[1] Yufan Lu

Optimization Study on Indoor Robot Path Planning based on Improved Differential Evolution Algorithm

Abstract: - Aiming at the problem that there are many obstacles indoor and the situation is complex, and the traditional robot path planning algorithm is ineffective, this paper proposes a grid path planning algorithm based on the improved differential evolution algorithm for the robot in the building. Firstly, the grid model is used to construct the path planning model of the robot in the building, and the running direction model of the robot indoor is obtained. On this basis, the intelligent algorithm optimization model of the path planning of the robot indoor is given. Secondly, the differential evolution algorithm is introduced to optimize the structural optimization index, and the Q learning method is used to improve the optimization algorithm to realize the building. Finally, the performance advantages of the proposed algorithm in obstacle avoidance and optimization planning of robot path are verified by the simulation of an example of robot path planning in building.

Keywords: differential evolution; in-building robots; grid model; path planning.

I. INTRODUCTION

The differential evolution algorithm is a new optimization calculation method, and it is not different in the design idea and structure form from other existing intelligent optimization algorithms. Its specialty is to treat the differential information between different numbers of population individuals as the disturbance quantity in the population evolution process[5~6], which will lead to a certain jump in the evolution process of the algorithm, thereby increasing its randomness in the search target. Besides, it has stronger field adaptability. The differential evolution algorithm has some comparative advantages in performance. The main reason is that in the early stage of population evolution, the large differences between individuals will lead to a large amount of the above set disturbance, which will increase the search and exploration performance of the algorithm individuals for the global range and make it not easy to have premature convergence. In the later stage of algorithm evolution, with the convergence of individual evolution, the differences between individuals are relatively small, which will decrease the disturbance in the evolutionary model, and then increase the local area mining performance of the algorithm in the later evolution stage, thus accelerating the convergence of the algorithm[1-2].

In this paper, to solve the path planning problem of indoor robots, the differential evolution algorithm is employed to optimize the robot path. At the same time, to improve the path planning accuracy of the differential evolution algorithm, based on the gridization of the object model, the differential evolution algorithm is improved by the Q reinforcement learning to improve the optimization performance of the algorithm. The experimental results reveal that the proposed algorithm improves the optimization effect of path planning of indoor robots[3-4].

II. PATH PLANNING MODEL OF INDOOR ROBOTS

A. grid environment model construction

F For the internal static environment, indoor obstacles can be measured. In the path-planning process of indoor robots, it can be assumed that the operating environment of the target object is given in advance. Under this assumption, the starting point, end point and obstacle condition of the operating environment of indoor robots are known in advance. The operating environment of indoor robots can be constructed into a two-dimensional motion space, where a two-dimensional plane rectangular coordinate system can be defined. The origin is located in the lower left corner of the space, the maximum value of the horizontal coordinate is set to $X_{\max}$, and the maximum value in the vertical direction is set to $Y_{\max}$. The resolution index of the coordinate space is set as the step size parameter $step$ during the robot operation, which is simplified to 1[5-6].

Using the grid model to divide the running environment of the robot. The total number of horizontal grid models is [10~11] $m = ceil\left(X_{\max} / step\right)$, the total number of vertical grid models is $n = ceil\left(Y_{\max} / step\right)$, and the

[1]*Correspondingauthor: Department of Electronic and Information Engineering(Sussex Artificial Intelligence Institute), Zhejiang Gongshang University, Hangzhou Zhejiang, 310018, China

grid model matrix of the running environment with a size of $m \times n$ can be obtained. According to the different sizes of obstacles, different numbers of grids can be used to represent them. Meanwhile, according to the location of obstacles, the grid can be divided into the grid that can be moved by the robot and the grid that cannot be moved by the robot. As the selected resolution index is relatively fine, if the size of the obstacle is less than that of a grid, it is approximated to a grid size, which will not have a great impact on the calculation process of the algorithm. The processing results of the grid model are shown in Fig.1, in which the blank area represents the grids that can be moved by the robot, while the black area represents the grids that cannot be moved by the robot[7-8].

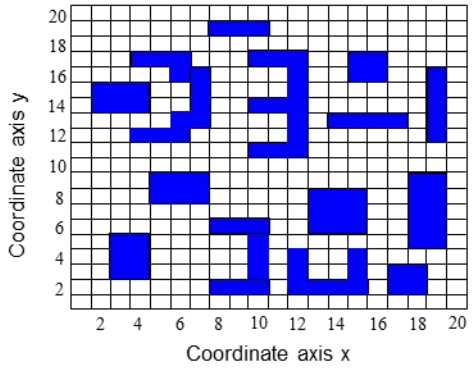


Fig.1 grid environment model

In essence, the construction process of the grid model of the robot's operating environment is a discretization process of the spatial region. It is impossible for the discretization to present the original information of the map model, and there will be some differences. In particular, the selected map grid resolution influences the efficiency and accuracy of the calculation process, and these two effects are mutually exclusive. The higher the resolution index is selected, the lower the calculation efficiency of the algorithm is, and the higher the calculation accuracy is. After constructing the discrete grid model of the robot's operating environment, the robot path optimization process is to find a set of grid subsets that can ensure the optimal operation of the robot in a series of discrete grid models. The process can be expressed as[12]:

$$P = \{p_1, p_2, \cdots, p_i, \cdots, p_n\} \quad (1)$$

$$P = \{(x_1, y_1), (x_2, y_2), \cdots, (x_i, y_i), \cdots, (x_n, y_n)\} \quad (2)$$

