Received September 1, 2020, accepted September 5, 2020, date of publication September 14, 2020, date of current version September 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3023999

A Firefly Algorithm With Self-Adaptive Population Size for Global Path Planning of Mobile Robot

FENGLING LI, XINGJIANG FAN, AND ZHIXIANG HOU, (Member, IEEE)
College of Automotive and Mechanical Engineering, Changsha University of Science & Technology, Changsha 410114, China

Corresponding author: Fengling Li (flli4789@126.com)

This work was supported in part by the National Natural Science Foundation of China under Grant NSFC 51408069 and Grant NSFC 51678077, in part by the Key Laboratory of Lightweight and Reliability Technology for Engineering Vehicle, Education Department of Hunan Province (Changsha University of Science & Technology) under Grant 2013kfjj09, and in part by the Changsha University of Science and Technology Graduate Research Innovation Project under Grant CX2019SS13.

ABSTRACT In the simulation experiment of path planning of mobile robot based on firefly algorithm, it is found that the matching relationship between the number of fireflies and obstacles in the iterative process has significant conflict impacts on exploration ability and computational complexity of the algorithm. In order to solve the above problem, an optimal method of path planning based on firefly algorithm with self-adaptive population size is proposed. Firstly, the evaluation of degree of collision is established at the cost of avoiding collision. Based on the degree of collision of the population, two nonlinear functions are proposed to determine the population size. Then, individuals are added or deleted for the firefly population. Individuals are added randomly. The feasible solution and the infeasible solution are distinguished in firefly population, and delete the fireflies in the infeasible solution first when performing the eliminating operation. Finally, on the basis of the existing methods for dealing with infeasible paths, a coefficient that is adaptively adjusted according to the population size is introduced to control the degree to which the infeasible path approaches the feasible area. Compared with fixed population size firefly algorithm, the proposed algorithm has better performance in terms of solution stability, convergence speed and running time.

INDEX TERMS Path planning, improved firefly algorithm, self-adaptive population size.

I. INTRODUCTION

The goal of path planning is to search for a collision free path between the starting point and target point according to the specific task requirements. Optimization algorithm is the core of path planning problem. Existing optimization methods for path planning problems can be divided into two categories [1]: classical path planning methods and heuristic intelligent methods. Scholars have done a lot of research on path planning based on classical methods, for example, rapidly-exploring random tree (RRT) [2], cell decomposition (CD) [3], roadmap approach (RA) [4], artificial potential field (APF) [5], and probabilistic roadmap (PRM) [6], etc. The classical algorithm has the advantages of high computational efficiency and high real-time performance in dynamic path planning. Therefore, classical algorithms are widely used in path planning. At the same time, the classical path planning methods also exists that the searched path is not optimal or the complete path cannot be found, especially in complex environment.

Path planning problem is a typical Nondeterministic Polynomial (NP) problem, and heuristic algorithms have shown excellent optimization ability for NP problem compared with classical algorithms. Nowadays, heuristic algorithms have been widely applied in the field of path planning and have been improved according to the requirement of application scenarios. Shao *et al.* [7] proposed a three dimensional path planning algorithm for UAV formation based on comprehensively improved particle swarm optimization (PSO), where two key parameters of PSO were adaptively adjusted. Besides, a mutation strategy where undesired particles were replaced by those desired ones was also proposed. Aiming at the problem of UAV global path planning, Zhang *et al.* [8] proposed an improved constrained differential evolution (DE) algorithm. The simple ant colony optimization meta-heuristic (SACO-MH) was used to solve the problem of path planning

The associate editor coordinating the review of this manuscript and approving it for publication was Heng Wang.

for mobile robots by Garcia *et al.* [9]. In order to overcome the disadvantage of single algorithm, hybrid algorithm has also attracted the attention of scholars. Combining the advantages of ant colony system (ACS) and firefly algorithms (FA), Goel *et al.* [10] developed a hybrid algorithm based on ant colony system (ACS) and firefly algorithm (FA). The basic framework of the hybrid algorithm was provided by ACS, and FA was used to search the unexplored solution space. Existing improved heuristic algorithms can be roughly divided into the following categories: optimizing parameters adaptively in the heuristic model, and combining with different methods to improve the effect of path optimization. In heuristic algorithms and their improved algorithms, population size was usually considered as a fixed value.

With the further research of the heuristic algorithms, scholars have found that population size affects the exploration and exploitation ability of the algorithm, and then affects the efficiency of the heuristic algorithms [11], [12]. At the same time, the heuristic algorithms based on the variable population size strategy have been widely concerned by scholars. Piotrowski *et al.* [13] introduced various improvements of differential evolutionary algorithm in self-adaptation population size, and summarizes four kinds of self-adaptive population size strategies. The first is based on the self-adaptation of population size at individual levels. The second is to start with a large population size and gradually reduce it in operation. The third is that the adjustment of population size depends on the diversity of fitness values, or the complexity of the problem. The last one is to adjust the population size depending on whether the optimal solution is improved. Lanzarini *et al.* [14] introduced the concepts of particle life and neighbor in PSO algorithm, and assigned life to each particle according to the number of neighbors, where particles without lifetimes will be eliminated. In the differential evolutionary algorithm, Tanabe *et al.* [15] proposed the strategy of linear reduction of population size. This strategy can only eliminate individuals with poor fitness values and does not increase the number of individuals. Therefore, in this kind of adaptive strategy, the initial size of the population needs to be set to the largest possible value in order to ensure the exploration ability of the algorithm. Some researchers used diversity to adjust population size. According to the diversity of the population in the current ladder final time [16], the number of individuals is increased or decreased periodically in the form of ladder function. Zhu *et al.* [17] proposed the status monitor to observe the change of optimal solution. When the optimal solution is not improved for many times, the monitor triggers the population increase mechanism. Correspondingly, when the optimal solution is improved, the monitor triggers the population elimination mechanism. However, Wang *et al.* [18] hold the completely contradictory view with Zhu W. There has been a wealth of literature on adaptive population size of other heuristic algorithms, but there is still little literature on firefly algorithm. There are also few researches on adaptive population size heuristic algorithms with the characteristics of practical optimization problems such as path planning.

Some scholars [19]–[21] have explored the influence of population size on firefly algorithm through comparative experiments of multiple groups of different population sizes in specific optimization problems. It is worth noting that the population size in these literatures did not change adaptively during the operation of the algorithm. It can be concluded from these studies that as the population size increases, the exploration ability of the algorithm is improved. And the algorithm can also search for better solutions. In the process of path planning, fireflies in the population converge from the initial randomly generated path across obstacles to the optimal path. During this period, the exploration ability of firefly algorithm is the key to ensure that the algorithm can find the optimal path. Therefore, in order to ensure that the firefly algorithm can search the optimal solution, the population size is often set to a larger value. But blindly increasing the population size will also significantly increase the computational complexity of firefly algorithm [22]. With the convergence of solutions, there will be a large number of redundant individuals in the population to reduce the computational efficiency of firefly algorithm. At this point, reducing the population size is helpful to improve the computational efficiency of firefly algorithm. Therefore, adaptive population size is an important means to balance the optimization ability and computational efficiency of firefly algorithm.

The adaptive population size method of heuristic algorithm is to monitor the numerical change during algorithm optimization and adjust the population size according to the change strategy. The monitoring methods include the above-mentioned individual life, improved state of optimal solution and population diversity, *et al.* However, these methods do not take into account the characteristics of actual optimization problems. In path planning problem, fireflies in the population move from the initial path to the optimal path across the obstacle, and the degree of collision between path and obstacle also changes correspondingly during the moving process. Comparing with the above methods, it is more direct and effective to monitor the population state by evaluating the degree of collision of paths in the population. In this paper, the degree of collision between path and obstacle is used to monitor the population status. Two different nonlinear functions are established to determine the population size. The feasible path and collision path within the firefly population are distinguished when individual removal operation is performed, and the collision path is eliminated preferentially. In order to further improve the searching ability of firefly algorithm, the processing means of collision path is created during firefly movement.

The rest of the paper is organized as follows: Section II describes the path planning problem. The increasing/decreasing strategies of adaptive population size and population are introduced in Section III. In Section IV, an improved firefly algorithm considering collision path movement is introduced. Experimental comparisons and results discussion are conducted in Section V and Section VI. Section VII summarizes the article.

II. PATH PLANNING PROBLEM DESCRIPTION

A. COORDINATES TRANSFORMATION OF ENVIRONMENT MAP

The method proposed by reference [23] is adopted to model the environment map. The model of environment map is shown as Figure 1. $O-XY$ is the global coordinates. The rectangles and circle represent obstacles. Blue multistage line is an arbitrary path from start to end. The new map coordinate system $S-X'Y'$ is constructed to facilitate computation. The path starting point S is defined as the origin, the line connecting the path starting point and the target point is the X'-axis, and the vertical direction of the line is the Y'-axis. The new coordinates $S-X'Y'$ is established. The corresponding transformation relation is formulated as:

$$\begin{bmatrix} x\prime \\ y\prime \end{bmatrix} = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} x_s \\ y_s \end{bmatrix} \tag{1}$$

where ϕ is clockwise rotation angle from the line S-T to the X-axis, (x_s, y_s) is the value of point S, and $(x\prime, y\prime)$ is the value of point (x, y) in the new coordinate $S-X'Y'$.

