e*-06	8.43*e*-08	**2.10*e*-10**
	Standard variance	2.78*e*-08	3.21*e*-08	8.33*e*-07	3.22*e*-06	3.49*e*-08	3.94*e*-10
	Minimum	7.22*e*-04	4.37*e*-09	8.66*e*-07	4.33*e*-03	8.25*e*-13	**2.43*e*-18**
f_7	Average	5.46*e*-03	8.26*e*-08	1.45*e*-06	5.55*e*-02	2.82*e*-12	**9.44*e*-13**
	Standard variance	6.28*e*-04	3.15*e*-08	7.34*e*-06	3.36*e*-02	7.88*e*-14	7.10*e*-19
	Minimum	4.52*e*-02	2.76*e*-03	6.99*e*-02	5.44*e*-01	**1.22*e*-05**	3.77*e*-05
f_8	Average	6.25*e*-02	9.74*e*-03	3.22*e*-02	6.12*e*-01	**2.64*e*-05**	8.94*e*-04
	Standard variance	1.13*e*-04	3.44*e*-04	4.54*e*-02	5.33*e*-01	6.74*e*-05	6.60*e*-05
	Minimum	9.95*e*-02	9.33*e*+01	7.26*e*-15	5.66*e*-12	6.53*e*+01	**2.73*e*-15**
f_9	Average	7.24*e*-01	9.66*e*+01	5.31*e*-14	7.81*e*-11	8.98*e*+01	**1.54*e*-14**
	Standard variance	6.58*e*-02	5.43*e*-02	5.76*e*-14	6.54*e*-11	7.11*e*-02	9.12*e*-15
	Minimum	1.56*e*-11	3.28*e*-07	2.46*e*-13	2.69*e*-11	8.93*e*-13	**1.19*e*-15**
f_{10}	Average	4.21*e*-11	8.26*e*-06	7.11*e*-13	4.33*e*-10	1.22*e*-08	**8.74*e*-14**
	Standard variance	8.72*e*-11	4.33*e*-06	2.36*e*-13	5.36*e*-10	5.69*e*-11	4.63*e*-12
	Minimum	5.46*e*-07	5.74*e*-05	1.93*e*-11	5.43*e*-07	4.44*e*-02	**9.56*e*-34**
f_{11}	Average	8.23*e*-07	8.73*e*-05	5.86*e*-10	6.72*e*-07	2.44*e*-01	**1.72*e*-30**
	Standard variance	9.42*e*-07	7.74*e*-05	7.42*e*-10	5.64*e*-06	6.93*e*-03	6.72*e*-32
	Minimum	2.96*e*-04	3.66*e*-02	4.08*e*-08	6.45*e*-08	5.35*e*+01	**3.54*e*-10**
f_{12}	Average	3.37*e*-04	2.98*e*-01	2.76*e*-07	9.45*e*-07	8.70*e*+01	**8.25*e*-10**
	Standard variance	4.29*e*-04	2.16*e*-02	5.45*e*-08	7.73*e*-08	4.24*e*-01	4.77*e*-11

where $x_j^i(t)$ is the value of the jth component of the ith salp in the chain at iteration t

(7) If an updated component moves over the boundary, set the position of the boundary, and then, the food source location is updated according to the optimal salp

(8) If the result meets the accuracy requirements or if the number of iterations is reached, output the current position; otherwise, turn to step 4 for further evolution

2.2. Robot Path Planning. The optimal path designed for a robot is usually calculated considering the constraints like time, distance, and energy consumption. The current most commonly used path planning optimization method is based on artificial intelligence algorithms. Deep learning has also been introduced to path planning, but the adaptability to environmental changes is relatively poor [35].

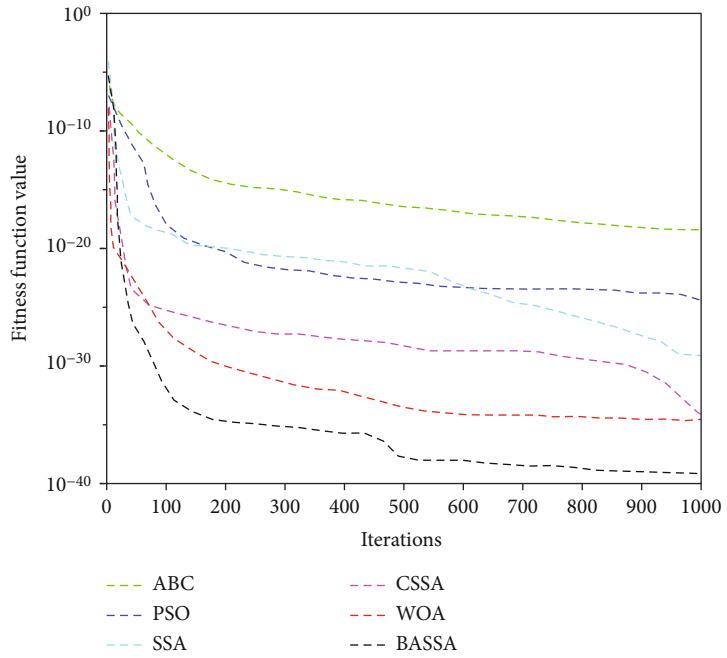

Figure 1: Comparison of the convergence curves of average fitness value for f_1.

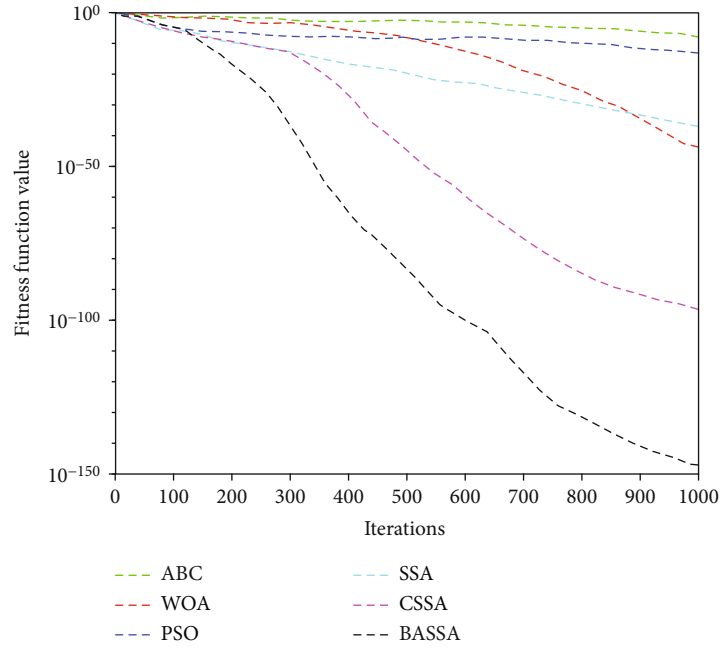

Figure 2: Comparison of the convergence curves of average fitness value for f_2.