Where parameter P is a grid subset solution scheme for the robot's operation; p_1 is the starting point grid for the robot's operation; p_n is the ending point grid of the robot's operation; p_i is the ith grid for the robot's operation. Selecting the optimal path of the robot as the target, the following formula can be obtained:

$$dist(i) = \sum_{i=2}^{n} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \quad (3)$$

Where parameter term $dist(i)$ represents a subset of the robot path running grid with the end point as grid i, and the total length of the running path can be characterized by the total distance of all grid nodes. Parameter term (x_i, y_i) represents the coordinate position of the grid; parameter term n represents is the total number of grids in the robot's running route.

B. 2.2 Objective function

The path optimization problem of indoor robots can be modeled as a target model optimization problem. What's the objective function characterized is the best running path of the robot, and the robot obstacle avoidance problem is modeled as the objective model constraint. The optimization objective model can be obtained as follows:

$$\min g(x), x \in R^n \quad (4)$$

$$f_i(x) \le T_i, i = 1, 2, \cdots, m \quad (5)$$

Where parameter term $g(x)$ is the optimization goal of robot path planning; parameter term $f_i(x)$ is the condition constraint of robot path planning; and parameter term m is the number of constraints for robot path

planning. For the trajectory optimization of the indoor robot, the total length of the robot's running path can be used as the optimization goal:

$$fit(i)=\frac{1}{dist(i)}=\frac{1}{\sum_{i=2}^{n}\sqrt{(x_i-x_{i-1})^2+(y_i-y_{i-1})^2}} \quad (6)$$

Where parameter term $fit(i)$ is the target adaptation value of the robot's running environment, and parameter term $dist$ is the robot's running distance parameter. The objective function should reserve the best grid position experienced in history and the current grid position information at the same time. Then grid fitness function during the operation of the robot can be characterized as follows:

$$fit(i)=distH(i)+distF(i) \quad (7)$$

$$distH(i)=\sum_{j=2}^{i}\sqrt{(x_j-x_{j-1})^2+(y_j-y_{j-1})^2} \quad (8)$$

$$distF(i)=\sqrt{(x_{ij}-x_i)^2+(y_{ij}-y_i)^2}+\sqrt{(x_n-x_{ij})^2+(y_n-y_{ij})^2} \quad (9)$$

Where $distH(i)$ represents the total path length from the robot's starting point grid to the current running grid position; x_i , y_i represents the horizontal and vertical coordinates of the robot's running grid position; x_{ij} and y_{ij} represent the horizontal and vertical coordinates on the position of the candidate grid nodes generated by evolution; and M represents the number of model evolution individuals. Then the fitness model can be constructed as shown in Fig. 2.

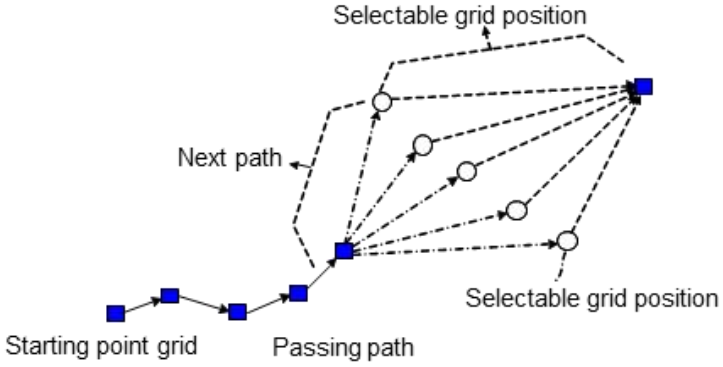


Fig. 2 Evolution process of fitness function

III. IMPROVED DIFFERENTIAL EVOLUTION ALGORITHM

A. Algorithm Description

Assuming that the population size in the differential evolution algorithm is NP, then the form of generation G of the differential evolution population can be obtained as[13~14]:

$$P_G=\{\mathbf{X}_1(G),\mathbf{X}_2(G),\cdots,\mathbf{X}_{NP}(G)\} \quad (10)$$

Where parameter term $\mathbf{X}_i(G), i\in 1,\cdots,NP$ is the population individual of the differential evolution algorithm. The calculation process of the differential evolution algorithm is as follows:

Procedure 1: (initial structure) In the initial stage of the differential evolution algorithm, namely, $G=0$, the individual $\mathbf{X}_i(0)$ of the algorithm population is randomly selected in the preset interval $[\mathbf{X}_{\min},\mathbf{X}_{\max}]$. The initial results are as follows:

$$\begin{cases}\mathbf{X}_{\min}=\{x_{\min-1},\cdots,x_{\min-D}\}\\ \mathbf{X}_{\max}=\{x_{\max-1},\cdots,x_{\max-D}\}\end{cases} \quad (11)$$

Where parameter term D represents is the population dimension parameter. In the early stage of population evolution, namely, G=0, the initialization process of element j in individual i is as follows:

$$x_{ij}(0) = x_{\min-j} + rand_{ij}(0,1) \times \left(x_{\max-j} - x_{\min-j}\right) \quad (12)$$

Where parameter term $rand_{ij}(0,1)$ is the parameter selection distribution function in the standard interval $[0,1]$, which is uniformly distributed. Similarly, the randomization method is used to select the crossover factor Cr in the standard interval $[0,1]$.

Procedure 2: (Individual variation) The operation process is to mutate the selected two individuals ($\mathbf{X}_{rand-1}(G)$, $\mathbf{X}_{rand-2}(G)$) in a random manner, and a new population individual $\mathbf{V}_i(G)$ can be obtained by superimposing the operation on the individual $\mathbf{X}_i(G)$:

$$\begin{aligned}\mathbf{V}_i(G) = \mathbf{X}_i(G) &+ F_1\left(\mathbf{X}_{best}(G) - \mathbf{X}_i(G)\right) \\ &+ F_2\left(\mathbf{X}_{rand-1}(G) - \mathbf{X}_{rand-2}(G)\right)\end{aligned} \quad (13)$$

Where parameter term F is the scale factor parameter of the differential evolution algorithm, with a value interval of $F \in [0,2]$, and its selection size affects whether the evolution process of the algorithm is local search or global development.

Procedure 3: (Individual crossover) There are many ways of individual crossover, such as exponential individual crossover and binomial individual crossover. Firstly, the definitions of these two ways are provided.

Binomial individual crossover: In the process of differential evolution algorithm, the individual crossover operation between the target $\mathbf{X}_i(G)$ and the donor $\mathbf{V}_i(G)$ can obtain new individual $\mathbf{U}_i(G)$ by crossover[15]:

$$u_{ij}(G) = \begin{cases} v_{ij}(G), \ if\ rand_{ij} \le Cr\ or\ j = j_{rand} \\ x_{ij}(G), otherwise \end{cases} \quad (14)$$

Index individual crossover: for setting the individual value interval $[1,D]$, the starting point n of the target $\mathbf{X}_i(G)$ is obtained by random method. The value is an integer, which marks the starting position of element exchange between individuals. In the same way, the integer L is selected randomly in the value interval $[1,D]$, which marks the number of elements involved in the individual crossover process. Thus, the individual crossover process can be obtained:

$$u_{ij}(G) = \begin{cases} v_{ij}(G), \ for\ j = \langle n \rangle_D, \cdots, \langle n + L - 1 \rangle_D \\ x_{ij}(G), otherwise \end{cases} \quad (15)$$

Where parameter term $\langle \cdot \rangle_D$ represents the modulus function of the differential evolution individual, and its modulus is D.

Procedure 4: (Individual selection) For the minimization optimization problem, the optimization goal is $f(\mathbf{x})$, and the elite selection of the differential evolution algorithm is:

$$\mathbf{X}_i(G+1) = \begin{cases} \mathbf{U}_i(G), if\ f\left(\mathbf{U}_i(G)\right) \le f\left(\mathbf{X}_i(G)\right) \\ \mathbf{X}_i(G), if\ f\left(\mathbf{U}_i(G)\right) > f\left(\mathbf{X}_i(G)\right) \end{cases} \quad (16)$$

B. Q learning factor

Q-learning is essentially an algorithm improvement strategy with reinforced learning ability. Through certain operations, it can give corresponding punishment (or reward) to the evolutionary process of the evolutionary algorithm, thus guiding the optimization calculation method to evolve towards a more clear goal. It is a self-learning optimization control process of evolutionary algorithm, and the reward return obtained is the neutralization process of "immediate reward" and "discount reward", which helps to balance the optimization process of the intelligent optimization algorithm.

Through the reinforcement learning of the Q learning factor, the algorithm can control and improve the evolutionary environment of differential evolution individuals, and use the set reward and punishment mechanism to influence the running direction of the differential evolution algorithm. However, in the specific

environment, it is very complicated to evaluate the reward and punishment of the unknown state s' in the current state s. The solution is to select the best reward return for processing, which is a discounted reward.