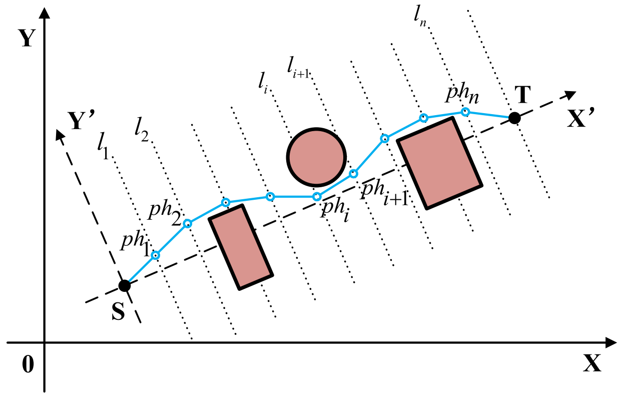

FIGURE 1. Modeling of environment map.

To reduce the computational complexity, line S-T is evenly divided into $n+1$ sections by lines $l_1, l_2, \cdots, l_n$ perpendicular to it. Then, the set $(ph_1, ph_2, \cdots, ph_n)$ can be obtained by sampling the lines $l_1, l_2, \cdots, l_n$. Hence, a complete path can be represented by the set of points PH. The initial path is generated by random sampling on lines $l_1, l_2, \cdots, l_n$.

$$PH = (S, ph_1, ph_2, \cdots, ph_n, T) \tag{2}$$

Therefore, the path planning problem is transformed into the n-dimensional coordinate optimization problem.

B. FITNESS FUNCTION AND EVALUATION OF DEGREE OF COLLISION

The fitness function of path planning consists of path length Length(PH) and the degree of collision of path Collis(PH) as follows:

$$f(PH) = \mu \text{Length}(PH) + (1-\mu)\text{Collis}(PH) \tag{3}$$

where μ is a weighting parameter between 0 and 1. Supposing that the start state S and the target state T are ph_0 and ph_{n+1}, the specific calculation formula of path length Length(PH) can be expressed as follows:

$$\text{Length}(PH) = \sum_{j=0}^{n} d(ph_j, ph_{j+1}) \tag{4}$$

where $d(ph_j, ph_{j+1})$ represents the length of the path segment $ph_j - ph_{j+1}$. Collis(PH) is calculated as follows:

$$\text{Collis}(PH) = \sum_{j=0}^{n} c(ph_j, ph_{j+1}) \tag{5}$$

where $c(ph_j, ph_{j+1})$ represents the degree of collision between path segment $ph_j - ph_{j+1}$ and all obstacles. In order to evaluate the degree of collision of the path segment, scholars often calculate the distance between the path segment and the center of the obstacle [24]. The degree of collision is related to the cost of path avoiding collision. In this paper, calculate the degree of collision from another angle. The specific calculation method is shown in Figure 2.

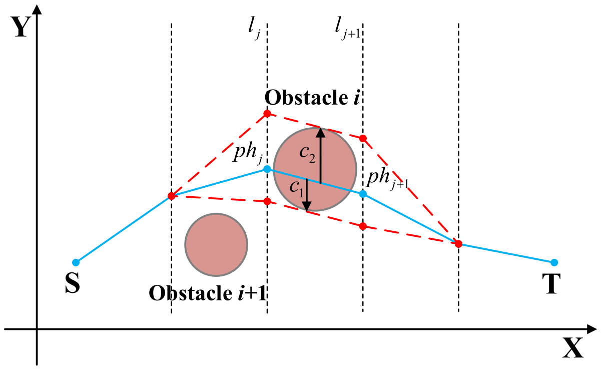

FIGURE 2. Evaluation method of the degree of collision.

As shown in Figure 2, the path segment $ph_j - ph_{j+1}$ of the blue path collides with the obstacle i. The red path represents the two safe paths of the path segment $ph_j - ph_{j+1}$ by moving up and down. c_2 and c_1 are the distances moved up and down, respectively. The degree of collision of the path segment $ph_j - ph_{j+1}$ with respect to obstacle i is represented by $p_{j,j+1}$, where $p_{j,j+1}(i) = \min(c_1, c_2)$. If the path segment does not collide with the obstacle, then $p_{j,j+1}(i) = 0$. The degree of collision $c(ph_j, ph_{j+1})$ between the path segment $ph_j - ph_{j+1}$ and all obstacles can be calculated as follows:

$$c(ph_j, ph_{j+1}) = \sum_{i=1}^{Nob} p_{j,j+1}(i) \tag{6}$$

where Nob is the number of obstacles.

III. SELF-ADAPTIVE POPULATION SIZE

In the algorithm proposed in this paper, the population size can change adaptively with the search of the algorithm. First, the method to determine the population size is introduced. Then the operation of adding or deleting individuals to the existing population is performed by comparing the current and determined population size. Finally, the methods of adding and removing individuals are described in detail respectively.

A. THE CHANGE STRATEGY OF POPULATION SIZE

According to the complexity of environment of robots, the dynamic adjustment of population size can reduce the computational cost of the algorithm and ensure the algorithm optimization performance. In the path planning problem, the optimal path to be searched should not collide with the obstacle. However, there are collisions in path optimization, as shown in Figure 3. In order to improve the ability of firefly algorithm to find feasible solutions, the degree of collision of population is an important factor to determine the population size. The mean value and standard deviation of the collision degree of all paths in the population are adopted to evaluate the collision degree of the population. The specific evaluation methods are as follows:

$$E(P) = k^{*}\text{mean}(\text{Collis}(P)) + (1-k)^{*}\text{std}(\text{Collis}(P)) \qquad (7)$$

where P is the firefly population, in the global path planning method based on firefly algorithm, a firefly represents a complete path, so P includes all paths; k is the weighted coefficient between the mean value mean(Collis(P)) and standard deviation std(Collis(P)) of the degree of collision of path, the value range of k is [0,1]. The calculation method of Collis(P) is shown in (5).

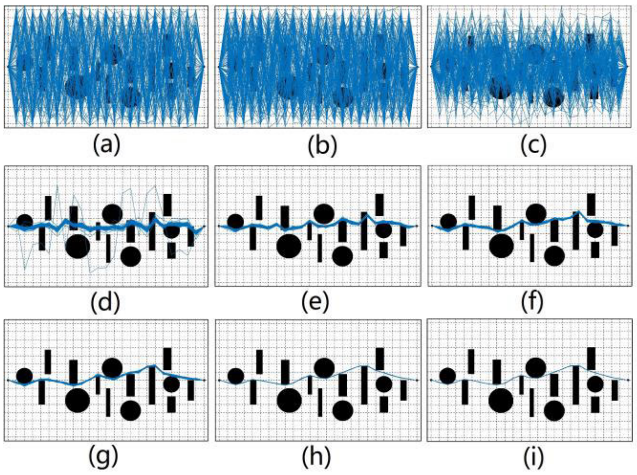

FIGURE 3. The iterative process of the firefly algorithm. (a) The initial path. (b) The number of iterations is 5. (c) The number of iterations is 9. (d) The number of iterations is 12. (e) The number of iterations is 15. (f) The number of iterations is 20. (g) The number of iterations is 30. (h) The number of iterations is 60. (i) The number of iterations is 100.

In path planning based on firefly algorithm, the initial path is usually randomly generated. With the iteration of the algorithm, the path will cross the obstacle and converge to the optimal path. In the course of crossing the obstacle, the degree of collision of path will obviously increase, as shown in Figure 4. Therefore, to ensure that the path can cross obstacles

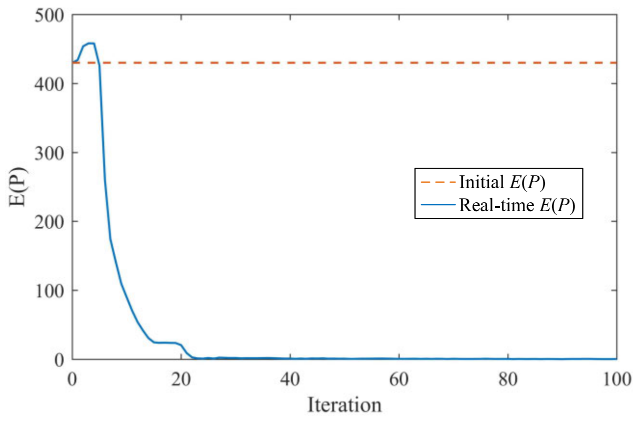

FIGURE 4. Population evaluation (k = 0.5).

and converge to the optimal path, it is necessary to increase the population size to improve the exploration capability of the firefly algorithm.

As shown in Figure 4, the initial $E(P)$ (notified as E_{init}) is the evaluation value of the initial population, and population evaluation $E(P)$ is divided into two parts by E_{init}. The two parts perform two different change strategies: up change strategy and down change strategy. The total change rules are described as (8), and the specific change strategies are described in detail in the following chapters..

$$\begin{cases} \text{Up change strategy} & , if\ E(P) > E_{\text{init}} \\ \text{Down change strategy} & , if\ E(P) \le E_{\text{init}} \end{cases} \qquad (8)$$

1) UP CHANGE STRATEGY

Up change strategy is adopted when $E(P) > E_{\text{init}}$. In up change strategy, the goal of changing the population size is to increase the diversity of the population and improve the solving ability of the FA, so the population size will only increase but not decrease. The number of population size increases according to the following formula:

$$PS(g+1) = PS(g) + \frac{\omega}{PS(g)} \qquad (9)$$

where $PS(g)$ is the current population size, $PS(g+1)$ is the changed population size and ω is the fixed coefficient. The number of individuals increased is determined by the second term of (9) and is related to the current population size. The number of individuals increases rapidly when the population size is small, similarly, only a small number of individuals will be added when the current population size is larger.

