IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 5 December 2022, accepted 26 December 2022, date of publication 4 January 2023, date of current version 9 January 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3234057

RESEARCH ARTICLE

An Improved Spherical Vector and Truncated Mean Stabilization Based Bat Algorithm for UAV Path Planning

BUQIAN CHEN[1], JIN YANG[1], HUIZHEN ZHANG[2], AND MENG YANG[1]
[1]School of Science, University of Shanghai for Science and Technology, Shanghai 200093, China
[2]School of Management, University of Shanghai for Science and Technology, Shanghai 200093, China

Corresponding author: Jin Yang (cbq981208@163.com)

This work was supported by the National Natural Science Foundation of China under Grant 12071293.

ABSTRACT Unmanned aerial vehicles have a wide range of applications. An intelligent optimization algorithm based on the traditional bat algorithm (BA) is investigated in this paper for UAV flight path planning in a static complex environment. The primary goal of this work is to develop a safer flight path while considering the feasibility of the UAV and the requirements for safe operation. This research proposes an improved spherical coordinate and truncated average stable strategy-based bat optimization algorithm (TMS-SBA). The algorithm uses the UAV's motion space to encode the operator, and by substituting a new bat for the worst of the old one after each iteration to increase population diversity, the algorithm can converge quickly in a complex environment while maintaining stable operation. In addition, the flight path is smoothly generated by using B-spline curves to make the planned path suitable for UAV. MATLAB simulation experiments show that, compared with other traditional swarm intelligent algorithms, TMS-SBA can successfully generate feasible and effective optimal solutions in complex environments and plan shorter, safer, and more accessible flight paths for UAV.

INDEX TERMS Unmanned aerial vehicle (UAV), path planning, bat algorithm, configuration space, truncated mean stabilization strategy.

I. INTRODUCTION

With the continued maturity of unmanned aerial vehicle (UAV) technology, it is now possible to deploy UAV to fulfill some challenging and hazardous tasks. In fact, UAVs play an important role in the military and daily lives of many countries. As a type of contemporary aviation equipment, the UAV has been extensively employed in several industries, including search and rescue, mapping, and forestry [1]. Path planning and optimization have always been crucial components of UAV operation throughout the history of UAV research. Flight path planning is usually described as an optimization problem. The UAV needs to find the shortest path with low cost, strong security, and high computational efficiency under various constraints, such as environmental and human threats. Traditional algorithms such as the A* algorithm [2], [3], the artificial potential field [4], [5], and linear programming [6] have been proposed by previous scholars to solve the problem of path planning. However, when these methods are used in UAV path planning, problems such as high computational complexity and easy falls into local minima will often occur [7]. Similarly, the complexity and threat of the flight environment also have an impact on UAV path planning.

It has been established that determining the optimal flight path can be regarded as an NP-hard problem [8]. In order to reduce the complexity of the problem, many researchers use a swarm intelligence algorithm to solve the path planning problem. Swarm intelligence (SI) technology is derived from research on the swarm behavior of social insects, represented by ants and bees [9]. It has good performance in solving various optimization problems in the real world,

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

such as parameter estimation, neural network training, the knapsack problem, and so on. Various swarm intelligence algorithms inspired by nature have been developed, such as Genetic Algorithm (GA) [10], Particle Swarm Optimization (PSO) [11], [12], [13], [14], [15], Bat Algorithm (BA) [16], Artificial Bee Colony (ABC) [17], [18], Differential Evolution (DE) [19], and Grey Wolf Optimization Algorithm (GWO) [20]. These SI-based algorithms perform well in dealing with UAV reconnaissance paths while comprehensively considering complex terrain and other constraints to achieve global optimality.

However, these metaheuristic algorithms also have unavoidable problems. Here is a theorem called ''No Free Lunch'' (NFL), which was put forth by David H. Woolpert and William G. Ready in 1997 [21]. It has been logically proved that there is no single meta-heuristic algorithm that can perfectly solve all optimization problems. To put it another way, a particular swarm intelligence algorithm may be effective in resolving a particular optimization problem, but it is ineffective in resolving other optimization problems. In order to find a more ideal solution, people began to try to improve different intelligent algorithms to adapt to the UAV path planning problem. For example, [22] proposed a Voronoi graph-based vibration genetic algorithm to effectively solve the shortest path problem; [23] suggested an aging-based ant colony optimization algorithm (ABACO); and to address the path planning issue, [24] proposed the spherical vector-based particle swarm optimization algorithm (SPSO). Therefore, more and more algorithms have been discovered that can effectively solve the UAV path planning problem.

Recently, as one of the swarm intelligence algorithms, the bat algorithm (BA) has attracted scholars' attention due to its advantages in dealing with optimization problems. The Bat Algorithm [25] is an intelligent optimization algorithm based on the predation behavior and echolocation abilities of bats, first proposed by Xin-She Yang in 2010. BA can be used in many scenarios. Reference [26] mentioned the use of BA to detect brain abnormalities on magnetic resonance images. Reference [27] apply BA to the operation management of microgrid (MG). In [28], BA is used to solve the shop scheduling problem. The BA algorithm has been widely employed in many engineering procedures because it has a straightforward structure, fewer parameters, and superior robustness when compared to other intelligent algorithms. However, BA also has several drawbacks, such as a simple propensity to settle for the local optimal solution, which severely restricts BA's ability to advance. Therefore, some improvements to BA can improve the algorithm's performance to a large extent, so that the Bat algorithm can better adapt to the problem of UAV route planning. Reference [29] combined BA with the differential evolution algorithm (DE) to mutate bats using the evolutionary operator of DE. Reference [30] combined the artificial bee colony algorithm into BA to get the solution quickly. Reference [31] used the food search mechanism of the Drosophila algorithm (FOA) to improve the local search ability of BA. Reference [32] proposed a path planning problem using chaos strategy to improve BA, while [33] used bat algorithm to calculate multiple UAV. Compared with the traditional BA algorithm, these algorithms have a great improvement in solving single-constrained UAV path planning but still need to consider the maneuverability characteristics and environmental factors of the UAV. In an environment of high dimensions, it is often difficult to use oneself to find the global optimal position. Similarly, the application of the BA algorithm to three-dimensional path planning is in its infancy, with few resources available. On this basis, a bat configuration space algorithm based on the truncated average strategy (TMS-SBA) is proposed. The algorithm uses the configuration space of UAV to encode, making it possible to generate the initial solution in the high-dimensional space quickly. At the same time, the truncated mean stability strategy was used to enhance the diversity of the bat population and improve convergence efficiency. When the algorithm falls into a local optimum, the expansion coefficient is introduced to expand the search space. The algorithm is easy to implement, and the effectiveness and robustness of the algorithm have been proven by numerical experiments. The main contributions of this paper are summarized as follows:

1) A spherical vector bat algorithm based on a truncated mean stability strategy was proposed to solve the problem of three-dimensional path planning for UAV. This algorithm can operate stably in the search space with many path nodes and has good convergence.
2) The precise expression of the objective function is given in order to address the problem of UAV flight safety. This function transforms path planning into an optimization problem and imposes constraints on the above problems. The B-spline strategy is used to smooth the path curve.
3) In comparison to other conventional algorithms, the TMS-SBA approach produces better experimental results when used to solve the UAV path planning problem.

The remainder of this essay is organized as follows: the UAV route planning problem, including route representation, objective function, and various restrictions in practical circumstances, is discussed in Section 2. Section 3 explains the basic principle of the bat algorithm. The detailed implementation of the proposed TMS-SBA algorithm is described in Section 4. Section 5 provides a comparison between the classical algorithm and the experimental results. Finally, the sixth part summarizes the whole paper and looks forward to future work.

II. MATHEMATICAL MODEL IN UAV PATH PLANNING

Path planning is a crucial part of UAV safe route planning, which is utilized to identify the best flying path while keeping in mind restrictions such as terrain height, radar threat, smoothing cost, fuel, and the shortest path. The objective function and mathematical model of it can be explained as follows:

A. FLIGHT PATH REPRESENTATION

In the three-dimensional frame $Sg - OXYZ$, the origin of coordinate O is placed at a certain point on the ground, and the X, Y, and Z axes are taken as three orthogonal directions, with the X and Y axes being on the horizontal plane and the Z axes being on the vertical direction. The beginning point and target point are denoted by the points $S(x_s, y_s, z_s)^T$ and $T(x_t, y_t, z_t)^T$, respectively. On the basis of satisfying the constraints, route planning will then produce a secure path from S to T. A point set formed of N path nodes except S and T, that is, $L = (S, P_1, \ldots P_N, T)$, can be used to represent the UAV route. Because the UAV can generate multiple paths, each path L_i corresponds to N waypoints, and each waypoint has a path node with the coordinates $P_{ij} = (x_{ij}, y_{ij}, z_{ij})$ on the search map. So we can have two paths of the Euclidean distance between nodes as $D_{ij} = \|\overrightarrow{P_{i,j}P_{i,j+1}}\|$.

B. OVERALL COST FUNCTION

By taking into account the optimality, efficiency, and safety criteria connected to a path L_i, the overall cost function can be constructed as follows:

$$F(L_i) = \sum_{k=1}^{3} \omega_k F_k(L_i), \tag{1}$$

where ω_k is the weight coefficient, and $F_1(L_i)$ to $F_3(L_i)$ are the costs associated with fuel, threat, and flight height, respectively. The decision variable is L_i, which includes a list of n waypoints $P_{ij} = (x_{ij}, y_{ij}, z_{ij})$. The following is a detailed breakdown of how various cost functions are calculated:

1) FUEL COSTS

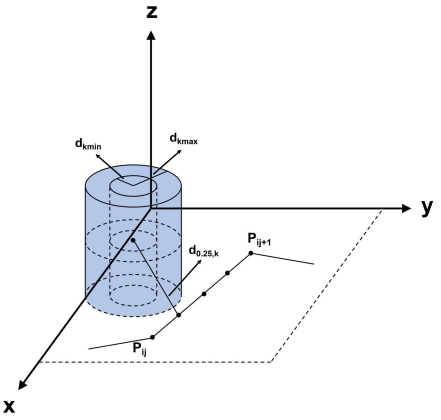
Considering the constraints of optimal path planning for UAV, we assume that the fuel cost per unit distance is the same, so the fuel distance and path length are positively correlated, and the related cost function $F_1(L_i)$ can be expressed as (2), where ρ_i is fuel consumption per kilometer.

$$F_1(L_i) = \sum_{j=1}^{n-1} \rho_i D_{ij} \tag{2}$$

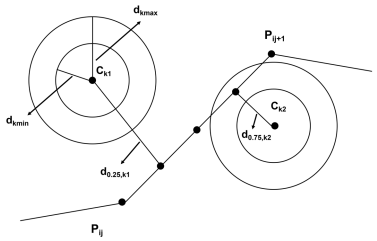
2) THREAT COSTS

When discussing UAV path planning, it is unavoidable to bring up the threat situation. For some complex and changeable terrain, there are often multiple threats, such as radar, climate, and missiles. These threats will affect the safe operation of drones. Assume that all threats are included in the set K, and that each threat is represented by a cylinder with a unique radius, the central coordinate of which is C_k (as shown in Fig.1). Equation (3) represents the likelihood of different dangers to the UAV:

$$P_k(d_k) = \begin{cases} 0, & if\ d_k > d_{kmax}; \\ \dfrac{1}{d_k^4}, & if\ d_{kmin} \le d_k \le d_{kmax}; \\ 1, & if\ d_k < d_{kmin}; \end{cases} \tag{3}$$



(a) Results of the three-dimensions view



(b) Results of the planar view

FIGURE 1. Threat cost and threat area.

where the threat probability posed by the UAV's *kth* threat source is $P_k(d_k)$. The length between the UAV and the threat source is indicated by d_k. Minimum and maximum ranges of the threat are d_{kmin} and d_{kmax}, respectively. The associated threat cost for a given path segment is proportional to its distance, d_k, from C_k. Each path is separated into an average of five segments for convenience of calculation, and the threat cost is determined separately for each discrete segment of each path. The hazard cost for this segment is predicated on the average value of the discrete segments. We may determine the price using the following (4):

$$F_2(L_i) = \sum_{j=1}^{n-1} \frac{l_j}{5} \sum_{k=1}^{u} \frac{1}{5} \cdot (T_{0,k}(j) + T_{0.25,k}(j) + T_{0.5,k}(j) + T_{0.75,k}(j) + T_{1,k}(j)) \tag{4}$$

where

$$T_{m,k}(j) = \begin{cases} 0, & if\ d_{m,k}(j) > d_{kmax}; \\ \|d_{kmax} - d_{m,k}(j)\|, & if\ d_{kmin} \le d_{k,i}(j) \le d_{kmax}; \\ \infty, & if\ d_{m,k}(j) < d_{kmin}; \end{cases} \tag{5}$$

In the equation above, u stands for the number of threatening circles, $d_{m,k}(j)$ denotes the distance between the *kth* threat center and the *mth* point on the segment.