$S=\{s_1,\cdots,s_n\}$ is the set of individual states in a given evolutionary environment; $A=\{a_1,\cdots,a_n\}$ is a set of actions that can be selected by individuals in an evolutionary environment; $r(s_i,a_j)$ is the reward that the evolutionary individual selects the action a_j under the state condition s_i; $\delta(s_i,a_j)$ is the probability of the occurrence of the transition function s_k when the evolutionary individual selects the action a_j in the state s_i; γ is the discount factor, with a value interval of $\gamma\in[0,1]$. The greater the value, the greater impact of the reward return on the guidance of the algorithm evolution; $Q(s_i,a_j)$ is the total reward return, corresponding to the selection situation of the action a_j in state s_i. The total reward return $Q(s_i,a_j)$ is:

$$\begin{aligned} Q(s,a) &= r(s,a)+\gamma \overset{*}{V} Q\left(\delta(s,a)\right) \\ &= r(s,a)+\gamma \max_{a'} Q\left(\delta(s,a),a'\right) \end{aligned} \quad (17)$$

Where parameter term V represents the total reward of the differential evolution individual in the state s. The improved form of the differential evolution based on the Q-learning factor is as follows:

$$\begin{aligned} Q(s,a)\leftarrow (1-\alpha)Q(s,a)+ \\ \alpha\times\left(r(s,a)+\gamma \max_{a'} Q\left(\delta(s,a),a'\right)\right) \end{aligned} \quad (18)$$

The function of the differential evolution operation is to make the selection of action a in the state $Q(s,a)$ incremental relative to the state $\delta(s,a)$, aiming to ensure that the subsequent reward $r(s,a)$ is increased relative to $Q(s,a)$, which can guarantee the correctness of the evolution direction. If $\alpha=0$, the differential evolution individual learning process is terminated; if $\alpha=1$, the differential evolution individual learning process is continued, and the latest differential evolution individual information is considered. The discount factor γ mainly aims to judge the importance of information. If $\gamma=0$, the current reward is more important, but if $\gamma=1$, the long-term reward is more important.

C. 3.3 Algorithm process description

The algorithm in this paper mainly sets the importance weight of the optimization object based on the differential evolution process, so as to guide the algorithm to achieve more effective global optimization. The algorithm process is as follows:

Procedure 1: (Initialization) Initialize the individual population of the differential evolution algorithm in the set initial interval. The population size is NP, and the population dimension is D. In the Q learning factor, the initial value of Q-table is small. If the maximum value of Q-value is set to 100, the maximum value of Q-value is assigned to 1;

Procedure 2: (Parameter self-learning) The scale factor F has an important impact on the reward and punishment factor of Q-table. For example:

If $Q(s_i,10F_j)$ is the maximum value in the set $Q(s_i,10F_l),l=1,\cdots,10$, then the value of $F=F_j$ is the maximum at the state s_i. In the factor library $\{F_1,\cdots,F_{10}\}$, the selection probability of $F=F_j$ is:

$$P(F_j)=Q(s_i,10F_j)\Big/\sum_{l=1}^{10}Q(s_i,10F_l) \quad (19)$$

To ensure the self-learning adaptability of Q-table, a specific r is selected by generating random value r in the interval $[0,1]$, and it meets:

$$\sum_{m=1}^{j-1} P\left(F=F_m\right) < r \le \sum_{m=1}^{j} P\left(F=F_m\right)$$
$$\Rightarrow \frac{\sum_{m=1}^{j-1} Q\left(s_i,10F_m\right)}{\sum_{l=1}^{10} Q\left(s_i,10F_l\right)} < 1 \le \frac{\sum_{m=1}^{j} Q\left(s_i,10F_m\right)}{\sum_{l=1}^{10} Q\left(s_i,10F_l\right)} \quad (20)$$

Procedure 3: (Differential operation) The individual state of the differential evolution algorithm is allocated. Let f_i be the target of individual i in the differential evolution algorithm. Then it is normalized by using $f_i \Big/ \sum_{j=1}^{NP} f_i$. The sorting list is obtained by descending operation, and the state of the individuals with grade r is s_r. The above procedures are repeated when $r = 1:NP$.

Procedure 5: (Updating of Q-table) If the state of the differential evolution individual changes from s_i to s_k after the operation F_j is implemented, then the corresponding individual target increases, and the positive reward parameter $Q\left(s_i,10F_j\right)$ is updated as:

$$Q\left(s_i,10F_j\right) = (1-\alpha)Q\left(s_i,10F_j\right) + \alpha\left(reward\left(s_i,10F_j\right) + \gamma \max_{F'}\left(s_k,10F'\right)\right) \quad (21)$$

Otherwise, the parameter factor $Q\left(s_i,10F_j\right)$ is updated based on negative rewards.