To find the optimal path in the obstacle environment for a robot, the model of the mobile environment is required to register. There are currently two types of methods for environment modeling, one is the road sign method and the other is the grid method. The road sign method is to line up the feasible path map by connecting the labeled points and the boundaries of obstacles, while the grid method abstracts the mobile environment of the robot into a grid space and marks all grids that belong to the path. Grid method is more convenient to use and easier to implement, and therefore, it is more popular than the road sign method [36]. In this paper, an optimized grid modeling method is calculated for robot path planning.

In the grid modeling of three-dimensional (3D) mobile environment, a two-dimensional (2D) is marked with grids of equal size to represent the 3D space, and each grid is labeled with 0 or 1 representing without or with an obstacle [37].

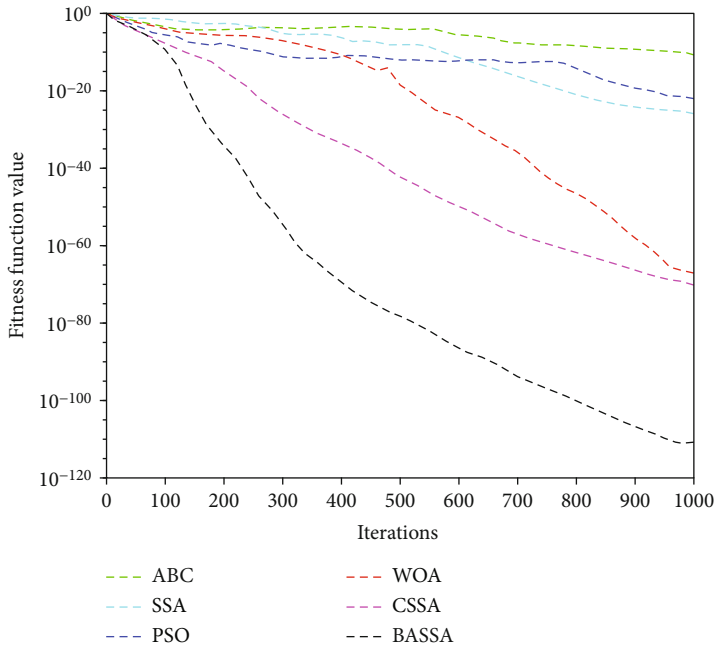

Figure 3: Comparison of the convergence curves of average fitness value for f_3.

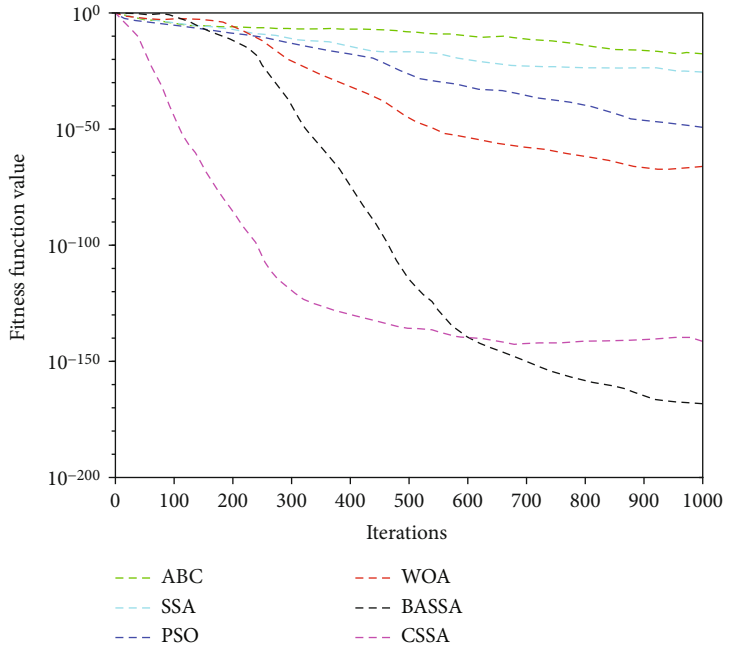

Figure 4: Comparison of the convergence curves of average fitness value for f_4.

The grid modeling is usually used to simulate a limited area, and a coordinate system xOy is established with the lower left corner as the origin and the horizontal and vertical orientations are x axis and y axis, respectively. The step length (l) of the robot represents the length of a single grid in the x or y axis; therefore, the number of grids in each x axis and y axis are $n_x = x_{\max}/l$ and $n_y = y_{\max}/l$, respectively. It is defined that each of the grids in the area is marked with a label starting from the upper left corner in the way of from left to right and from top to bottom, such that $A = \{1, 2, \cdots, N\}$. Then, the relationship between the coordinates and the label number can be given as:

$$\begin{aligned} x_i &= ((i-1) \bmod n_x) + 1, \\ y_i &= n_y - \operatorname{ceil}\left(\frac{i}{n_x}\right) + 1, \end{aligned} \tag{5}$$

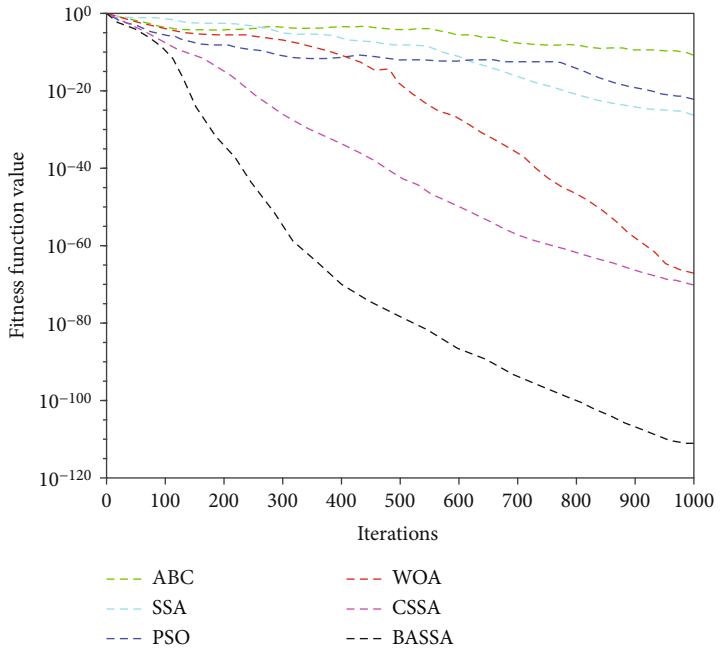

Figure 5: Comparison of the convergence curves of average fitness value for f_5.