2) DOWN CHANGE STRATEGY

Down change strategy is adopted when $E(P) \le E_{\text{init}}$. In down change strategy, the population size adaptively changes with the size of the population evaluation value. And the population size after change is related to the current population assessment value. The nonlinear variation rules are as follows:

$$PS(g+1) = \delta^{*}E_{g}^{\eta}(P) \qquad (10)$$

where $E_g(P)$ is the population assessment value of the current state; η is fixed coefficient, and the value range is $0 < \eta < 1$; δ is calculated as follows:

$$\delta = \frac{PS_{\max}}{E_{\text{init}}^{\eta}} \tag{11}$$

where $PS_{\max}$ is the maximum population size so far, it is worth noting that $PS_{\max}$ may increase with the iteration of the algorithm. So δ is not a fixed constant. Different from up change strategy, in down change strategy, $PS(g+1)$ may increase or decrease compared with $PS(g)$, which is related to population evaluation value. The curve of $PS(g+1)$ changing with $E_g(P)$ is shown in Figure 5.

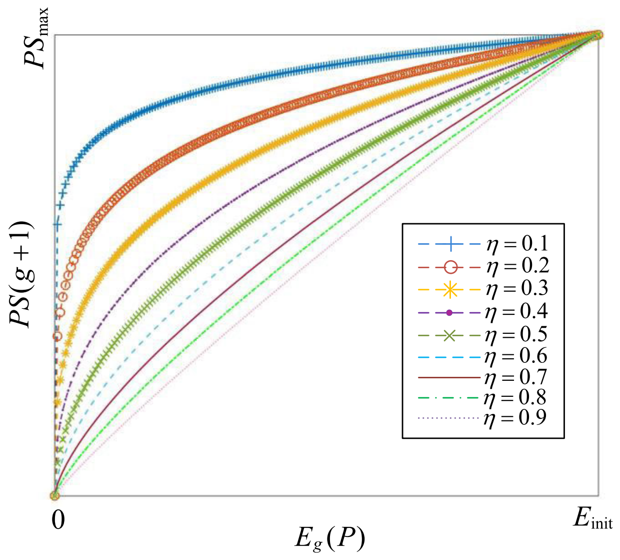

FIGURE 5. **The relationship between $PS(g+1)$ and $E_g(P)$.**

As shown in Figure 5, when $E_g(P)$ is near E_{init}, $PS(g+1)$ has a gentle change relation with $E_g(P)$, and when $E_g(P)$ is near 0, $PS(g+1)$ will rapidly decrease. With the increase of parameter η, the change relationship between $PS(g+1)$ and $E_g(P)$ tends to be linear.

B. INCREASING IMPLEMENTATION METHOD OF INDIVIDUAL

When the population size $PS(g+1)$ after changing strategy is larger than the current population size $PS(g)$, new individuals need to be added to the population. The number of individuals added is $\Delta PS = abs(PS(g+1) - PS(g))$. The purpose of expanding the population size is to improve the population richness so as to enhance the solving performance of the firefly algorithm. New population individuals are randomly generated according to the upper and lower boundaries of the current population. The generation method of individual x_i is shown below:

$$x_i = x^d + (x^u - x^d) * rand \tag{12}$$

where x^u is the upper boundary of the current population, x^d is the lower boundary; *rand* is the random number between [0,1] and the data dimension is consistent with x_i.

C. DELETION IMPLEMENTATION METHOD OF INDIVIDUAL

When the population size $PS(g+1)$ is smaller than $PS(g)$, the individuals in the population need to be removed. The number of reduced individuals is ΔPS, where $\Delta PS = abs(PS(g+1) - PS(g))$. The aim of population reduction is to reduce the computational complexity by eliminating the individuals with poor performance in the population. Individuals in the population are divided into feasible path sets and infeasible path sets according to whether they collide with obstacles or not. Individuals in an infeasible path set will be preferentially eliminated. The population decreasing method is shown in Figure 6.

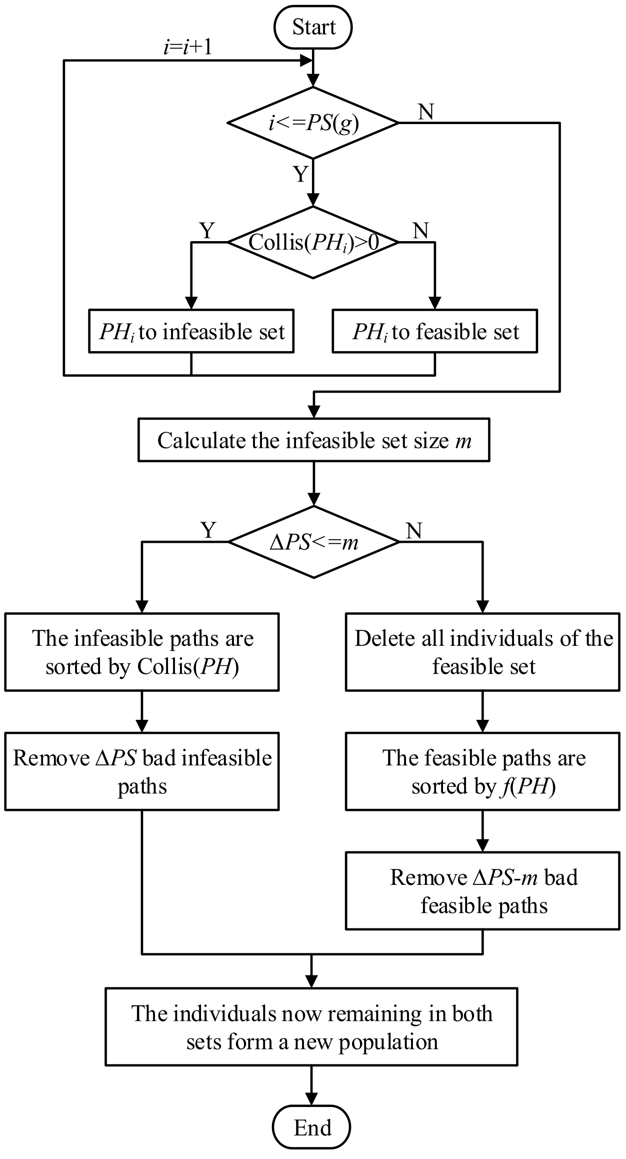
Start
i=i+1
i<=PS(g)
N
Y
Collis(PH_i)>0
Y
N
PH_i to infeasible set
PH_i to feasible set
Calculate the infeasible set size m
ΔPS<=m
Y
N
The infeasible paths are sorted by Collis(PH)
Delete all individuals of the feasible set
Remove ΔPS bad infeasible paths
The feasible paths are sorted by f(PH)
Remove ΔPS-m bad feasible paths
The individuals now remaining in both sets form a new population
End

FIGURE 6. **Individual deletion method framework.**

As shown in Figure 6, the degree of collision is given priority in the elimination of the individual. If the number of individuals in the infeasible path set is less than the number (ΔPS) of individuals to be eliminated, the strategy will also delete individuals in the feasible path set with a higher fitness value. Complete the goal of eliminating a specified number of individuals.

IV. IMPROVED FIREFLY ALGORITHM

A. FIREFLY ALGORITHM

Firefly algorithm was proposed by Yang X S [25]. The FA was inspired by the flashing activity of fireflies and is utilized in optimization. Each member (firefly) in the population represents a candidate solution in the search space. Fireflies move toward other positions and find potential candidate solutions. The attractiveness is determined by the intensity of the emitted light, which is usually measured by the fitness value.

Let X_i be the *ith* firefly in the population. The attractiveness between two fireflies X_i and X_j can be calculated as follows:

$$\beta(\mathrm{r}_{ij}) = \beta_0 e^{-\gamma r_{ij}^2} \tag{13}$$

where the parameter β_0 denotes the attractiveness at the distance $\mathrm{r} = 0$, and γ is the light absorption coefficient; r_{ij} is the distance between X_i and X_j, The specific calculation formula is as follows:

$$\mathrm{r}_{ij} = \left\| X_i - X_j \right\| = \sqrt{\sum_{d=1}^{n} (x_{id} - x_{jd})^2} \tag{14}$$

Firefly X_i is compared with all other fireflies X_j. If X_j is brighter (better) than X_i, X_i will be attracted to and move toward X_j. The movement of X_i can be defined by the following formula:

$$x_i(t+1) = x_i(t) + \beta_0 e^{-\gamma r_{ij}^2}(x_j(t) - x_i(t)) + \alpha(rand - 1/2) \tag{15}$$

The main steps of the FA are described in Algorithm 1, where n is the population size. Light intensity I_i at X_i is determined by fitness function $f(X_i)$.