Procedure 5: (Convergence judgment) If the convergence condition of the algorithm is not satisfied, the above procedures 2 ~ 4 should be performed repeatedly.

D. 3.4 Convergence and complexity analysis

Theorem 1: In the evolution of the differential evolution algorithm, the development direction of the algorithm population shows a monotonic variation trend. Meanwhile, the Q-learning process still satisfies the monotonic variation trend, that is:

$$f\left(X_i(n+1)\right) < f\left(X_i(n)\right) \quad (22)$$

Proof: The evolution process of the differential evolution algorithm selects and maintains excellent individuals by using the greedy method, so its evolution process shows a certain monotonic trend.

Theorem 1: The differential evolution algorithm population with Q-learning reinforcement presents a Markov chain sequence form, which converges with a probability of 1. The convergence position is the global optimal solution M of the target model, and the convergence model form can be obtained: $\lim_{n\to\infty} P\left\{X(n) \in M \,\middle|\, X(0) = B_0\right\} = 1$.

Proof: Supposing that the population individual has been located at the global optimal position of the target solution when the differential evolution algorithm with Q-learning reinforcement is at the moment n, then according to the convergence model described in Theorem 1, at the moment $n+1$, the population individual $X(n+1)$ will also be located at the global optimal position M of the target solution. The model form can be obtained as follows:

$$P\left\{X(n+1) \in M \,\middle|\, X(n) \in M\right\} = 1 \quad (23)$$

Thus,

$$P\left\{X(n+1) \in M\right\} = P\left\{X(n) \notin M\right\} \cdot P\left\{X(n+1) \in M \,\middle|\, X(n) \notin M\right\} + P\left\{X(n) \in M\right\} \cdot P\left\{X(n+1) \in M \,\middle|\, X(n) \in M\right\} \quad (24)$$

The above formula can be denoted as: $P\left\{X(n+1) \in M \,\middle|\, X(n) \notin M\right\} \ge g(n) \ge 0$, $\lim_{n\to\infty} \prod_{i=1}^{n} \left[1 - g(i)\right] = 0$. Then:

$$\begin{aligned}&1-P\{X(n+1)\in M\}\le\\&[1-g(n)]\cdot[1-g(n-1)]\cdot[1-P\{X(n-1)\in M\}]\\&\Rightarrow\lim_{n\to\infty}P\{X(n+1)\in M\}\ge 1\end{aligned}\tag{25}$$

As the condition $P\{X(n+1)\in M\}\le 1$ is met, $\lim_{n\to\infty}P\{X(n+1)\in M\}=1$ can be obtained, namely, the differential evolution algorithm using Q-learning reinforcement meets the convergence requirements. The proof process ends.

The complexity of the local exploration of each target position in the differential evolution stage is $O(FN)$, the complexity of calculating the fitness value is $O(FN)$, and the time complexity of calculating the percentage of the return rate of each local target position in the entire differential evolution population is $O(FN)$; in the evolution process, the roulette method is used for individual evolution, so in the worst case, the time complexity is $O(FN^2)$. Thus, the total computational complexity of the algorithm is $O(FN^2)$.

IV. EXPERIMENTAL ANALYSIS

Experimental environment: The computer's CPU is Intel i5-4500K, the computer's memory model is DDR4-2400K, the operating system is win10 flagship version, and the simulation platform is MATLAB R2013a. The comparison algorithm selects the standard ant colony algorithm, differential evolution algorithm, and the proposed algorithm.

A. 4.1 Algorithm performance test

The selected test model is as follows: $f1$ is the Dejong test model, and $f2$ is the Griewank test model. The global optimal value of the test models $f1\sim f2$ is 0. The test models are as follows:

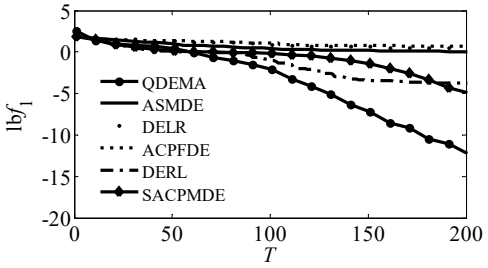
$$f1=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(x_i/\sqrt{i}\right)+1,\ [-60,60]\tag{26}$$

$$f2=\frac{\sin^2\left(\sqrt{x_1^2+x_2^2}\right)-0.5}{\left(1+0.001\left(x_1^2+x_2^2\right)\right)^2}-0.5,\ [-100,100]\tag{27}$$

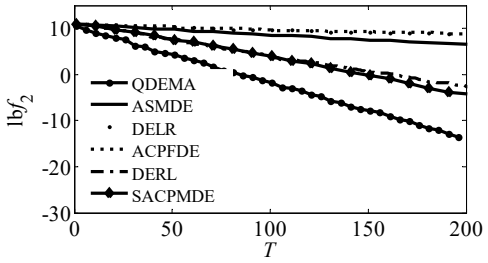
Performance comparison algorithms include ACPFDE, ASMDE, DERL, SACPMDE, and DELR. Parameter settings are as follows: population dimension parameter $D=30$, population size parameter $NP=200$. Population size setting affects the evolutionary performance of the algorithm. The larger the population, the better the evolutionary convergence of the algorithm, but the computational complexity of the algorithm is greatly increased. The simulation convergence accuracy is VTR=10^{-6}, and the iteration upper limit is 200.