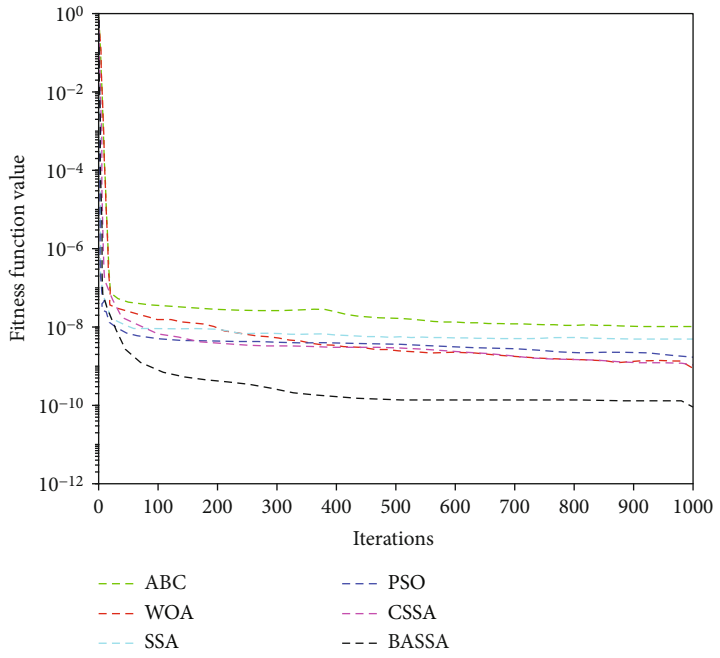

Figure 6: Comparison of the convergence curves of average fitness value for f_6.

where i represents the ith label number, mod is the remainder operation, ceil means rounding operation, n_x, n_y represent the number of grids per row and column, respectively, and x_i,y_i denote the ordinate of the grid center of the ith grid. In this optimization task, it is expected to find the shortest Euclidean distance between the current position (x_1, y_1) and the target position (x_2, y_2), which is regarded as the fitness vale, such that $h(x) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$ [38].

3. The Improved SSA

In SSA, the leader is designed to make global exploration while the follower makes a full local search, and in this way, the chance of falling in a local optimum is greatly reduced. Since the SSA requires fewer parameters than other evolution optimization methods and therefore it is easier to implement, however, like most swarm intelligence

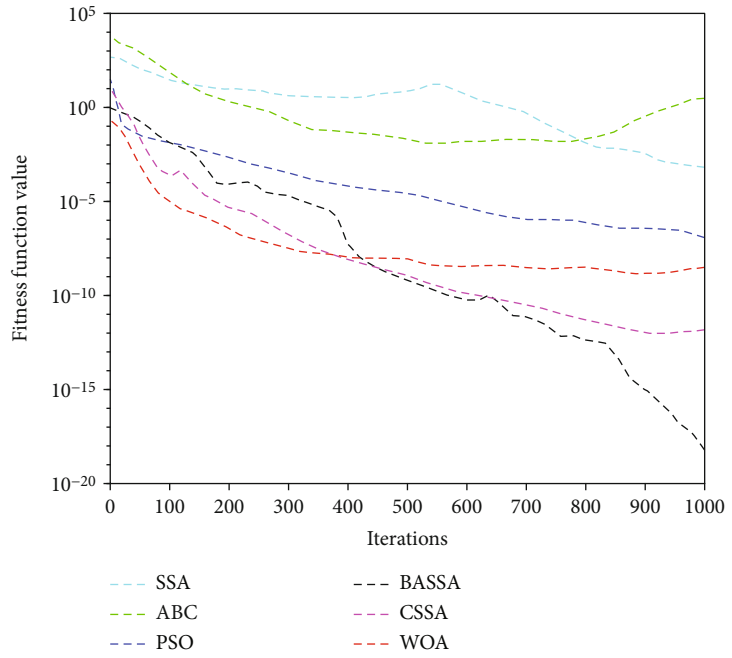

Figure 7: Comparison of the convergence curves of average fitness value for f_7.

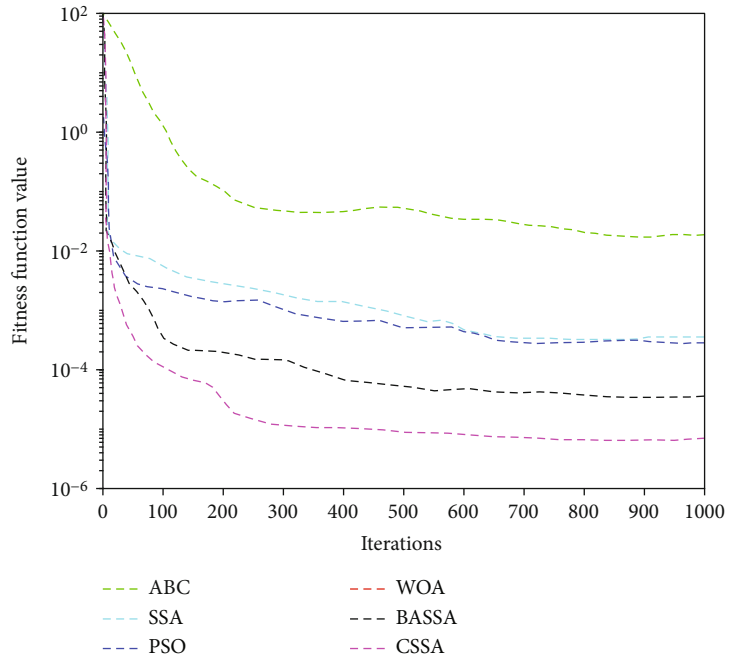

Figure 8: Comparison of the convergence curves of average fitness value for f_8.

algorithms, it is difficult for the SSA to converge at the later stage of optimization.