3) ALTITUDE COSTS

Flight height is often restricted to two extremes: minimum altitude and maximum altitude. This is because in some military applications, UAVs must hide behind the landscape and cannot fly too high, but they also run the risk of collapsing into mountains at low altitudes [24]. The minimum height depends on the terrain. The likelihood of the UAV being destroyed is 1 if its flying height is lower than its peak height; conversely, it is 0. Let the minimum and maximum heights be h_{min} and h_{max}, respectively. A waypoint P_{ij} associated altitude cost is calculated as follows:

$$H_{ij} = \begin{cases} |h_{ij} - \dfrac{(h_{max} + hmin)}{2}|, & if\ h_{min} \le h_{ij} \le h_{max}; \\ \infty, & otherwise; \end{cases} \tag{6}$$

where h_{ij} is the flight altitude relative to the ground. In this paper, the altitude is restricted by setting a penalty item so that it can always maintain a relatively stable state and avoid the collision caused when the flight altitude is too low. Sum H_{ij}over all waypoints to get the height cost:

$$F_3(L_i) = \sum_{j=1}^{n} H_{ij} \tag{7}$$

Given the need to generate a UAV-safety route, this paper introduces yaw and pitch angle constraints to constrain the UAV's path:

$$\psi_k = |arctan(\frac{y_{k+1} - y_k}{x_{k+1} - x_k})| \le \psi_{max} \tag{8}$$

$$\phi_k = |arctan(\frac{z_{k+1} - z_k}{x_{k+1} - x_k})| \le \phi_{max} \tag{9}$$

In this formula, ψ_{max} is the maximum pitch angle, and ϕ_{max} is the maximum yaw angle, and ψ_k and ϕ_k are respectively the pitch angle and yaw angle of the waypoint P_k.

Under the above definition, the cost function F is completely determined and can be used as input in the path planning process, which is conducive to the calculation of the simulation experiment below.

III. CLASSICAL BAT ALGORITHM

The classic bat algorithm is a swarm intelligence algorithm. The social interactions of bats and their use of echolocation for hunting and avoiding obstacles served as the basis for their search approach. In addition, the bat algorithm is a promising method that, to some extent, combines the benefits of the PSO and GA algorithms [34].

Some bats in the wild use their sight and smell in addition to echolocation to locate food and avoid predators. Even the sounds and frequencies that bats produce are dynamic. The purpose of this paper is to study and simulate the flight path of the UAV in three-dimensional space, so this paper idealizes some of the echoic characteristics of bats for the sake of simplicity, assuming that their movements are entirely echo-related and that they can automatically change the wavelength and frequency of their transmitted pulses when in pursuit of prey. They fly at position X_i with a speed of V_i. Let a fixed frequency of f_{min}, a loudness of A_0, and a pulse frequency of $r \in [0, 1]$ that is constantly adjusted according to the distance from the target. It is important to note that loudness is not a fixed value, it ranges from the smallest constant, A_{min} to A_0. The revised formula for the velocity V_i^t, frequency f_i, and position X_i^t of the *ith* bat in a state with time step t is as follows:

$$f_i = f_{min} + (f_{max} - f_{min}) \times \beta \tag{10}$$

$$V_i^t = V_i^{t-1} + (X_i^{t-1} - X^w) \times f_i \tag{11}$$

$$X_i^t = X_i^{t-1} + V_i^t \tag{12}$$

The best solution to the current time step is X^w. Where $\beta \in [0, 1]$ is a random value.Frequency is a random constant. Regarding the local search section, it uses a random number $rand_i \in [0, 1]$. If $rand_i > r_i$, let a new solution X_{new} replace the original solution X_i^t.

$$X_{new} = X^w + \varepsilon A^t \tag{13}$$

where A^t is the average loudness of all bats at time step t and the random number $\varepsilon \in [-1, 1]$. Equations (14) and (15) are used to update when the fitness $f(X_i^t)$ is smaller than the fitness of the current optimal solution $f(X^w)$:

$$A_i^{t+1} = \alpha A_i^t \tag{14}$$

$$r_i^{t+1} = r^0[1 - exp(-\iota t)] \tag{15}$$

where α, ι, r^0 are constants. Based on the analysis above, Fig.2 and Algorithm 1 describe the main portion of the traditional bat algorithm.

IV. IMPORVED BAT ALGORITHM

The standard BA algorithm is enhanced in order to find the global optimal solution of the BA more quickly and precisely, and a spherical coordinate system is constructed to solve the flight path based on the maneuvering characteristics of the UAV configuration space, thus significantly reducing the search space. On the other hand, considering that the BA algorithm is still unable to overcome the problem of falling into local optimality when solving optimization problems, the truncated mean problem strategy is adopted in this paper to replace bats with the worst fitness values to increase the diversity of the population.