The value of the cross probability parameters is $CR\in[0.3,0.9]$. The parameter settings of SACPMDE and ASMDE are the same as those in Reference [4], and the parameter settings of DERL and DELR are the same as those in Reference . The simulation results are shown in Fig.3.

Figures 3a~3b are the convergence experimental results of the selected simulation test algorithm on the selected two simulation test models. There are two convergence trends in the experimental comparison data of the algorithm shown in Figure 3a: ① The proposed algorithm (QDEMA) and SACPMDE algorithm show good convergence results, and there is no premature convergence failure. The SACPMDE algorithm presents significant slow characteristics. ② The other simulation comparison algorithms are premature convergence, and the algorithm evolution process fails. Figure 3b shows that several simulation comparison algorithms are not premature convergence, but the convergence speed of the QDEMA algorithm shows a significantly faster convergence trend.



（a）Dejong



（b）Griewank

Fig.3 Comparison of simulation test results

To verify the computational complexity of the proposed algorithm, the optimized calculation time of the algorithms is selected as the experimental index for comparison tests. The experimental results are shown in Table 1.

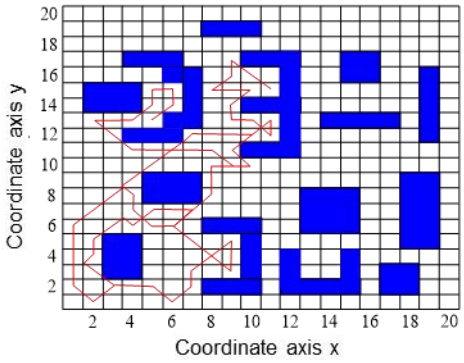
Table 1 Calculation time of algorithms

Test model	Algorithm	Convergence precision	Calculation time
Dejong	QDEMA	4.36E-7	2.562
	SACPMDE	6.94E-7	3.981
	DERL	2.35E-6	3.574
	ASMDE	6.83E-6	5.693
	ACPFDE	9.35E-6	6.821
	DELR	5.67E-5	5.618
Griewank	QDEMA	2.98E-7	2.276
	SACPMDE	5.83E-7	3.564
	DERL	1.87E-6	3.473
	ASMDE	5.16E-6	5.548
	ACPFDE	8.25E-6	6.715
	DELR	4.68E-5	5.287

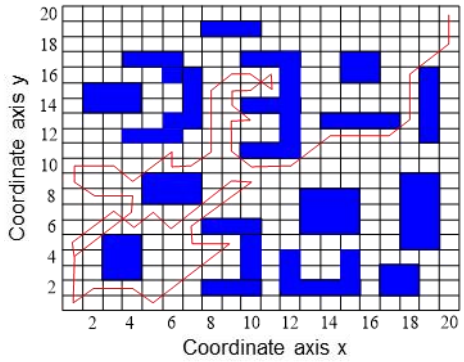
Table 1 shows the convergence accuracy and algorithm calculation time of the proposed algorithm and the selected centralized comparison algorithm. In the calculation process of the algorithms, the simulation convergence accuracy is selected as VTR=10^{-6}, and the iteration upper limit is 200. Therefore, the convergence accuracy obtained in this experiment is near the convergence accuracy. Among them, the proposed algorithm has better convergence accuracy and calculation time than the selected experimental comparison algorithm, which reflects the better convergence accuracy and lower computational complexity of the proposed algorithm.

B. Simulation scene tests

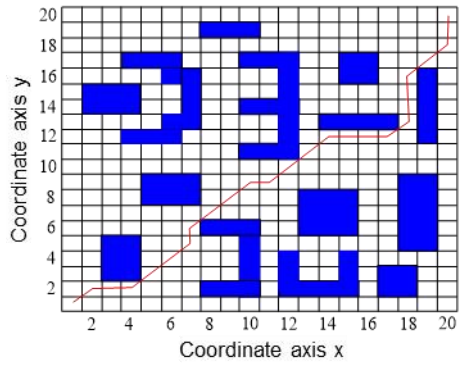
Scenario 1: Set the obstacles in the grid environment as diverse and complex as possible. The location of the obstacles is relatively independent and there are no specified rules. The starting point of the grid corresponding to the starting point of the indoor robot is set to be 1. The coordinate position of the starting point is $(0.5,0.5)$, and the end point of the grid is 400. The coordinate position of the end point is $(19.5,19.5)$. The differential evolution population size parameter is set to 200, and the evolution termination algebra is set to 50. The simulation results are shown in Fig. 3.