In the optimization using SSA, the initialization of the population is given within a certain range such that $x_m^i = lb(m) + \text{rand}(N, D) * (ub(m) - lb(m))$. Therefore, if the initial positions of the population are too concentrated, there will be a lack of diversity, resulting in the convergence to a local minimum. While if the initial positions are too scattered, the convergence process will be greatly slowed down. Besides, in the salp chain, the position of the individual is updated from one to the next along the chain; in some special cases, the value cannot be passed on, or in some cases,

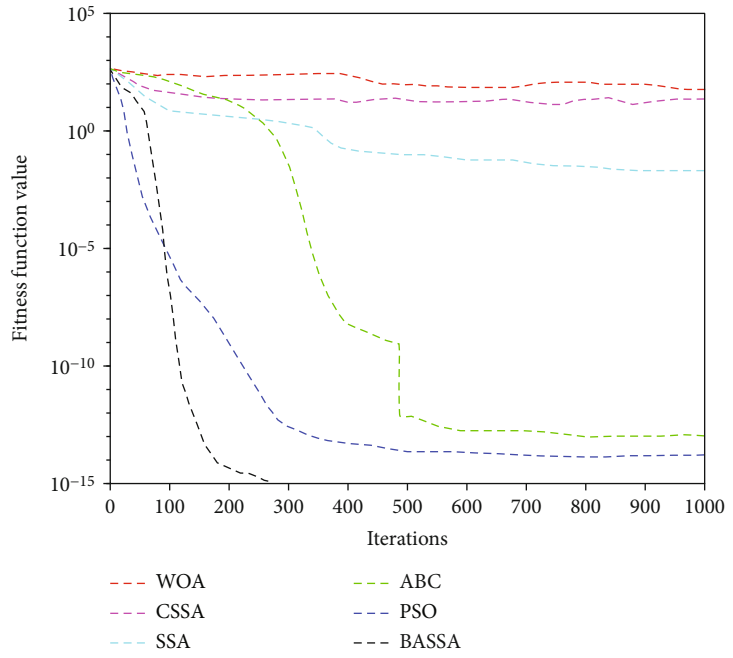

Figure 9: Comparison of the convergence curves of average fitness value for f_9.

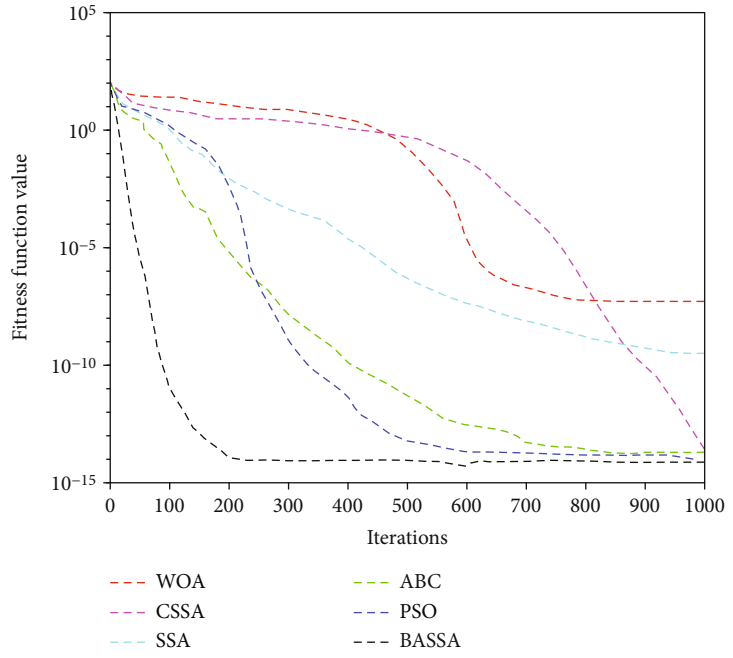

Figure 10: Comparison of the convergence curves of average fitness value for f_{10}.

some of the salp individuals may pass on inaccurate position values, and then, the optimization will fall into a local trap. To overcome these problems, a modified version of the SSA is required.

Since the convergence of the SSA is strongly influenced by the initial population at a later stage of iteration, and the random distribution of the initial population cannot guarantee the diversity, the tent chaotic sequence can be used to increase the randomness, diversity, and aperiodicity of the initial population [39], such that

$$z_{k+1} = \begin{cases} 2z_k & 0 \le z_k < 0.5, \\ 2(1 - z_k) & 0.5 \le z_k \le 1, \end{cases} \quad k = 0, 1, 2 \cdots. \tag{6}$$

In Equation (6), the initial value of z_k can be randomly

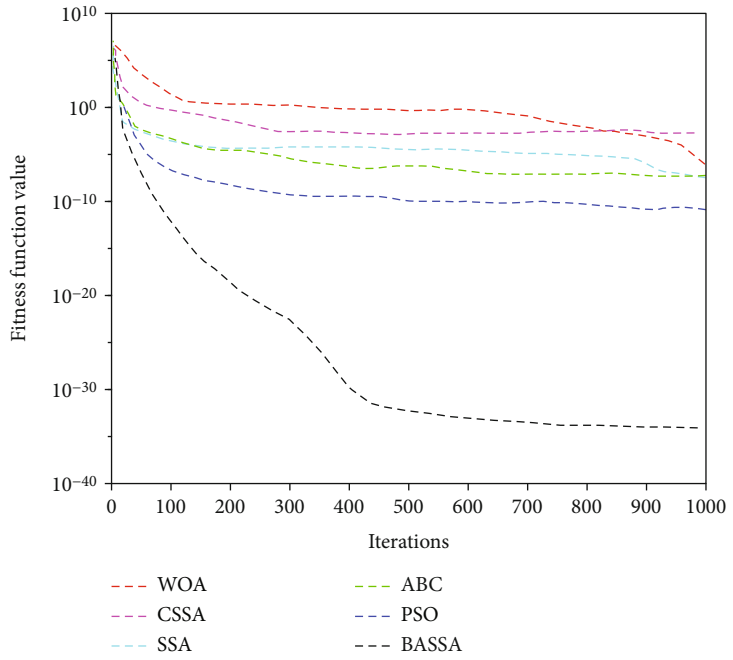

Figure 11: Comparison of the convergence curves of average fitness value for f_{11}.

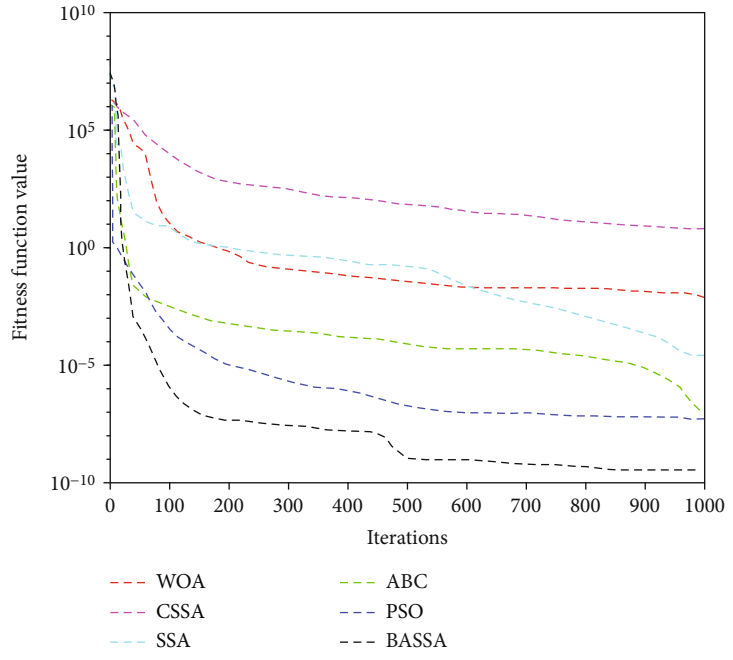

Figure 12: Comparison of the convergence curves of average fitness value for f_{12}.