A. CONFIGURATION SPACE CODING

When determining the three-dimensional path of a UAV, the swarm intelligence algorithm frequently searches in the cartesian coordinate system, which is inefficient and difficult to solve the constraint conditions. The configuration space of the UAV can be regarded as a spherical space; therefore, the main function of using the spherical vector in the BA algorithm is to represent the attitude of the camera. The amplitude, elevation, and azimuth components of the UAV in the configuration space are used to limit the speed, yaw angle, and pitch angle of the UAV, and the relationship between path

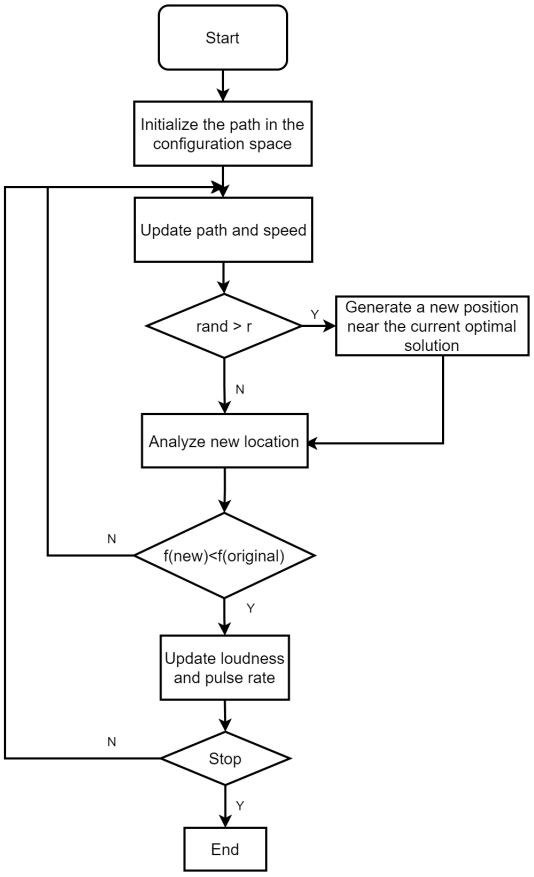


FIGURE 2. Flow chart of BA algorithm.

points is used to reduce the search space and accelerate the convergence.

The complexity of UAV flying is decreased, and the convergence rate is accelerated by addressing the problem in configuration space and mapping it to cartesian space. We encode each path L_i for the UAV flight path $L = (S, p_1, \ldots, p_N, T)$ stated in Part 2.1, where $P_{ij} = (x_{ij}, y_{ij}, z_{ij})$ is the number of waypoints on the path L_i, and N trajectory segments make up the path. A spherical coordinate system is constructed for P_{ij}, including amplitude r, elevation ψ, and yaw angle ϕ.

The formula for the conventional spherical coordinate transformation is given in (16), where d_{ij} stands for the sphere's radius. On the basis of this, spherical coordinates are created for each path node, turning it into a distinct entity.

$$\begin{aligned} x_{ij} &= d_{ij} sin\psi_{ij} cos\phi_{ij} \\ y_{ij} &= d_{ij} sin\psi_{ij} sin\phi_{ij} \\ z_{ij} &= d_{ij} cos\psi_{ij} \end{aligned} \tag{16}$$

As shown in Fig.3, a sphere is constructed on each path node, where r is the amplitude of the UAV, which is related to both the sphere radius of the path node and the path length. The three elements of each waypoint $P_{ij} = (r_{ij}, \phi_{ij}, \psi_{ij})$ are

Algorithm 1 Classical BA Algorithm

Require:
Ensure:
Initialization. Set the generate counter $t = 1$, the amount of bat populations np, initialize A_0, V_0, r, and the maximum number of iterations T_{max}.
Get the fitness of each bat $f(X_i^0)$.
for $t = 1 : T_{max}$ **do**
  for $i = 1 : np$ **do**
    Update frequency by (10);
    Update bat speed by (11);
    Update bat locations by (12);
    if $rand_1 > r_i^t$ **then**
      Produce a new solution X_{new} instead of X_t;
      $X_{new} = X^w + \varepsilon A^t$
    end if
    Get the new solution's fitness $f(X_i^t)$
    if $f(X_i^t) < f(X^w)$ **then**
      Accept the fresh solutions, then update r_i^t, A_i^t
    end if
  end for
  Update the present optimal solution X^w.
end for
Output
End

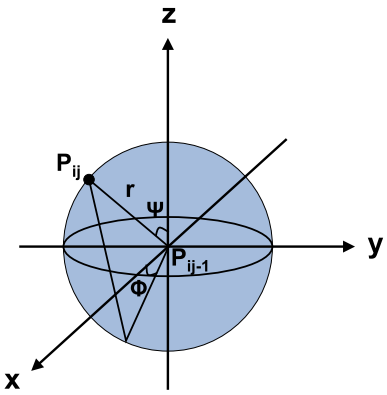


FIGURE 3. Configuration space coding.

bound in the following ways to adhere to the prior limitations:

$$\begin{aligned} r_i &\in (0, pathlength) \\ \phi_{ij} &\in (\phi_{max} - \phi_{i,j-1}, \phi_{max} + \phi_{i,j-1}) \\ \psi_{ij} &\in (\psi_{max} - \psi_{i,j-1}, \psi_{max} + \psi_{i,j-1}) \end{aligned} \tag{17}$$

where ϕ_{max} and ψ_{max} are the maximum angles of yaw and elevation, respectively. Then, for a complete path L, it is expressed in the form of a spherical coordinate sequence as shown in (18).

$$L_{inew} = (S, (r_{i1}, \phi_{i1}, \psi_{i1}), \ldots, (r_{iN}, \phi_{iN}, \psi_{iN}), T) \tag{18}$$

Since the UAV is in configuration space, its velocity also has three components related to spherical coordinates. To limit the bat speed at each path node, we can similarly set

upper and lower bounds on the UAV speed, as shown in (19):

$$\begin{aligned} Vr_i &\in (-\alpha(r_{max} - r_{min}), \alpha(r_{max} - r_{min})) \\ V\phi_{ij} &\in (-\alpha(\phi_{max} - \phi_{min}), \alpha(\phi_{max} - \phi_{min})) \\ V\psi_i &\in (-\alpha(\psi_{max} - \psi_{min}), \alpha(\psi_{max} - \psi_{min})) \end{aligned} \tag{19}$$

where α is a random number on [0,1]. We denote the spherical vector of the waypoint on L_i as xS and the velocity update as vS, then the iterative equation of a bat's velocity and position at time step t can be updated as follows:

$$vS_i^t = vS_i^{t-1} + (xS_i^{t-1} - xS^w)f_i \tag{20}$$

$$xS_i^t = xS_i^{t-1} + vS_i^t \tag{21}$$

Calculating the fitness value, which is the corresponding cost function of each path, is important when determining the individual and global optimum of the bat population. At this point, the flight route computed above must be remapped to cartesian space. Due to the interdependence of path nodes, the following formula can be used to determine the decoding mode of each path node coordinate:

$$\begin{aligned} x_{ij} &= x_{i,j-1} + r_{ij} sin\psi_{ij} cos\phi_{ij} \\ y_{ij} &= y_{i,j-1} + r_{ij} sin\psi_{ij} sin\phi_{ij} \\ z_{ij} &= z_{i,j-1} + r_{ij} cos\psi_{ij} \end{aligned} \tag{22}$$

Based on this, we obtained the basic principle of the SBA algorithm, which is more conducive to dealing with angle and velocity constraints between adjacent path nodes. This algorithm enables individual bats to converge to the optimal position faster but also faces the problem of falling into local optimality. In view of this situation, the algorithm was further improved in this paper.