Algorithm 1 Firefly Algorithm

```
1  Set algorithm parameters and population size;
2  Randomly initialize the population and compute the light
   intensity of each firefly;
3  while ($g < g_{\max}$)
4    for $i = 1$: $n$ all $n$ fireflies
5      for $j = 1$: $n$ all $n$ fireflies
6        if ($I_j > I_i$)
7          Move firefly $i$ towards $j$ according to (15);
8          Evaluate new solutions and update light intensity;
9        end if
10     end for $j$
11   end for $i$
12 end while
13 Output optimization results.
```

B. THE TREATMENT OF INFEASIBLE PATH

In a real-world application, the path planning workspace contains many obstacles. If no collision algorithm is adopted, the path will inevitably collide with the obstacle in the search process. Although the collision path can be moved to the feasible region depending on the searching ability of the algorithm itself, it requires many iterations of the algorithm and is a slow process. Therefore, some methods [26], [27] are proposed to move the collision path to the feasible region. All of these processing methods speed up the convergence of the algorithm by moving the collision path to the outside of the obstacle in one operation, and also reduce the infeasible paths in the population rapidly. But the infeasible path is helpful to improve the diversity of the population and enhance the exploration ability of the algorithm [28], especially in the early search process of the algorithm. On the other hand, the infeasible path can be used as a bridge to explore isolated feasible areas, especially if the feasible areas are relatively small. In order to solve the contradiction between eliminating infeasible path quickly and preserving infeasible path to improve population diversity. On the basis of the existing infeasible path treatment methods, this paper controls the external moving distance of the infeasible radial obstacle to balance this contradiction. As shown in Figure 7.

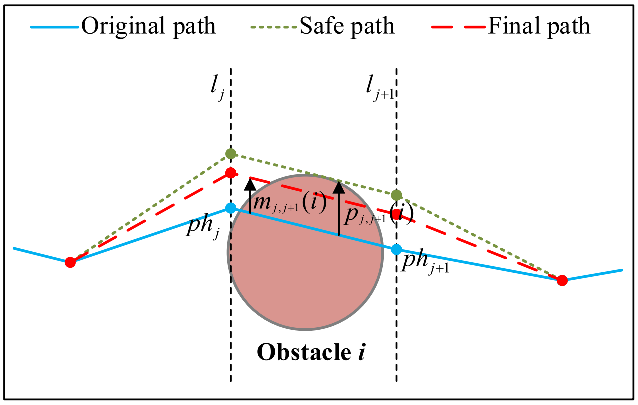

FIGURE 7. Operation method for the infeasible path.

In Figure 7, path segment $ph_j - ph_{j+1}$ collides with the obstacle. Where $p_{j,j+1}(i)$ is the distance between path segment $ph_j - ph_{j+1}$ and safe path; $m_{j,j+1}(i)$ is the actual parallel motion distance of collision path segment $ph_j - ph_{j+1}$. $m_{j,j+1}(i)$ and $p_{j,j+1}(i)$ satisfy the following relation:

$$m_{j,j+1}(i) = \varphi p_{j,j+1}(i) \tag{16}$$

the calculation method of $p_{j,j+1}(i)$ is shown in section II. Where φ is the degree to which the original path is close to the safe path, and its value range is [0,1]. When φ is large, the infeasible path can move quickly to the feasible region. At the same time, the diversity of the population is reduced and the algorithm is easy to fall into the local optimal state. On the other hand, when φ is small, the algorithm needs to spend more iteration times to search feasible paths. In order to obtain better performance, the value of parameter φ combined with the population size will be adjusted adaptively during the algorithm iteration. The variation rules are as follows:

$$\varphi(g) = \frac{2PS_{\max}}{PS^2(g)} \tag{17}$$

when the population size is large, the purpose is to increase the diversity of the population to enhance the exploration ability of the algorithm. In this case, parameter φ should be set as a small value to reduce the damage to the population diversity. However, when the population size is small, increasing φ can speed up the processing of infeasible paths and improve the development ability of the algorithm. Parameter φ can only change within the limited range [0,1].

In order to further reduce the influence of the treatment of infeasible path on the algorithm's exploration ability, the treatment of infeasible path is only implemented after the population is close to convergence. When the algorithm approaches the convergence state, the standard deviation std(Collis(P)) of the collision degree of the whole population will decrease. A threshold value T_{low} is set in the algorithm to determine whether to deal with the infeasible path. Operations on infeasible paths are allowed to run in the algorithm when std(Collis(P)) $\leq T_{\text{low}}$.

C. IMPLEMENTATION OF THE PROPOSED ALGORITHM

Based on the above research on the adaptive population size and the processing of infeasible paths, the general steps of the proposed path planning algorithm are described as algorithm 2.

In the proposed algorithm, the infeasible path processing strategy is implemented after the path is moved. The change of population size and the determination of parameter φ to control the infeasible path movement are performed after the algorithm completes an iteration. It is worth noting that parameter φ determines the value according to the population size $PS(g+1)$ after the change.

Algorithm 2 Proposed Algorithm

```
1 Set algorithm parameters and initial population size;
2 Randomly initialize the population and compute the light
   intensity of each firefly;
3 Calculate the initial population evaluation value $E_{\text{init}}$;
4 while ($g < g_{\max}$)
5    for $i = 1$: $PS(g)$ all fireflies
6       for $j = 1$: $PS(g)$ all fireflies
7          if ($I_j > I_i$)
8             Move firefly $i$ towards $j$ according to (15);
9             If allowed, perform treatment of infeasible path;
10            Evaluate new solutions and update light intensity;
11         end if
12      end for $j$
13   end for $i$
14 Calculate the population evaluation value $E(P)$;
15 Determine population size $PS(g+1)$ according to
      population change strategy;
16 Perform an individual add or delete operation;
17 if (std(Collis($P$)) $\leq T_{\text{low}}$)
18    Allows treatment of infeasible path;
19    The parameter $\varphi(g+1)$ is determined by population
      size $PS(g+1)$;
20 else
21    Disables treatment of infeasible path;
22 end
23 end while
24 Output optimization results.
```

V. EXPERIMENT

In order to verify the effectiveness of the proposed algorithm in this paper, three complex test scenarios were set up, with five algorithms in each scenario for comparison, and all experiments were independently repeated for 20 times. In the last part of this section, the parameters of the algorithm are studied. All algorithms have been coded on Matlab 16.0 and the simulations were done on AMD 3500U CPU 2.1 GHz laptop. The three complex test scenarios are: circular obstacles scenario, rectangular obstacles scenario, and mixed obstacles scenario. In all test scenarios, it is assumed that the mobile robot moves in a 1000×600 workspace with the starting point coordinate (20,300) and the target point coordinate (980,300). And the actual size of the mobile robot is ignored. In three scenarios, the symbol descriptions of all of the five algorithms are shown in Table 1.

TABLE 1. The symbol description of different algorithms.

Symbol	Introduction
FA-60	Classical firefly algorithm and the population size is 60
FA-80	Classical firefly algorithm and the population size is 80
FA-100	Classical firefly algorithm and the population size is 100
SPSFA	An self-adaptive population size firefly algorithm is proposed in this paper
SPSFA+TIP	On the basis of SPSFA, an improved treatment of infeasible path is added

In the three experimental scenes, the parameters of each algorithm were not changed with the changes of the scenes. The following parameters will be used in all algorithms: Maximum iteration number $g_{\max} = 100$, $\gamma = 0.000005$, $\beta_0 = 0.1$ and $\alpha = 2$. The following parameters are also required in SPSFA and SPSFA+TIP: fitness function weight coefficient $\mu = 0.1$, population evaluation weight coefficient $k = 0.1$, up change strategy coefficient $\omega = 1000$, down change strategy coefficient $\eta = 0.2$, and threshold $T_{\text{low}} = 20$ for opening treatment of infeasible path.

A. CASE 1: CIRCLE OBSTACLES SCENARIO

In this case study, an environment with 14 circular obstacles is utilized to demonstrate the effectiveness of the proposed

TABLE 2. Obstacles information in the CASE 1.

NO.	Center	Radius	NO.	Center	Radius
1	100, 250	40	8	500, 320	80
2	200, 320	50	9	580, 200	40
3	200, 450	60	10	640, 400	60
4	220, 200	40	11	680, 280	50
5	360, 400	60	12	780, 200	50
6	440, 180	40	13	780, 360	50
7	340, 260	50	14	860, 280	40

path planning algorithm for mobile robots. Table 2 describes the center and radius of all circular obstacles. The convergence curves of the five algorithms in CASE 1 are shown in Figure 8. The population size change curves of the two proposed adaptive population size algorithms in CASE 1 are shown in Figure 9. All the five comparison algorithms run independently for 20 times. Figure 10 shows the optimal path obtained by the algorithms. The statistical results of optimization solutions and running time are shown in Table 3 and Table 4 respectively, where Figure 11 shows the results of the run as a Box-plot.

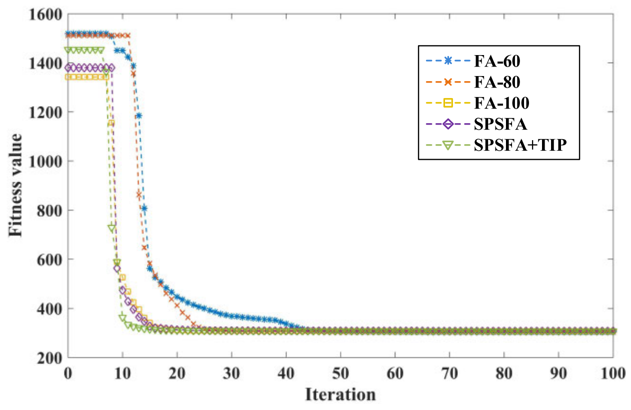

FIGURE 8. The convergence curves in the CASE 1.