generated within the value range, and it can be converted into the SSA variables such as

$$x_m^i = lb(m) + \text{rand}\,(N, D) * (ub(m) - lb(m)) * z_k \; k = 0, 1, 2 \cdots. \tag{7}$$

In this way, the procedure for initializing using tent chaotic mapping can be given as,

Step 1. According to the number of variables in the target function n, the initial value of z_k in Equation (6) is assigned with z_0

Step 2. Generate chaotic sequence variables $\{z_{i,k}\, i = 1, 2, \cdots n\}$ according to Equation (7)

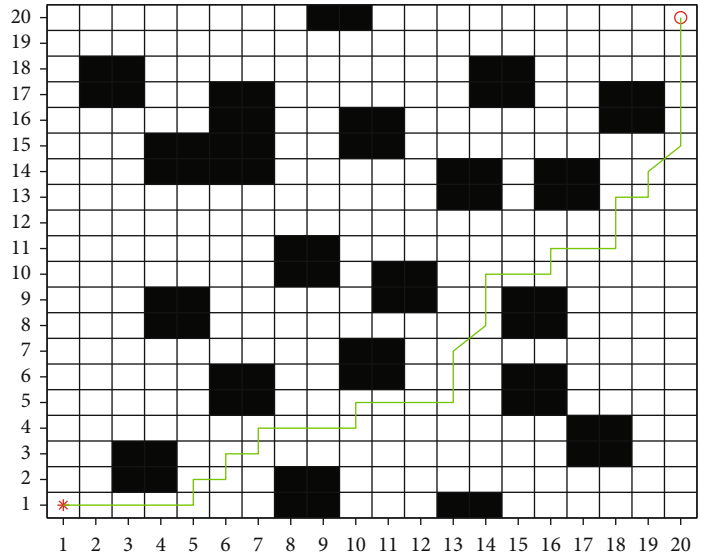

Figure 13: The simulation of path planning with the ABC method.

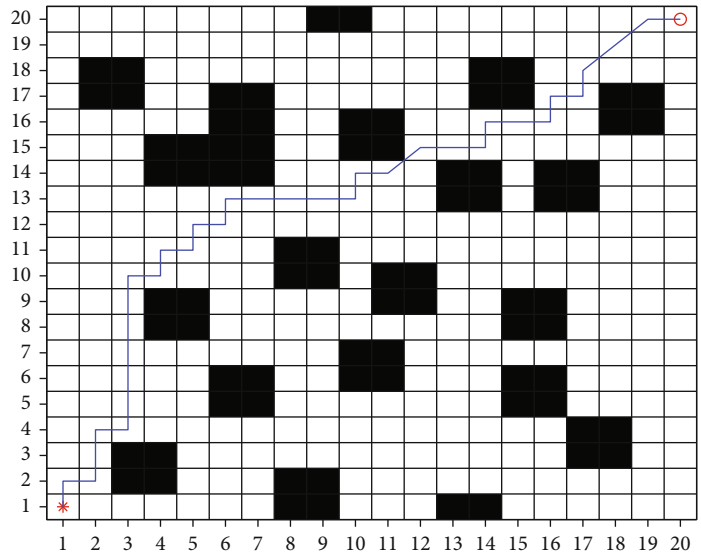

Figure 14: The simulation of path planning with the PSO method.

Step 3. Use Equation (7) to map the chaotic variable z_{ik} to the population solution space to complete the initialization

According to the update rule for the followers, if the jth salp passes on incorrect information to the next, the movement of all subsequent individual will be affected, especially when it finds a local minimum, it will not be likely to move out of this hole, and the whole chain will stay there forever. To solve this problem, an inertia weight strategy is introduced in the position update of the SSA [40], such that

$$\omega_t = (\omega_{\max} - \omega_{\min})\left(\frac{T_{\max} - t}{T_{\max}}\right) + \omega_{\min} \times Z_{k+1}, \quad (8)$$

where $\omega_{\max}$ and $\omega_{\min}$ are the initial and final values of the weights, $T_{\max}$ is the maximum number of iterations, t is the current number of iterations, and Z_{k+1} is the chaotic mapping coefficient [39]. However, the mobile performance of this linearly decreasing inertia adjustment strategy is not satisfactory in the global search. Since the weighting factor is decreasing, in the initial stage of search, the algorithm tends to search globally. However, this duration is too short, the global search performance cannot be fully exploited before the weight factor becomes too small, and the whole chain may have already been trapped in a local optimum. In addition, when the values of $\omega_{\max}, \omega_{\min}$, and t are fixed, the amplitude of ω is also fixed, resulting in a deterioration in the performance of solving complex and nonlinear optimization problems. Therefore, it is required that a large weight is maintained to enhance the global search at the initial stage of optimization, while at the later stage, a small

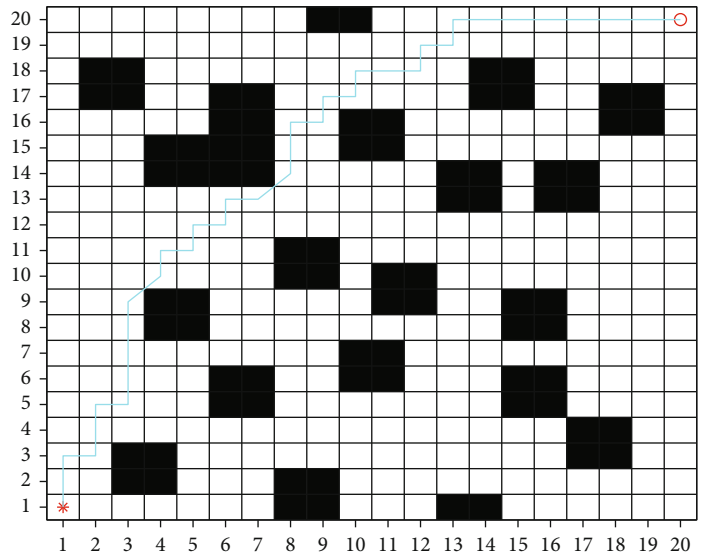

Figure 15: The simulation of path planning with the SSA method.