(a) Optimization results of ant colony algorithm



(b) Evolution results of differential evolution algorithm



(c) Results of the improved algorithm in this paper

Fig.3 Comparison of Path Planning Results

According to the experimental results shown in Fig.3, in the path planning of indoor robots conducted by the ant colony algorithm, there are some problems in the early stage of the algorithm evolution, such as weak evolutionary goal, large degree of freedom of individual population, and aimless search. The main reason is the pheromone search method adopted by the ant colony algorithm itself doesn't have obvious pheromone differences at different positions in the early algorithm evolution stage, which leads to the lack of guidance. The differential evolution algorithm is also complicated in the evolution process. This situation is contrary to the pheromone situation. In the early evolution stage of the differential evolution algorithm, there is a large random distribution between individuals, which leads to a large randomness of the algorithm when the algorithm individual is used as the perturbation term of the differential evolution process, thus resulting in a large jump of the individual in the early stage of the algorithm, but the algorithm's realizable solution can be found in the later stage. As shown in Fig. 3c, the indoor robot path planning algorithm using the improved differential evolution algorithm can find relatively better path planning results. Fig. 4 shows the convergence curves of the three algorithms.

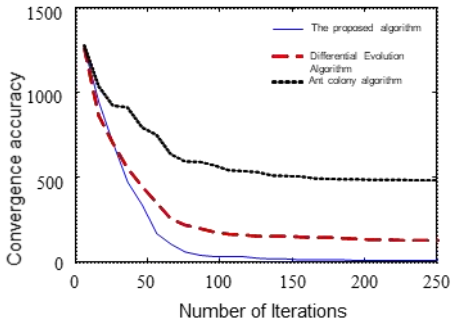
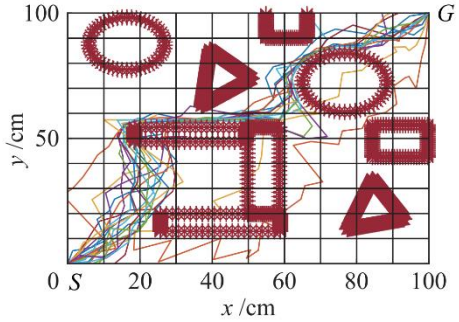


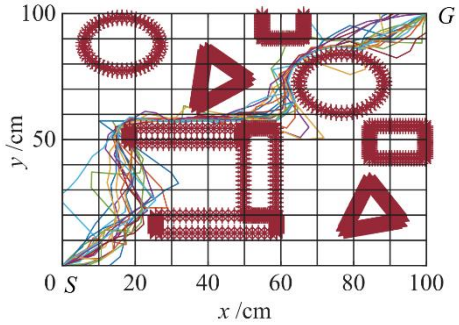
Fig.4 Contrast Diagram of Convergence Curves

According to the results shown in Fig. 4, among the three algorithms, the ant colony algorithm and the differential evolution algorithm have relatively high convergence target values (the convergence value is the difference between the algorithm optimization value and the real shortest path planning value), and the proposed algorithm can obtain experimental convergence results that are close to 0, indicating that the proposed algorithm can acquire relatively better path planning results in robot path planning. The experimental results have verified the effectiveness of the proposed algorithm.

Scenario 2: Select more complex experimental scenarios, which contain triangular obstacles, U-shaped obstacles, and circular obstacles. The comparison algorithm selects the robot path planning results of the standard differential evolution algorithm. The experimental results select the results of 20 times to present the path optimization trajectory through the visualization method. According to the planning results (Figs.5a ~ Fig.5b), the robot path planning results of the proposed algorithm are obviously better than those of the standard differential evolution algorithm.



(a) Standard difference evolution algorithm



(b) Results of the proposed algorithm

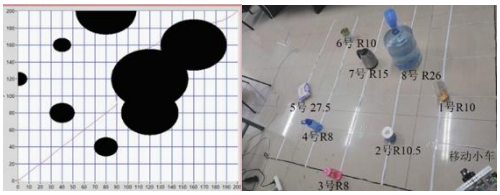
Fig.5 Comparison of scenario 2 path planning results

It can be seen from the experimental results of Figs. 5a to 5b that in the comparison of the experimental results of Scenario 2, almost every path planning result of the proposed algorithm can perfectly avoid the set obstacles, and there is a relatively more ideal path planning distance, reflecting the advantages of the algorithm path planning results. There are multiple path planning failures and multiple path crossings with obstacles in the comparison algorithm, revealing that the robot has collided with obstacles in the process and the path planning process of the algorithm fails.