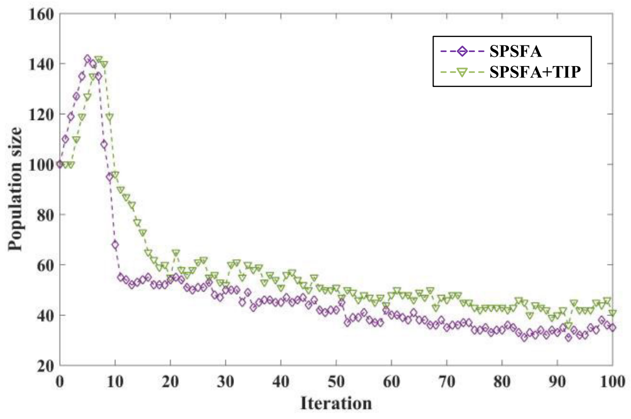

FIGURE 9. The population size change curves in the CASE 1.

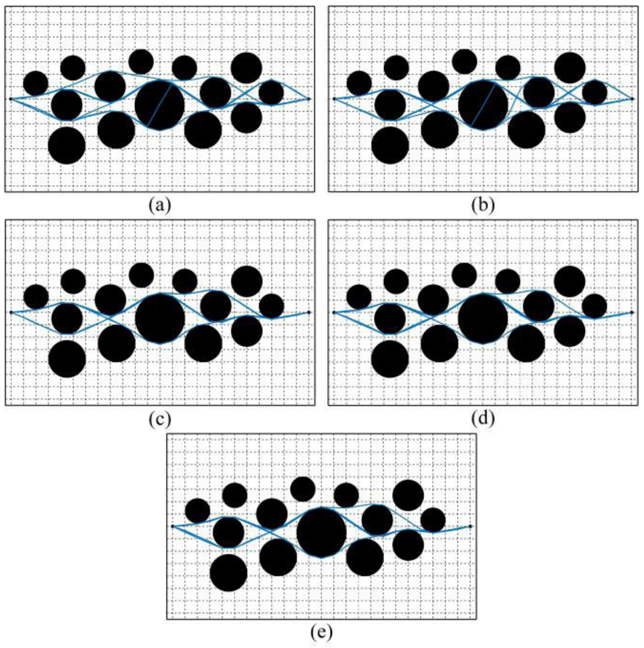

FIGURE 10. Optimized paths in the CASE 1. (a) FA-60. (b) FA-80. (c) FA-100. (d) SPSFA. (e) SPSFA-TIP.

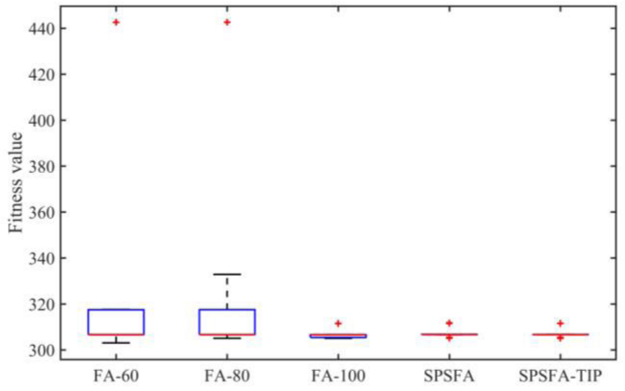

FIGURE 11. Optimal solutions boxplot graphs in the CASE 1.

TABLE 3. The statistical results of the optimal solutions in the CASE 1.

Algorithm	Maximum	Minimum	Average	Standard Deviation
FA-60	442.6158	**303.0075**	316.0910	29.4427
FA-80	442.5918	305.0318	318.0346	29.4368
FA-100	**311.4962**	305.0269	**306.6842**	**1.7087**
SPSFA	311.6977	305.1202	307.2759	1.8688
SPSFA-TIP	311.5351	305.0436	307.1524	1.9107

TABLE 4. The running time in the CASE 1 (in s).

Algorithm	Maximum	Minimum	Average	Standard Deviation
FA-60	**171.6168**	158.3337	164.6528	**4.8313**
FA-80	308.4826	281.5166	292.4876	10.4160
FA-100	476.8535	442.4749	458.6832	13.5196
SPSFA	195.8751	**106.4820**	**146.4548**	25.1090
SPSFA-TIP	248.9887	132.2167	177.1657	35.0899

B. CASE 2: RECTANGULAR OBSTACLES SCENARIO

In this case study, there are 15 rectangular obstacles in the scene where the mobile robot is located. Table 5 describes the four vertex coordinates of all rectangular obstacles. The convergence curves of the five algorithms are shown in Figure 12. The population size change curves of the

TABLE 5. Obstacles information in the CASE 2.

NO.	Point-1	Point-2	Point-3	Point-4
1	100, 260	130, 260	130, 380	100, 380
2	190, 220	220, 220	220, 300	190, 300
3	180, 320	200, 320	200, 400	180, 400
4	270, 150	300, 150	300, 260	270, 260
5	270, 280	300, 280	300, 320	270, 320
6	270, 340	300, 340	300, 460	270, 460
7	380, 220	420, 220	420, 380	380, 380
8	490, 120	520, 120	520, 260	490, 260
9	500, 290	520, 290	520, 420	500, 420
10	590, 340	620, 340	620, 480	590, 480
11	600, 170	640, 170	640, 300	600, 300
12	710, 230	740, 230	740, 420	710, 420
13	820, 180	840, 180	840, 290	820, 290
14	800, 340	840, 340	840, 460	800, 460
15	900, 280	930, 280	930, 380	900, 380

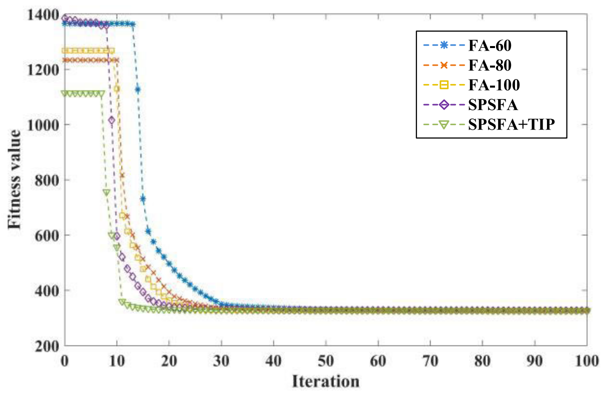

FIGURE 12. The convergence curves in the CASE 2.

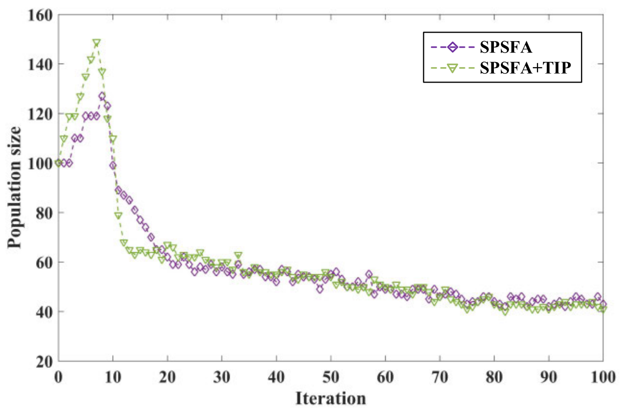

FIGURE 13. The population size change curves in the CASE 2.

two proposed adaptive population size algorithms are shown in Figure 13. And Figure 14 shows the optimal path obtained by all algorithms through 20 independent runs. The statistical results of optimization solutions and running time are shown in Table 6 and Table 7 respectively, where Figure 15 shows the results of the run as a Box-plot.

C. CASE 3: MIXED OBSTACLES SCENARIO

In the Case 3, the effectiveness of the proposed algorithm is verified in a mixed environment with 5 circular obstacles and 10 rectangular obstacles. Table 8 and 9 respectively describe

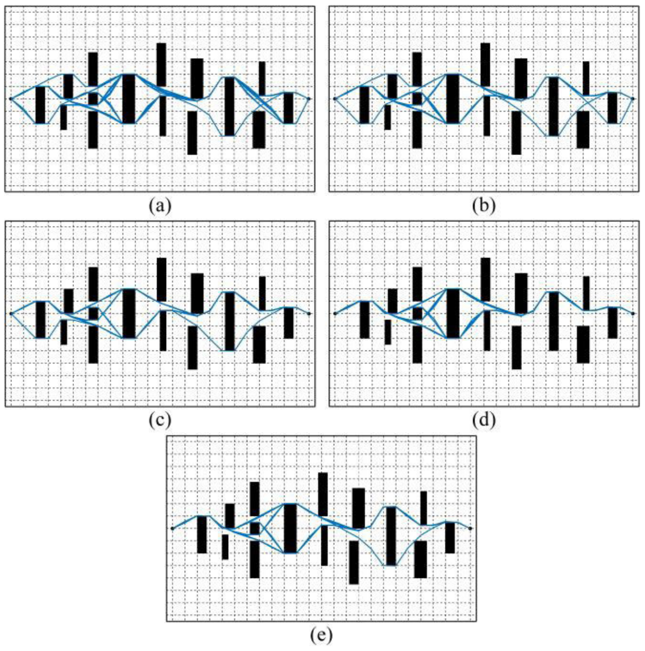

FIGURE 14. Optimized paths in the CASE 2. (a) FA-60. (b) FA-80. (c) FA-100. (d) SPSFA. (e) SPSFA-TIP.

TABLE 6. The statistical results of the optimal solutions in the CASE 2.