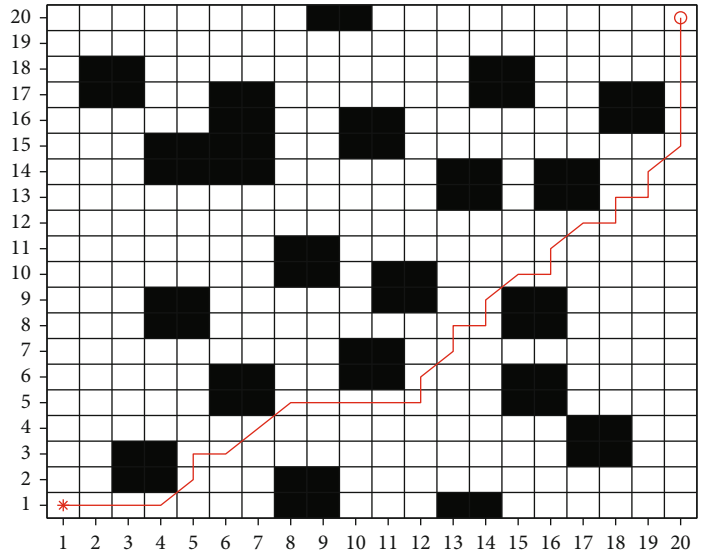

Figure 16: The simulation of path planning with the WOA method.

weight helps to focus on the local search. Inspired by deep learning methods, this nonlinear mapping process can be modeled with a Sigmoid function such that

$$\omega = \omega_{\max} - (\omega_{\max} - \omega_{\min})\left(\frac{1}{1 + e^{-(T_{\max} - t)/T_{\max}}}\right)^3. \quad (9)$$

It can be seen that the output of Sigmoid function has a large initial value, which can ensure the global search capability of the algorithm, and the output value will be reduced gradually to the balance between the global and local search strategy.

To increase the possibility of achieving a global optimum, GA can be introduced in the later stage of the optimization by assigning a big mutation probability such as 0.1 in this paper.

In this way, the optimization process with the improved SSA can be divided into three stages, the first stage is when $t < (T_{\max}/2)$, the improved SSA without the GA is used; at the second stage, when $(T_{\max}/2) < t \leq (2/3)T_{\max}$, if the global output value does not change for 10 consecutive times, the GA operation is used. When $t > (2/3)T_{\max}$, many experiments show that it is very likely to fall into a local optimum at this stage; therefore, a GA process is needed to help with the global search.

4. Experiments and Analysis

To test the effectiveness and performance of the improved algorithm, this paper compares the performance of the proposed algorithm with the original SSA [4], the Chaotic Salp Swarm Algorithm (CSSA) [41], and other intelligent algorithms such as the WOA [39], PSO, and ABC.

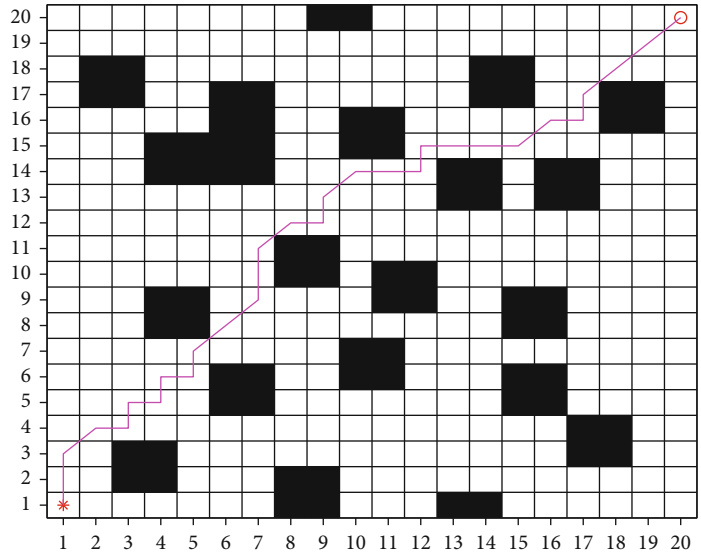

Figure 17: The simulation of path planning with the CSSA method.

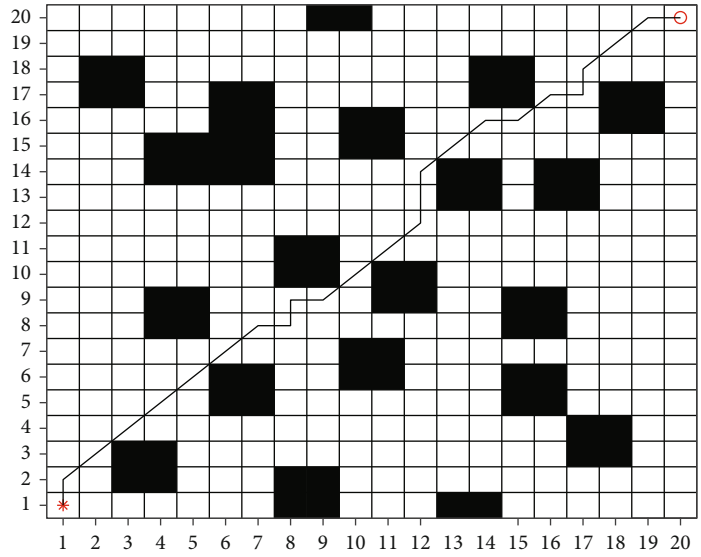

Figure 18: The simulation of path planning with the BASSA method.

In the experiment, there are 12 functions to be tested, among which $f_1 \sim f_7$ are unimodal functions for convergence speed testing, and $f_8 \sim f_{12}$ are multimodal functions for global search testing. The test functions are listed in Table 1.

The tests are carried out on the platform of Matlab 2018a on a PC with 16 G RAM. To confirm the result, each testing is independently run for 50 times to take the average. The dimension of the test function is set to 50, and the number of iterations is 2000. The parameter settings of other algorithms are consistent with the corresponding references. The test results are shown in Table 2.

From the experimental data, it can be seen that in the unimodal function ($f_1 \sim f_7$), the proposed algorithm achieves the best optimal value as well as the average value. This test shows that the proposed algorithm has better optimization stability and the optimal values achieved are at least several orders of magnitude better than that of other algorithms. For f_2 function, the optimal value of the proposed algorithm is nearly 110 orders of magnitude smaller that of the original SSA. In the test with f_2 function, the optimal value of the proposed algorithm is about 144 orders of magnitude less than that of the original SSA and 20 orders of magnitude less than that of the second best PSO algorithm. In the test with functions of f_1, f_3, f_5, f_6, and f_7, the improved SSA has also achieved the minimum values among several algorithms. It shows that the optimization accuracy of the improved SSA has obvious advantage than other algorithms for unimodal functions.

Among the five multimodal functions ($f_8 \sim f_{12}$) tested, the results of the proposed method are significantly better than other algorithms, except for the average value of f_8 in

Table 3: The comparison of simulation results for the six methods in robot path planning.