B. STRATEGY FOR TRUNCATED MEAN STABILIZATION

The SBA algorithm is produced in the analysis above by solving the BA algorithm in a spherical coordinate system. To address the issue of bat populations being prone to falling into local optimum, a population stabilization strategy was proposed to increase population diversity while avoiding falling into local optimum. The purpose of the truncated mean stabilization strategy is to compute the sample average by removing the same number of extreme values from both ends of the sample. Without reducing the number of samples, this method can replace the worst sample value. The number of samples to be eliminated is usually a percentage of the sample or a fixed number. Reference [35] proposed a multi-scale quantum harmonic oscillator algorithm based on a truncated average stabilization strategy. The algorithm demonstrated that it is possible to broaden the diversity of particles by employing the truncated average strategy. Therefore, this paper used it to develop a new algorithm (TMS-SBA).

First of all, the SBA algorithm is used to obtain the location, speed, and other specific information of the bat population, then the fitness value of each bat is sorted, a certain proportion of bat individuals are selected to obtain

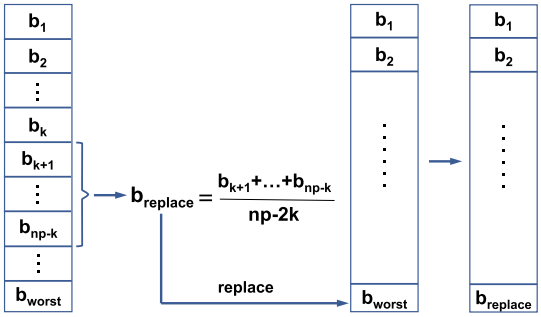


FIGURE 4. Truncated mean stabilization strategy.

the average value of their positions, and the bat individuals with the highest fitness, namely the so-called worst bats, are replaced. As the number of iterations increased, the bats with the worst fitness scores were replaced each time, increasing the diversity of the population. The fitness value here can be replaced by the cost function:

$$f(L_i) = F(L_i) \tag{23}$$

Let $X = (X_1, X_2, \ldots, X_{T_{max}})$, where X_i is the set of bat population position vectors after the *ith* iteration, then the specific strategy to truncate the average stability is as follows:

Mark the location of the bat population as $b_1, b_2, \ldots, b_{np}$ after one iteration X. Each bat's fitness value is determined; sort it from smallest to largest, and the one with the highest value is designated as b_{worst}, and we get $b_1, b_2, \ldots, b_{worst}$. After selecting a random value $\beta \in (0, 1)$, the number of bats we choose to intercept can be set to k. Here is the formula for calculating k:

$$k = [np \times \frac{\beta}{2}] \tag{24}$$

where np is the number of bat populations, then the position of the truncated particle is going to be $b_k, b_{k+1}, \ldots, b_{np-k}$. So the truncated average vector of the bat's position can be obtained by:

$$b_{replace} = \frac{1}{np - 2k} \sum_{i=k}^{np-k} b_i \tag{25}$$

We replace b_{worst} with $b_{replace}$, and the new bat position vector obtained can be written as $b_1, b_2, \ldots, b_{replace}$.

Figure 4 shows the flow of truncating the mean stability policy. Through iteration after iteration, a new bat is generated each time without regard to locally optimal information. The bats with the worst fitness in the bat population are replaced each time, so as to increase the bat population, improve the convergence performance of the algorithm, and reduce the possibility of the algorithm falling into the local optimal to a certain extent.

Similarly, considering an extreme case where the algorithm's fitness value does not change over multiple iterations, it means that the algorithm is falling into local optimality and all bat individuals have the same fitness value. In this case, this paper adopts a strategy of expanding the search space.

Algorithm 2 TMS-SBA Algorithm in UAV Path Planning

Require:

Ensure:

Initialization. Set np, T_{max}, A_0, V_0, r, the generate counter $t = 1$, and constant $b_{limit} = 0$. Get coordinate information for the search map.

for $i = 1 : np$ **do**

Create an initial path L^0_{inew} in a spherical coordinate system, where the number of path nodes is the search space dimension D.

The cost function f for each bat is calculated in Cartesian space and set to the fitness value. Initializes the global optimal location.

end for

for $t = 1 : T_{max}$ **do**

for $i = 1 : np$ **do**

The speed and position of each bat are calculated on the UAV configuration space.

Update the velocity and position by (20) and (21)

if $rand_1 > r_i^t$ **then**

Get a new solution X_{new};

end if

The position information is mapped to the Cartesian coordinate system using (22), and then calculate the new solution's fitness $f(X_i)$.

if $f(X_i^t) < f(X^w)$ **then**

Accept the new solutions and update r_i^t, A_i^t

end if

end for

Update the current optimal solution X^*.

The position information $b_1^t, b_2^t, \ldots, b_{np}^t$(of course this position information is in spherical coordinates) of each bat is obtained by sorting $f(X_i)$ from small to large.

Select the number of bats to be intercepted k by (24):

Identify replacement bats $b^t_{replace}$ by (25)

Checks whether b^t_{worst} and $b^t_{replace}$ are equal, if they're equal, $b_limit = 1$

$b^t_{worst} = b^t_{replace}$

if $b_limit = 1$, expand the search space and re-initialize the bat population

The path cost is calculated by greedy criterion and the original path is replaced by the optimal path.

Update the optimal solution

end for

Choose the optimal solution as the final result.

Save the best path associated with the global optimal solution L_{new}

End

The detection is performed after each iteration of the algorithm. If $b_{replace} = b_{worst}$, expanding the search space with the aid of the expansion coefficient w [35] may aid in the population's departure from the local optimal. The primary component of TMS-SBA is described in Fig.5 and Algorithm 2.