Algorithm	Maximum	Minimum	Average	Standard Deviation
FA-60	363.4167	326.3189	343.7111	10.2863
FA-80	381.7140	326.2476	337.2823	11.7255
FA-100	357.0749	**326.1359**	336.3130	7.8811
SPSFA	**344.3411**	326.2554	334.1019	5.7965
SPSFA-TIP	343.7910	326.2723	**333.8889**	**5.5564**

TABLE 7. The running time in the CASE 2 (in s).

Algorithm	Maximum	Minimum	Average	Standard Deviation
FA-60	**185.7692**	168.3544	**175.9578**	**6.3928**
FA-80	330.6995	299.2875	312.2085	10.8351
FA-100	514.4831	468.5175	486.6934	14.7624
SPSFA	285.3812	**138.6407**	220.1808	48.9066
SPSFA-TIP	348.4017	144.7955	233.4750	71.1037

TABLE 8. Circular obstacles information in the CASE 3.

NO.	Center	Radius	NO.	Center	Radius
1	100, 280	40	4	620, 450	50
2	360, 400	60	5	820, 320	40
3	530, 240	50			

the position information of the circular obstacles and the rectangular obstacles. The convergence curves of the five algorithms in Case 3 are shown in Figure 16. The population size change curves of the proposed adaptive population size algorithms are shown in Figure 17. All the five comparison algorithms run independently for 20 times. Figure 18 shows the optimal paths obtained by the algorithms. The statistical results of optimization solutions and running time are shown

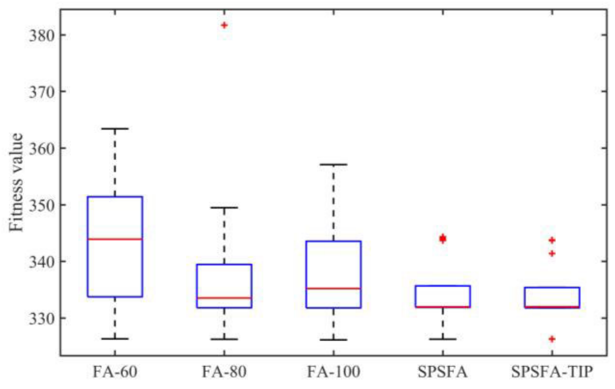

FIGURE 15. Optimal solutions boxplot graphs in the CASE 2.

TABLE 9. Rectangular obstacles information in the CASE 3.

NO.	Point-1	Point-2	Point-3	Point-4
1	200, 150	230, 150	230, 270	200, 270
2	170, 300	200, 300	200, 420	170, 420
3	320, 200	360, 200	360, 320	320, 320
4	500, 340	520, 340	520, 480	500, 480
5	450, 280	470, 280	470, 370	450, 370
6	600, 270	640, 270	640, 380	600, 380
7	710, 230	740, 230	740, 420	710, 420
8	780, 140	820, 140	820, 250	780, 250
9	800, 380	840, 380	840, 460	800, 460
10	900, 300	930, 300	930, 400	900, 400

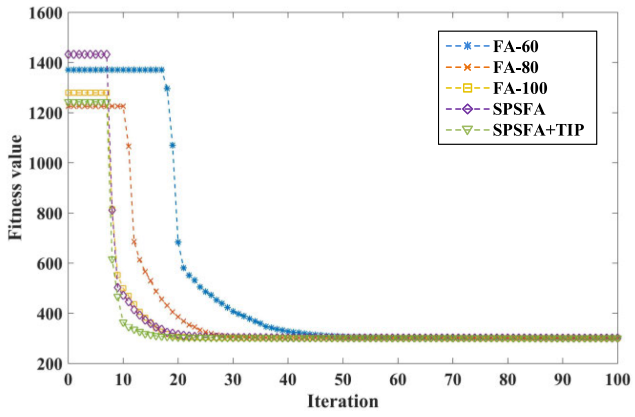

FIGURE 16. The convergence curves in the CASE 3.

TABLE 10. The statistical results of the optimal solutions in the CASE 3.

Algorithm	Maximum	Minimum	Average	Standard Deviation
FA-60	377.8497	301.0494	323.8432	21.5089
FA-80	368.1436	300.9645	319.7457	22.8871
FA-100	371.5501	300.9717	320.5980	22.4212
SPSFA	341.3666	301.0223	306.4953	9.2023
SPSFA-TIP	**331.3199**	**300.9588**	**305.4635**	**7.9265**

in Table 10 and Table 11 respectively, where Figure 19 shows the results of the run as a Box-plot.

D. PARAMETERS STUDY

Compared with the firefly algorithm with fixed population size, the proposed algorithm adds several new parameters. The impact of these parameters on the performance of the algorithm is explored in this section. In the experiment, CASE 3 is selected as the test scenario, and SPSFA-TIP is used as the test algorithm. The parameters of the algorithm are consistent with the treatment described above except for the parameters discussed.

1) RESEARCH ON PARAMETER ω

ω is the fixed coefficient in the up change strategy. In the experiment, eight different levels were selected from a range

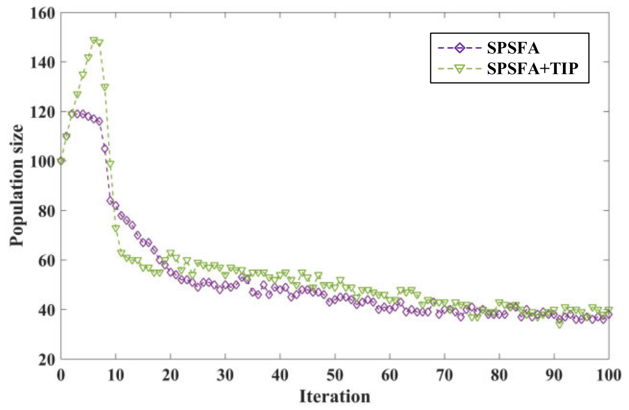

FIGURE 17. The population size change curves in the CASE 3.

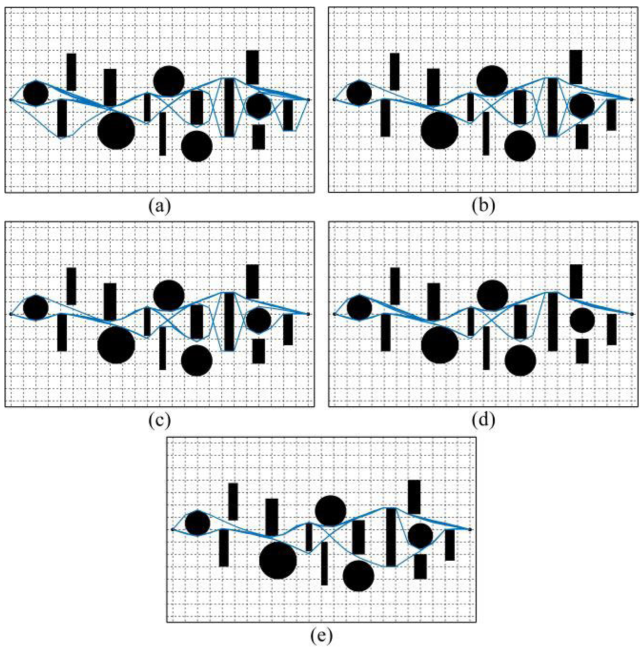

FIGURE 18. Optimized paths in the CASE 3. (a) FA-60. (b) FA-80. (c) FA-100. (d) SPSFA. (e) SPSFA-TIP.

TABLE 11. The running time in the CASE 3 (in s).

Algorithm	Maximum	Minimum	Average	Standard Deviation
FA-60	**183.1546**	167.5562	174.5719	**6.0650**
FA-80	325.2651	297.2246	310.9700	10.4816
FA-100	503.5106	466.3409	483.0614	14.9578
SPSFA	261.0559	**118.6097**	**172.2112**	41.2675
SPSFA-TIP	308.1163	125.6084	205.5591	50.8039

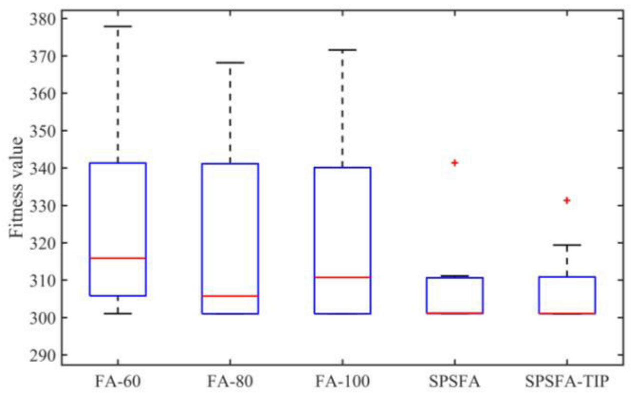

FIGURE 19. Optimal solutions boxplot graphs in the CASE 3.

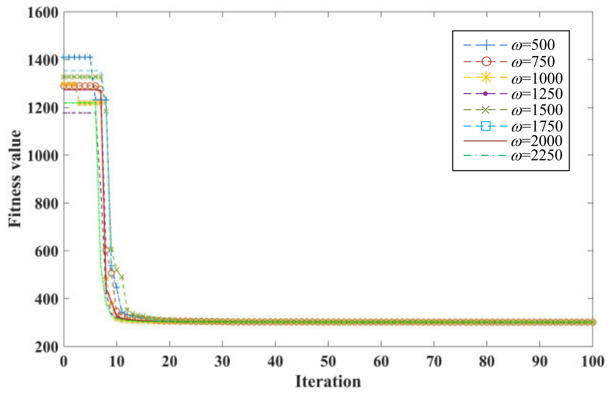

FIGURE 20. The convergence curves at different levels of parameter ω.