Methods	The optimal length	Average number of iterations	Average number of turnings
ABC	36.8324	93.3	22.4
PSO	36.2426	92.2	24.6
SSA	36.8284	86.4	25.1
WOA	33.8995	64.7	23.7
CSSA	32.7279	58.5	21.6
Proposed	29.7990	26.6	14.3

which case the CSSA algorithm has achieved the best but very close to the proposed method. In the test with function f_9, it was found that the WOA and the CSSA have been trapped in local minima under the same number of iterations as the proposed algorithm. In the test with functions $f_{10} \sim f_{12}$, the improved algorithm has achieved all the minimum values, which shows better exploration ability in multimodal function to avoid local optima.

Figures 1–12 show the convergence curves of average fitness value for $f_1 \sim f_{12}$, respectively, and it can be seen that the proposed method has the advantages of convergence rate and optimization accuracy over all other algorithms. In the test with the functions of f_2, f_3, f_5, f_7, and f_{11}, the proposed method can jump out of the local optimal solution at an earlier stage than others. Among the 12 functions tested, only the result for the function f_{10} is not optimal for the proposed method, but it is very close to the optimal result. It is clear from these experimental results that the proposed algorithm has better performance in optimization accuracy and convergence speed compared to the original SSA, as well as the CSSA, WOA, PSO, and ABC. The global search performance of the proposed method benefits from the increased diversity of the population and the mobility at the later convergence stage with the GA.

5. The Application in Robot Path Planning

This paper applies the proposed algorithm to the optimization of path planning for robots, and for better evaluation, the above methods are also tested and compared. In this experiment, a 20 ∗ 20-grid map is used to simulate the robot mobile environment, and the parameter settings of each algorithm are listed as follows:

(1) ABC: the number of artificial bees is $M = 50$, and the maximum number of attempts limit = 15

(2) PSO: the number of particles, $M = 50$, the constant of inertia $\omega_{\max}=0.8, \omega_{\min}=0.3$, $c_1=0.5$, and $c_2=0.5$

(3) SSA: the number of salps is $M = 50$, and the individual dimension is $d = 28$

(4) WOA: the position dimension: 20, population size: 50, spiral coefficient $b = 1$, and selection probability $P = 0.5$

(5) CSSA: the settings are consistent with those in the literature [42]

(6) The proposed algorithm: the same as the SSA

For all methods, the total iteration number is 300.

In the test, a 20 ∗ 20-grid map is randomly generated, and the simulated routes of all methods for robot path planning are shown in Figures 13–18, and the related results are listed in Table 3.

From the simulation results, it can be seen that the improved SSA achieves the shortest path length and is 23.6% shorter than ABC algorithm with the longest path length and 9.83% shorter than the CSSA. The average number of iterations achieved by the proposed method is less than half of that of the CSSA. This is due to the high optimization accuracy and better convergence rate of the proposed method. The average number of turnings for the proposed method is 43% less than the original SSA and 33.8% less than the CSSA. This shows that the proposed algorithm travels more straight in the current simulation environment, effectively avoiding unnecessary turns. From the comparison data, it can be seen that the improved SSA is a more efficient way to solve the robot path planning problem.

6. Conclusion

This paper proposed an improved SSA to solve the problems of locomotion and slow convergence of the original SSA. A tent chaotic mapping procedure is introduced to the initialization of the population, which effectively increases the diversity. During the optimization, dynamic inertia weight coefficients are used to maintain the diversity of population and the balance between the global and local search. At a later stage of optimization, GA is implemented to strengthen the global search ability of the algorithm. The proposed method is tested on the 12 most popular test functions and compared with five other evolution methods. The results show that the improved algorithm has better performance in convergence speed and optimization accuracy. Finally, the proposed algorithm is applied to the optimization of robot path planning and compared with the above methods. The experimental data shows that the proposed method finds the optimal path faster than other intelligent algorithms in the same environment with a better route and less iterations.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The author states that there is no conflict of interest.

Acknowledgments

This work is partially supported by the High-level Personnel Startup Program of Beibu Gulf University (grant no.

2018KYQD39) and partially supported by the Special Fund for Bagui Scholars and 100 Scholar Plan of the Guangxi Zhuang Autonomous Region.

References

[1] R. Eberhart and J. Kennedy, "Particle swarm optimization," *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, 1995.

[2] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[3] D. Karaboga and B. Akay, "A comparative study of artificial bee colony algorithm," *Applied Mathematics and Computation*, vol. 214, no. 1, pp. 108–132, 2009.

[4] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp swarm algorithm: a bio-inspired optimizer for engineering design problems," *Advances in Engineering Software*, vol. 114, no. 6, pp. 163–191, 2017.

[5] M. H. Qais, H. M. Hasanien, and S. Alghuwainem, "Enhanced salp swarm algorithm: application to variable speed wind generators," *Engineering Applications of Artificial Intelligence*, vol. 80, pp. 82–96, 2019.

[6] L. Abualigah, M. Shehab, A. Diabat, and A. Abraham, "Selection scheme sensitivity for a hybrid salp swarm algorithm: analysis and applications," *Engineering with Computers*, vol. 38, pp. 1149–1175, 2020.

[7] H. Faris, S. Mirjalili, I. Aljarah, M. Mafarja, and A. A. Heidari, "Salp swarm algorithm: theory, literature review, and application in extreme learning machines," in *Nature-Inspired Optimizers. Studies in Computational Intelligence, vol 811*, S. Mirjalili, J. Song Dong, and A. Lewis, Eds., vol. 811, pp. 185–199, Springer, Cham, 2020.

[8] S. Ahmed, M. Mafarja, H. Faris, and I. Aljarah, "Feature selection using salp swarm algorithm with chaos," in *Proceedings of the 2nd international conference on intelligent systems, metaheuristics & swarm intelligence*, pp. 65–69, New York, 2018.

[9] A. A. Abusnaina, S. Ahmad, R. Jarrar, and M. Mafarja, "Training neural networks using salp swarm algorithm for pattern classification," in *Proceedings of the 2nd international conference on future networks and distributed systems*, pp. 1–6, New York, 2018.

[10] A. Ibrahim, A. Ahmed, S. Hussein, and A. E. Hassanien, "Fish image segmentation using salp swarm algorithm," in *International Conference on advanced machine learning technologies and applications*, pp. 42–51, Springer, Cham, 2018.

[11] H. T. Ibrahim, W. J. Mazher, O. N. Uçan, and O. Bayat, "Feature selection using salp swarm algorithm for real biomedical datasets," *International Journal of Computer Science and Network Security*, vol. 17, no. 12, pp. 13–20, 2017.

[12] A. Ibrahim, S. Mohammed, H. A. Ali, and S. E. Hussein, "Breast cancer segmentation from thermal images based on chaotic salp swarm algorithm," *IEEE Access*, vol. 8, pp. 122121–122134, 2020.