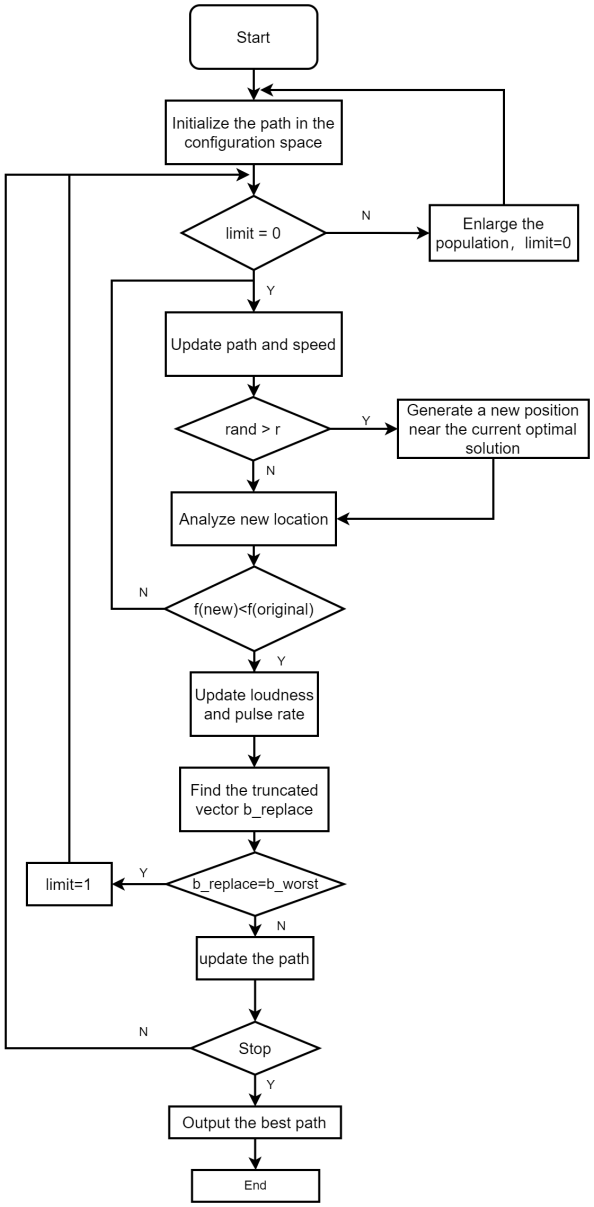


FIGURE 5. Flow chart of path planning for TMS-BA algorithm.

V. RESULT

A. PATH SMOOTHING

When the TMS-SBA method is used to address the UAV route planning problem, considering that the generated path is usually difficult to fly accurately, the B-spline curve smoothing strategy is adopted to dynamically smooth the flight path. B-spline is a generalization of Bessel curves, allowing us to build accurate models for more general geometry. The B-spline curve has many excellent properties, such as geometric invariance and convexity preservation. Similarly, it only needs a few variables to define the complex curve path.

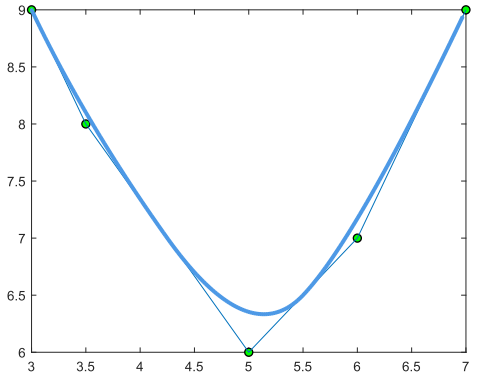


FIGURE 6. B-spline curve.

Suppose there are $n+1$ control points $c_i(i = 0, 1 \ldots, n)$ and a node vector u, b-spline curve can be defined as:

$$B(u) = \sum_{i=0}^{n} c_i M_{i,k}(u) \tag{26}$$

where $M_{i,k}(u)$ is B-spline basis function of degree k, also called harmonic function, its recursive formula can be expressed as follows:

$$\begin{cases} M_{i,0}(u) = \begin{cases} 1, & if\ u_i \le u \le u_{i+1} \\ 0, & otherwise \end{cases} \\ M_{i,k}(u) = \dfrac{u - u_i}{u_{i+k} - u_i} N_{i,k-1}(u) + \dfrac{u_{i+k+1} - u}{u_{i+k+1} - u_{i+1}} \end{cases} \tag{27}$$

The smoothing of the path using the B-spline approach is shown in Fig.6. Because the degree of the polynomial cannot be raised no matter how many points are added, the B-spline is better suited to dealing with three-dimensional path problems than the Bezier curve.

B. UAV PATH PLANNING PROBLEM

In this section, the TMS-SBA algorithm is simulated in a three-dimensional environment. Not only the parameters of the algorithm are compared and analyzed, but also the TMS-SBA algorithm is compared with other traditional evolutionary algorithms to verify the feasibility and effectiveness of the algorithm in UAV path planning. In addition, the software used in this paper is MATLAB R2018a, and the experimental simulation is carried out under the computer processor Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz,.80GHZ, 64-bit Windows 10 operating system.

This work constructs four three-dimensional flight scenes, namely three simple flat terrain scenes and one relatively complex scene, in an effort to simulate the real flight environment as closely as possible. Data clusters are employed in complex situations to replicate the altitudes of mountains and establish the range and quantity of threats depending on geographic information. This paper first discusses the algorithm's performance in a simple environment. Setting the start node $S(200, 100, 0)$ and target node $T(800, 800, 150)$ of all paths. For comparison, set the maximum number of iterations to 200. We conducted 50 simulation tests on each algorithm to ensure the accuracy of the experiment.

TABLE 1. The information of threat areas.