TABLE 12. The running time of the algorithm at different levels of parameter ω (in s).

$\omega = 500$	$\omega = 750$	$\omega = 1000$	$\omega = 1250$
204.7044	227.5110	283.8155	250.3793
$\omega = 1500$	$\omega = 1750$	$\omega = 2000$	$\omega = 2250$
191.6361	199.3538	438.0723	532.4390

of 500 to 2250, and the two adjacent levels differed by 250. Table 12 shows the running time of the algorithm. Figure 20 shows the convergence curve of the algorithm. The changes in population size are shown in Figure 21.

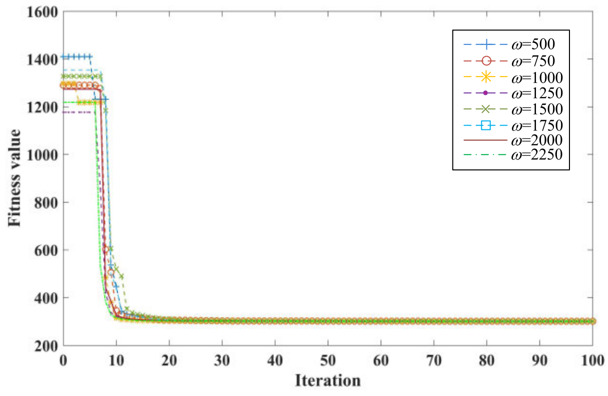

FIGURE 21. The population size at different levels of parameter ω.

It can be seen from Figure 20 that at different levels of ω, the algorithms all converge to similar positions. As shown in Figure 21, when ω is 2000 and 2250, the population size of the algorithms in the iterative process is significantly larger than that of other comparison algorithms. At the same time, it can also be seen from Table 12 that when ω takes 2000 and 2250, the running time of the algorithms is obviously more than that of other comparison algorithms. Because the larger population size will increase the computational complexity of the algorithm and further increase the running time of the algorithm. The experimental results show that the solving ability of the proposed algorithm is not sensitive to the value of parameter ω. But the running time of the algorithm will be significantly increased ω if selects too large value.

2) RESEARCH ON PARAMETER η

The coefficient η in the down change strategy affects the rate at which population size $PS(g+1)$ changes with population evaluation value $E_g(P)$. In order to study the influence of different values η of on the algorithm, 9 levels with equal intervals were selected in the experiment from 0.1 to 0.9. Table 13 shows the running time of the algorithm. Figure 22 shows the convergence curve of the algorithm. The changes in population size are shown in Figure 23.

TABLE 13. The running time of the algorithm at different levels of parameter η (in s).

η=0.1	η=0.2	η=0.3	η=0.4	η=0.5
807.7688	288.9072	94.4276	66.5299	61.6417
η=0.6	η=0.7	η=0.8	η=0.9	
65.5006	61.2663	51.2536	60.4777	

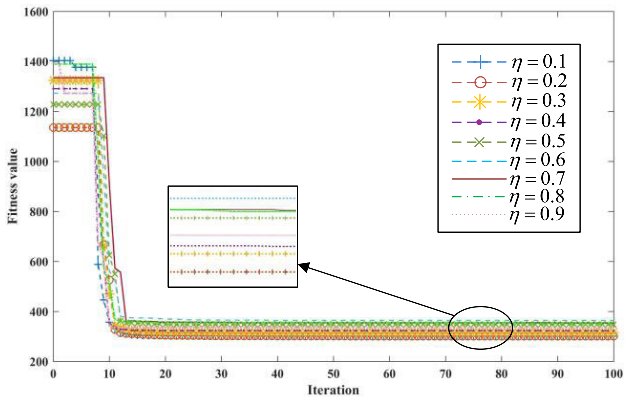

FIGURE 22. The convergence curves at different levels of parameter η.

As shown in Figure 22, when is set at 0.1 and 0.2, the algorithms can converge to the lowest position. However, it is known from Table 13 that the running time of the algorithm when is set at 0.2 is lower than that when is set at 0.1. It can be seen from Figure 23 that when takes the level between 0.3 and 0.9, the population size change curve of the algorithms does not show significant difference. When was set at 0.1 and 0.2, the population size was greater than that at other levels, and

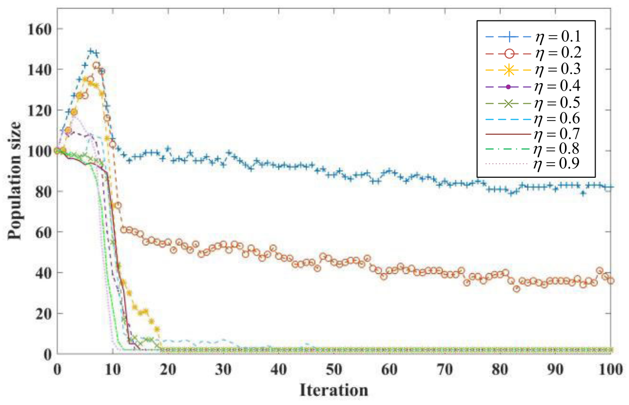

FIGURE 23. The population size at different levels of parameter η.

when was set at 0.1, the population size was the largest. When parameter is taken as a large value, it has no obvious influence on the algorithm. When parameter is taken as a small value, it is beneficial for the algorithm to converge to a lower position, and the running time of the algorithm will also increase correspondingly.

3) RESEARCH ON PARAMETER T_{low}

Threshold T_{low} is used to control the opening and closing of the treatment of infeasible path in the algorithm, and 10 levels are uniformly selected in the experiment, ranging from 10 to 100. Figure 24 shows the running time of the algorithm. Figure 25 shows the convergence curve of the algorithm.

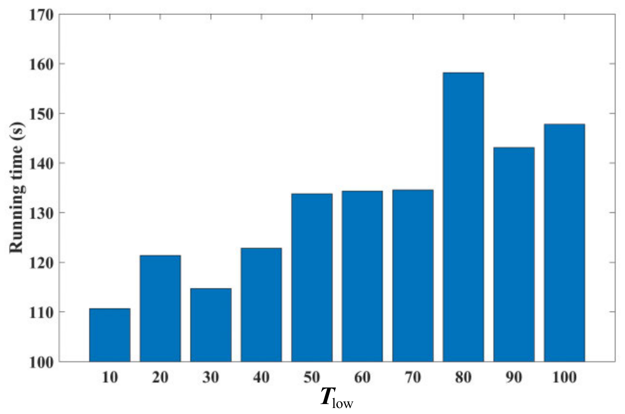

FIGURE 24. Running time at different levels of parameter T_{low}.

As can be seen from the convergence curve of the algorithm in Figure 25, when T_{low} is between 10 and 50, the algorithm can converge to a lower position, as shown in Figure 24. Although the running time of the algorithm fluctuates, it generally increases with the increase of the value of T_{low}.

VI. DISCUSSION

It can be seen from the convergence curves in Figure 8, Figure 12, and Figure 16 that the convergence speed increases with the increase of population size in the three firefly algorithms with fixed population size. However, compared with the two self-adaptive population size firefly algorithms

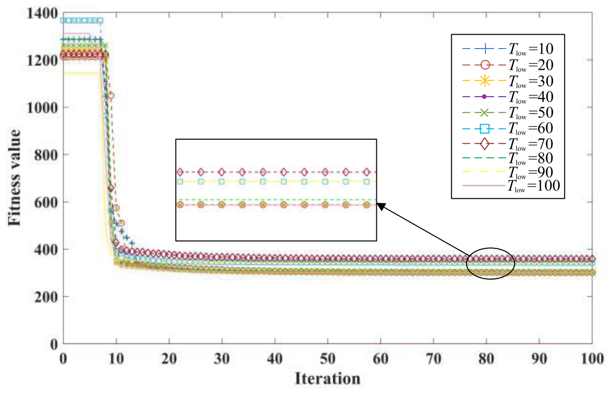

FIGURE 25. The convergence curves at different levels of parameter T_{low}.

(SPSFA and SPAFA-TIP) proposed in this paper, both SPSFA and SPAFA-TIP have faster convergence rates than the three fixed population size firefly algorithms. The convergence speed of SPSFA-TIP is the fastest among the five algorithms.

After repeated running for 20 times in the three scenarios, the minimum values of the optimal solution searched by the five algorithms are very close to each other, as can be seen from Table 3, Table 6, and Table 10. In terms of the mean and standard deviation of the optimal solution, the two algorithms proposed in this paper are superior to the three algorithms with fixed population size. This conclusion can also be verified in Figure 11, Figure 15, and Figure 19. After 20 repeated runs, the search results of the two proposed algorithms are more concentrated than those of the other three algorithms, indicating that the results of the proposed algorithm are more stable. In particular, in Case 1, the optimal paths searched by FA-60 and FA-80 both have serious collisions with obstacles, as shown in Figure 10.

It can be concluded from Table 4 and Table 11 that in Case 1 and Case 3, the two proposed algorithms are superior to the other three algorithms in terms of the average calculation time. As shown in Figure 9, Figure 13, and Figure 17, the proposed adaptive population size firefly algorithms remove redundant individuals from the population after population convergence and improve the computational efficiency of the algorithm. In Case 2, the average calculation time of FA-60 is the best, but SPSFA and SPSFA-TIP are still superior to FA-80 and FA-100. Compared with firefly algorithms with fixed population size, SPSFA and SPSFA-TIP are more unstable in terms of time spent in algorithm operation, but their average running time is still at a low level.