[13] W. He, Y. Xie, H. Lu, M. Wang, and H. Chen, "Predicting coronary atherosclerotic heart disease: an extreme learning machine with improved salp swarm algorithm," *Symmetry*, vol. 12, no. 10, p. 1651, 2020.

[14] J. Gao, W. Ye, J. Guo, and Z. Li, "Deep reinforcement learning for indoor mobile robot path planning," *Sensors*, vol. 20, no. 19, p. 5493, 2020.

[15] A. Stentz, "Optimal and efficient path planning for partially known environments," in *Intelligent Unmanned Ground Vehicles*, pp. 203–220, Springer, Boston, MA, 1997.

[16] M. A. Contreras-Cruz, V. Ayala-Ramirez, and U. H. Hernandez-Belmonte, "Mobile robot path planning using artificial bee colony and evolutionary programming," *Applied Soft Computing Journal*, vol. 30, pp. 319–328, 2015.

[17] F. H. Ajeil, I. K. Ibraheem, A. T. Azar, and A. J. Humaidi, "Grid-based mobile robot path planning using aging-based ant colony optimization algorithm in static and dynamic environments," *Sensors*, vol. 20, no. 7, p. 1880, 2020.

[18] H. Zhang, W. Lin, and A. Chen, "Path planning for the mobile robot: a review," *Symmetry*, vol. 10, no. 10, p. 450, 2018.

[19] Y. Chen, G. Luo, Y. Mei, J. Q. Yu, and X. L. Su, "UAV path planning using artificial potential field method updated by optimal control theory," *International Journal of Systems Science*, vol. 47, no. 6, pp. 1407–1420, 2016.

[20] P. Vadakkepat, K. C. Tan, and W. Ming-Liang, "Evolutionary artificial potential fields and their application in real time robot path planning," in *Proceedings of the 2000 congress on evolutionary computation. CEC00 (Cat. No. 00TH8512)*, pp. 256–263, IEEE, La Jolla, CA, USA, 2000.

[21] Q. Zhang, D. Chen, and T. Chen, "An obstacle avoidance method of soccer robot based on evolutionary artificial potential field," *Energy Procedia*, vol. 16, pp. 1792–1798, 2012.

[22] F. Duchoň, A. Babinec, M. Kajan et al., "Path planning with modified a star algorithm for a mobile robot," *Procedia Engineering*, vol. 96, pp. 59–69, 2014.

[23] X. Liu and D. Gong, "A comparative study of A-star algorithms for search and rescue in perfect maze," in *2011 International Conference on Electric Information and Control Engineering*, pp. 24–27, IEEE, Wuhan, 2011.

[24] C. Wang, L. Wang, J. Qin et al., "Path planning of automated guided vehicles based on improved A-star algorithm," in *2015 IEEE International Conference on Information and Automation*, pp. 2071–2076, IEEE, Lijiang, China, 2015.

[25] Y. Deng, Y. Chen, Y. Zhang, and S. Mahadevan, "Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment," *Applied Soft Computing*, vol. 12, no. 3, pp. 1231–1237, 2012.

[26] Y. Chen, S. Shen, T. Chen, and R. Yang, "Path optimization study for vehicles evacuation based on Dijkstra algorithm," *Procedia Engineering*, vol. 71, pp. 159–165, 2014.

[27] A. Sedeno-Noda and M. Colebrook, "A biobjective Dijkstra algorithm," *European Journal of Operational Research*, vol. 276, no. 1, pp. 106–118, 2019.

[28] S. M. LaValle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning," *The Annual Research Report*, p. 98, 1998.

[29] J. Zhu, S. Zhao, and R. Zhao, "Path planning for autonomous underwater vehicle based on artificial potential field and modified RRT," in *2021 International Conference on Computer, Control and Robotics (ICCCR)*, pp. 21–25, IEEE, Shanghai, China, 2021.

[30] R. Mashayekhi, M. Y. I. Idris, M. H. Anisi, I. Ahmedy, and I. Ali, "Informed RRT*-connect: an asymptotically optimal single-query path planning method," *IEEE Access*, vol. 8, pp. 19842–19852, 2020.

[31] B. Song, Z. Wang, and L. Zou, "An improved PSO algorithm for smooth path planning of mobile robots using continuous high-degree Bezier curve," *Applied Soft Computing*, vol. 100, article 106960, 2021.

[32] Z. Masoumi, J. Van Genderen, and N. A. Sadeghi, "An improved ant colony optimization-based algorithm for user-centric multi-objective path planning for ubiquitous environments," *Geocarto International*, vol. 36, no. 2, pp. 137–154, 2021.

[33] K. Liu, C. Xv, D. Huang, and X. Ye, "UAV path planning based on improved whale optimization algorithm," in *2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE)*, pp. 569–573, IEEE, Guangzhou, China, 2021.

[34] Z. Yan, J. Zhang, J. Zeng, and J. Tang, "Water wave optimization algorithm for autonomous underwater vehicle path planning problem," *Journal of Intelligent & Fuzzy Systems*, vol. 40, no. 5, pp. 9127–9141, 2021.

[35] M. A. Hossain and I. Ferdous, "Autonomous robot path planning in dynamic environment using a new optimization technique inspired by bacterial foraging technique," *Robotics and Autonomous Systems*, vol. 64, pp. 137–141, 2015.

[36] M. B. Metea, "Route planning for intelligent autonomous land vehicles using hierarchical terrain representation," *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, vol. 4, pp. 1947–1952, 1987.

[37] A. Botea, M. Muller, and R. J. Schaeffe, "Near optimal hierarchical path-finding," *Journal of Game Development*, vol. 1, no. 1, pp. 7–28, 2004.

[38] G. Liu, T. Li, Y. Peng, and X. Hou, "The ant algorithm for solving robot path planning problem," *Proceedings of the Third International Conference on Information Technology and Applications. (ICITA'05)*, vol. 2, 2005.

[39] M. Qinghua, Y. Lin, and W. Yanliang, "Grey wolf algorithm based on improved tent chaos and simulated annealing," *Mathematics Practice and Cognition*, vol. 51, no. 5, pp. 147–161, 2021.

[40] M. A. Arasomwan and A. O. Adewumi, "On the performance of linear decreasing inertia weight particle swarm optimization for global optimization," *The Scientific World Journal*, vol. 2013, Article ID 860289, 12 pages, 2013.

[41] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

[42] G. I. Sayed, G. Khoriba, and M. H. Haggag, "A novel chaotic salp swarm algorithm for global optimization and feature selection," *Applied Intelligence*, vol. 48, no. 10, pp. 3462–3481, 2018.