Scenario	Threat center	Threat radius
1	(400,500,0)	80
	(500,300,10)	70
	(700,400,0)	70
2	(400,500,0)	80
	(600,200,50)	70
	(500,350,20)	80
	(700,550,0)	70
	(200,400,50)	70
3	(400,500,0)	80
	(600,200,50)	70
	(500,350,20)	80
	(350,200,40)	70
	(700,550,50)	70
	(650,750 0)	80
	(700,400,30)	50
	(200,700,50)	70
	(200,400,10)	70

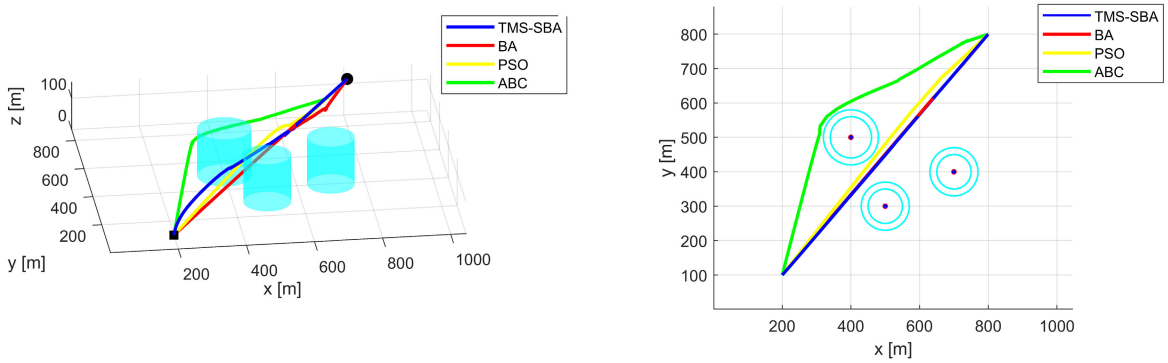
Table 1 shows the threat center and radius of the three threat areas in a simple environment, according to the hypothesis. The threat area is set as a cylinder, which the UAV cannot circumnavigate. Because the terrain is relatively flat, the flight parameters are set as $h_{min} = 0$, $h_{max} = 200$. The drone mission area is 1,000 kilometers long and 1,000 kilometers wide. Table 2 shows the comparison results between the TMS-SBA algorithm and the BA, PSO, and ABC algorithms in the simple environment. Where n is the number of control path nodes. *Mean*, *std*, *worst*, and *optimal* represent the average fitness value (average cost function value), standard deviation, worst fitness value, and optimal fitness value, respectively.

Fig.7-Fig.9 shows the optimal UAV route generated by the above four algorithms in scenarios 1, 2 and 3, respectively, in $n = 10$ and 50 independent operations. Part (a) shows a three-dimensional view of the drone route in the digital terrain, where the cylinder represents the threat area. Part (b) is a two-dimensional top view of (a) in the contour map. As can be seen, all algorithms can generate viable paths that meet the path length, threat, height requirements, and constraint requirements. However, their optimality varies from scenario to scenario. It can be observed that scenario 1 has good convergence for the four algorithms due to the small number of threat areas, with slightly different fitness values. The average of the total running time of each algorithm on all functions is regarded as the average running time, due to the existence of truncated mean strategy and spherical coordinate transformation, the CPU running time of TMS-SBA is slightly slower than that of the other three algorithms. The PSO algorithm has the best performance when generating a simple path, and the average cost function value is 480.2139.

In scenario 2, we slightly increased the number of threat areas in the space, making the whole search space complicated. As can be seen from Fig.8, TMS-SBA generated

TABLE 2. Comparison results of three threat regions with other algorithms in simple scenario (n = 10).

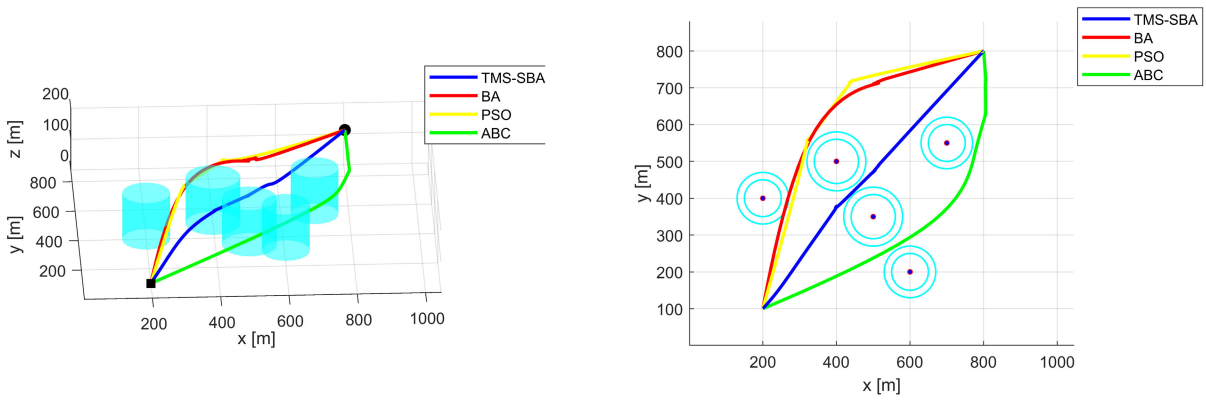
Scenario		TMS-SBA	BA	PSO	ABC
1	Mean	481.6402	482.7582	**480.2139**	497.5963
	Std	**9.5292**	13.2648	15.0324	75.7092
	Worst	**516.8299**	519.5787	530.1891	1009.634
	Optimal	471.0628	472.2682	**470.1322**	470.5402
	CPU processing time	77.452	**71.477**	76.229	75.167
2	Mean	**487.7242**	543.0238	574.2817	551.1881
	Std	**8.093**	74.2397	50.0664	25.0235
	Worst	**515.4777**	1023.137	815.9874	644.0192
	Optimal	**476.2464**	502.1716	521.1681	511.2166
	CPU processing time	101.231	107.12	**99.907**	110.736
3	Mean	**489.6649**	571.8182	587.9411	545.6102
	Std	**12.5106**	24.4708	17.5628	31.3244
	Worst	**535.7644**	628.1477	623.4705	605.1711
	Optimal	**476.2464**	522.386	535.111	519.2341
	CPU processing time	**131.714**	371.185	392.997	316.0243



(a) Results of a 3D view comparison

(b) Results of a overlooking view comparison

FIGURE 7. Comparison results in Scenario 1.