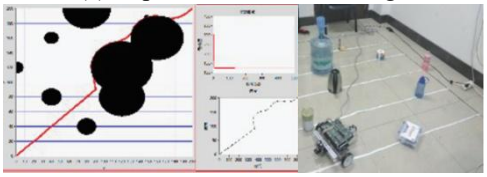
C. System development experiment

In this experiment, the Lab view2018 simulation platform is selected to simulate the robot path development experiment. The experimental robot Labview robotics starter kit2.0 is selected, and ultrasonic sensors detect

obstacles in the path planning process. The experimental scene selects the 5×5 grid test environment, with a resolution of $40\text{cm} \times 40\text{cm}$. The experimental scene and the ideal path planning results are shown in Fig. 6a, in which the sequence position of the robot motion process in the experimental scene is provided. The left image of Fig. 6a shows the mapping scene of the actual scene, and the right image presents the actual test environment. Fig.6b shows the experimental results of the proposed algorithm in the set robot path planning scene. The left figure of Fog. 6b shows the experimental simulation results and the right figure exhibits the experimental results of the robot passing through the path scene obstacle.



(a) Experimental scene settings



(b) Results of the algorithm in this paper

Fig.6 Experimental results of system development

To prevent the robot from colliding, its size radius is set to double. According to the results shown in Fig.6, under the premise of considering the radius of the robot, the proposed path planning algorithm has a higher path planning accuracy. The $\text{Labview robotics starter kit2.0}$ robot selected in this experiment can achieve collision-free obstacle avoidance motion according to the current planning results, which has a high degree of fit with the real ideal planning route. Therefore,, the proposed path planning algorithm has excellent path planning performance, has practical value for robot path planning, and can be used for actual robot path planning research.

The indoor robot path planning algorithm based on the improved differential evolution algorithm proposed in this paper has the following major contributions: ① The grid model is used to construct the model and simplify the indoor robot path planning problem into an optimization problem; ② The differential evolution algorithm is introduced, and the neighborhood perturbation process based on Q-learning is used to improve the performance of the algorithm. ③ The simulation platform of the indoor robot path planning problem is established by using the matlab software, which has research expansibility. The future study will mainly improve the computational efficiency of the algorithm: ① A parallel path optimization algorithm will be designed; ② The algorithm itself will be perfected to improve its convergence speed.

REFERENCES

[1] Zhang Juan, Mao Xiaobo, Chen Tiejun. Review of Research on Moving Target Tracking Algorithms [J]. Computer Application Research, 2009 (12): 4.

[2] Wei Wenming, Liu Weiping, Fan Zhipeng, et al. Variable step difference algorithm for calculating acceleration in autonomous vehicle testing [J]. Journal of Automotive Safety and Energy Conservation, 2023, 14 (4): 457-462

[3] Li Dan, Cui Wenfeng, Chen Guipeng. Exploration of Optimal Dynamic Response of SIMO Buck Converter Based on Differential Evolution Algorithm [J]. Journal of Beihang University, 2024.

[4] Wen Shangsheng, Qiu Zhiqiang, Xu Hanming, et al. Self resetting Particle Filter Algorithm Based on Differential Algorithm Optimization [J]. Journal of South China University of Technology: Natural Science Edition, 2023, 51 (3): 133-145.

[5] Hou Rongbo, Wei Wu, Huang Ting, et al. Indoor robot localization and 3D dense map construction based on ORB-SLAM [J]. Computer Applications, 2017, 37 (5): 6. DOI: 10.11772/j. issn.1001-9081.2017.05.1439

[6] He Shaojia, Liu Ziyang, Shi Jianqing. Indoor robot obstacle detection scheme based on monocular vision [J]. Computer Application, 2012, 032 (009): 2556-2559. DOI: 10.3724/SP.J.1087.2012.02556.

[7] Zhong K, Wang Y, Pei J, et al. Super efficiency SBM-DEA and neural network for performance evaluation[J]. Information Processing & Management, 2021, 58(6): 102728.

[8] Jan N, Gwak J, Pei J, et al. Analysis of networks and digital systems by using the novel technique based on complex fuzzy soft information[J]. IEEE Transactions on Consumer Electronics, 2022, 69(2): 183-193.

[9]Yu Z, Pei J, Zhu M, et al. Multi-attribute adaptive aggregation transformer for vehicle re-identification[J]. Information Processing & Management, 2022, 59(2): 102868.

[10] Li J, Li S, Cheng L, et al. BSAS: A Blockchain-Based Trustworthy and Privacy-Preserving Speed Advisory System[J]. IEEE Transactions on Vehicular Technology, 2022, 71(11): 11421-11430

[11] Liu Peng, Ren Gongchang. Fusion algorithm for indoor robot path planning based on feature maps [J]. Computer Science and Exploration, 2023, 17 (11): 2755-2766. DOI: 10.3778/j.issn.1673-9418.2207001