To sum up, compared with the fixed population-size firefly algorithm, the adaptive population size algorithm proposed in this paper has faster convergence rate, more stable solving ability and less running time.

VII. CONCLUSION

In this paper, a firefly algorithm with adaptive population size for path planning is proposed. In the process of searching, the paths inevitably collide with the obstacles. With the convergence of the algorithm, other paths will also cross the obstacles to approach the optimal path. Therefore, combined with the above characteristics of path planning, the population size is dynamically adjusted with the degree of collision of the firefly population, and two kinds of nonlinear functions are established to determine the population size to improve the optimization ability, convergence speed and operational efficiency of firefly algorithm. When the operation of increasing individuals is carried out, individuals are added to the population randomly to achieve the purpose of increasing the diversity of the population and improving the exploration ability of the algorithm. In an individual delete operation, the infeasible path is deleted first. Because the infeasible path has worse performance than the feasible path. Superior to the existing methods for dealing with infeasible paths, a coefficient which changes dynamically with population size is introduced to control the degree of infeasible path approaching to feasible path, so as to accelerate the convergence speed of the algorithm. Compared with the Firefly algorithm with fixed population size, the algorithm proposed in this paper has better performance in terms of convergence speed, stability of optimization results and average running time. In particular, an operation for the infeasible path is added on the basis of firefly algorithm with adaptive population size, which further accelerates the convergence speed of the proposed algorithm. However, the proposed algorithm is more volatile in running time than the firefly algorithm with fixed population size. In order to solve this challenge, in the future work, one research direction is to improve the running time stability of firefly algorithm with adaptive population size. In this paper, path planning in two-dimensional space is studied only. Therefore, another research focus is to explore the performance of the proposed algorithm in the path planning problem of three-dimensional space.

REFERENCES

[1] T. T. Mac, C. Copot, D. T. Tran, and R. De Keyser, "Heuristic approaches in robot path planning: A survey," *Robot. Auto. Syst.*, vol. 86, pp. 13–28, Dec. 2016.

[2] H. Zhang, Y. Wang, J. Zheng, and J. Yu, "Path planning of industrial robot based on improved RRT algorithm in complex environments," *IEEE Access*, vol. 6, pp. 53296–53306, 2018.

[3] F. Lingelbach, "Path planning using probabilistic cell decomposition," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 1, Jun. 2004, pp. 467–472.

[4] L. Lulu and A. Elnagar, "A comparative study between visibility-based roadmap path planning algorithms," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Jun. 2005, pp. 3263–3268.

[5] Q. Yao, Z. Zheng, L. Qi, H. Yuan, X. Guo, M. Zhao, Z. Liu, and T. Yang, "Path planning method with improved artificial potential field—A reinforcement learning perspective," *IEEE Access*, vol. 8, pp. 135513–135523, 2020.

[6] D. Kim, Y. Kwon, and S.-E. Yoon, "Dancing PRM*: Simultaneous planning of sampling and optimization with configuration free space approximation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 7071–7078.

[7] S. Shao, Y. Peng, C. He, and Y. Du, "Efficient path planning for UAV formation via comprehensively improved particle swarm optimization," *ISA Trans.*, vol. 97, pp. 415–430, Feb. 2020.

[8] X. Zhang and H. Duan, "An improved constrained differential evolution algorithm for unmanned aerial vehicle global route planning," *Appl. Soft Comput.*, vol. 26, pp. 270–284, Jan. 2015.

[9] M. A. P. Garcia, O. Montiel, O. Castillo, R. Sepúlveda, and P. Melin, "Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation," *Appl. Soft Comput.*, vol. 9, no. 3, pp. 1102–1110, Jun. 2009.

[10] R. Goel and R. Maini, "A hybrid of ant colony and firefly algorithms (HAFA) for solving vehicle routing problems," *J. Comput. Sci.*, vol. 25, pp. 28–37, Mar. 2018.

[11] X.-S. Yang and X. He, "Nature-inspired optimization algorithms in engineering: Overview and applications," in *Nature-Inspired Computation in Engineering* (Studies in Computational Intelligence). Cham, Switzerland: Springer, 2016, pp. 1–20.

[12] I. Fister, U. Mlakar, X.-S. Yang, and I. Fister, "Parameterless bat algorithm and its performance study," in *Nature-Inspired Computation in Engineering* (Studies in Computational Intelligence). Cham, Switzerland: Springer, 2016, pp. 267–276.

[13] A. P. Piotrowski, "Review of differential evolution population size," *Swarm Evol. Comput.*, vol. 32, pp. 1–24, Feb. 2017.

[14] L. Lanzarini, V. Leza, and A. De Giusti, "Particle Swarm Optimization with Variable Population Size," in *Artificial Intelligence and Soft Computing—ICAISC*. Berlin, Germany: Springer, 2008, pp. 438–449ss.

[15] R. Tanabe and A. S. Fukunaga, "Improving the search performance of SHADE using linear population size reduction," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2014, pp. 1658–1665.

[16] D. Chen and C. Zhao, "Particle swarm optimization with adaptive population size and its application," *Appl. Soft Comput.*, vol. 9, no. 1, pp. 39–48, Jan. 2009.

[17] W. Zhu, Y. Tang, J.-A. Fang, and W. Zhang, "Adaptive population tuning scheme for differential evolution," *Inf. Sci.*, vol. 223, pp. 164–191, Feb. 2013.

[18] X. Wang and S. Zhao, "Differential evolution algorithm with self-adaptive population resizing mechanism," *Math. Problems Eng.*, vol. 2013, pp. 1–14, Jan. 2013.

[19] T. Niknam, R. Azizipanah-Abarghooee, and A. Roosta, "Reserve constrained dynamic economic dispatch: A new fast self-adaptive modified firefly algorithm," *IEEE Syst. J.*, vol. 6, no. 4, pp. 635–646, Dec. 2012.

[20] K. G. Dhal, S. Sahoo, A. Das, and S. Das, "Effect of Population Size Over Parameter-less Firefly Algorithm," in *Applications of Firefly Algorithm and its Variants: Case Studies and New Developments*, N. Dey, Ed. Singapore: Springer, 2020, pp. 237–266.

[21] K. G. Dhal, M. Iqbal Quraishi, and S. Das, "Development of firefly algorithm via chaotic sequence and population diversity to enhance the image contrast," *Natural Comput.*, vol. 15, no. 2, pp. 307–318, Jun. 2016.

[22] H. Wang, W. Wang, X. Zhou, H. Sun, J. Zhao, X. Yu, and Z. Cui, "Firefly algorithm with neighborhood attraction," *Inf. Sci.*, vols. 382–383, pp. 374–387, Mar. 2017.

[23] C. Qu, W. Gai, J. Zhang, and M. Zhong, "A novel hybrid grey wolf optimizer algorithm for unmanned aerial vehicle (UAV) path planning," *Knowl.-Based Syst.*, vol. 194, Apr. 2020, Art. no. 105530.

[24] Y. Liu, X. Zhang, Y. Zhang, and X. Guan, "Collision free 4D path planning for multiple UAVs based on spatial refined voting mechanism and PSO approach," *Chin. J. sAeronaut.*, vol. 32, no. 6, pp. 1504–1519, Jun. 2019.

[25] X.-S. Yang, "Firefly algorithms for multimodal optimization," in *Stochastic Algorithms: Foundations and Applications*, vol. 9. Berlin, Germany: Springer, 2009, pp. 169–178.

[26] F. H. Ajeil, I. K. Ibraheem, M. A. Sahib, and A. J. Humaidi, "Multi-objective path planning of an autonomous mobile robot using hybrid PSO-MFB optimization algorithm," *Appl. Soft Comput.*, vol. 89, Apr. 2020, Art. no. 106076.

[27] C. Liu, Y. Zhao, F. Gao, and L. Liu, "Three-dimensional path planning method for autonomous underwater vehicle based on modified firefly algorithm," *Math. Problems Eng.*, vol. 2015, pp. 1–10, Jan. 2015.

[28] Y. Zhang, D.-W. Gong, and J.-H. Zhang, "Robot path planning in uncertain environment using multi-objective particle swarm optimization," *Neurocomputing*, vol. 103, pp. 172–185, Mar. 2013.

FENGLING LI was born in Hunan, China, in 1974. She received the M.Sc. and Ph.D. degrees in control theory and control engineering from Central South University, Changsha, China, in 2009. She is currently a Full Associate Professor with the College of Automobile and Mechanical Engineering, Changsha University of Science & Technology, China. Her scientific interests include intelligent instrument system design and robot motion planning. She is the Director of Hunan Instrument Association.

XINGJIANG FAN received the bachelor's degree in mechanical design, manufacturing, and automation from the Changsha University of Science & Technology, in 2014, where he is currently pursuing the master's degree in mechanical engineering. His research interests include motion control for mobile robots and intelligent optimization algorithms.

ZHIXIANG HOU (Member, IEEE) received the master's degree in control theory and control engineering from the Changsha University of Science & Technology, China, in 2002, and the Ph.D. degree in control theory and control engineering from the Central South University, in 2006. He is currently a Full Professor with the College of Automobile and Mechanical Engineering, Changsha University of Science & Technology. His research interests include robot control and intelligent algorithm.