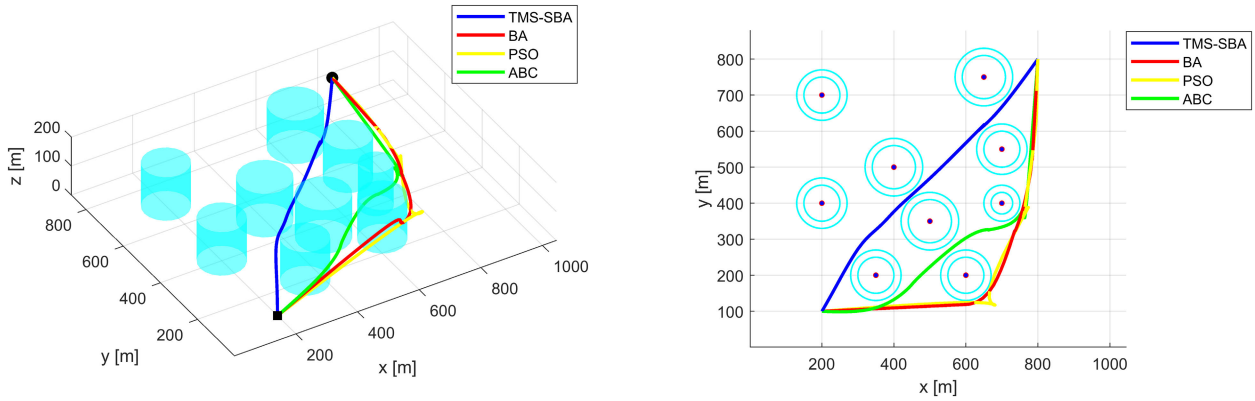


(a) Results of the 3D view comparison

(b) Results of the overlooking view comparison

FIGURE 8. Comparison results in Scenario 2.

a feasible path with the minimum cost function, the average fitness value was 487.7242, and the standard deviation of the algorithm was 8.093, showing good performance. While the results of BA, PSO, and ABC can better meet the path planning requirements, PSO is more likely to fall into the local optimal value during an iteration, which will result in additional threat costs; the ABC algorithm's convergence speed is sluggish. At the same time, in 50 experiments, BA is easy to fall into the local optimal solution and cannot jump out, so the worst fitness value is 1023.137. The



(a) Results of the 3D view comparison

(b) Results of the overlooking view comparison

FIGURE 9. Comparison results in Scenario 3.

TABLE 3. Comparison the cost value of the comparison algorithm changes with the number of iterations in simple scenario 2 (n = 10).

T_{limit}		TMS-SBA	BA	PSO	ABC
50	Mean	**496.0207**	655.0141	631.6318	642.7448
	Worst	**530.1679**	745.2813	1243.9626	823.0273
	Optimal	**493.1003**	601.2753	600.9562	589.0825
100	Mean	**486.1722**	587.9128	593.1688	590.6681
	Worst	**516.7413**	671.3263	854.2098	743.0622
	Optimal	**482.3896**	510.9849	521.8671	504.2164
500	Mean	**485.2751**	587.2594	590.2642	589.48871
	Worst	**518.0221**	509.8172	853.9763	744.3324
	Optimal	**482.1003**	508.9371	519.6115	503.5458
1000	Mean	**484.1826**	587.1918	589.0192	588.8222
	Worst	517.4272	**508.3592**	853.0249	741.7064
	Optimal	**481.2596**	508.0376	517.9894	503.0554

TABLE 4. The influence of different population size on the algorithm in Scenario 2(n = 10).

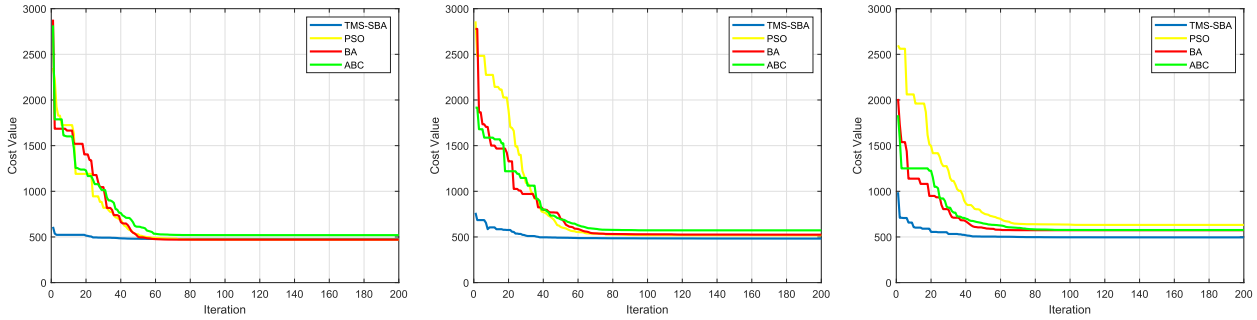
population		TMS-SBA	BA
50	Mean	**528.0925**	602.0553
	Worst	**562.0104**	739.9185
	Optimal	**498.9471**	528.5669
	CPU processing time	17.855	**17.039**
100	Mean	**513.5281**	572.0553
	Worst	**555.9014**	664.9418
	Optimal	**485.1075**	521.2579
	CPU processing time	35.105	**33.435**
200	Mean	**493.5741**	556.0899
	Worst	**546.1476**	588.7565
	Optimal	**482.8726**	517.8965
	CPU processing time	**67.73**	72.709
500	Mean	**491.2041**	552.7473
	Worst	**512.3108**	584.8397
	Optimal	**487.7352**	513.4642
	CPU processing time	**140.647**	168.232

standard deviation of the BA algorithm is the largest, and its performance is not stable.

In Scenario 3, we set up nine threat areas, which made it much more challenging to search the region. It can be seen that TMS-SBA can develop the initial solution quickly and can approach the ideal solution with an average fitness value of 489.6649. However, as the threat area grows, PSO and BA's ability to find the best solution suffers, and they become more vulnerable to the impact of extreme values in their own populations. The ABC algorithm tends to generate more average pathways as a result of the compromise between various bee types; hence, the performance is decent. Meanwhile, in scenario 3, the time to initialize the path increases due to the complexity of the environment. Since TMS-SBA is generated in the spherical coordinate system of the UAV, and the coordinates of each path node are related to each other, the time to generate the initial solution is much shorter than the other three algorithms. The results show that TMS-SBA can reach the optimal value in most cases.

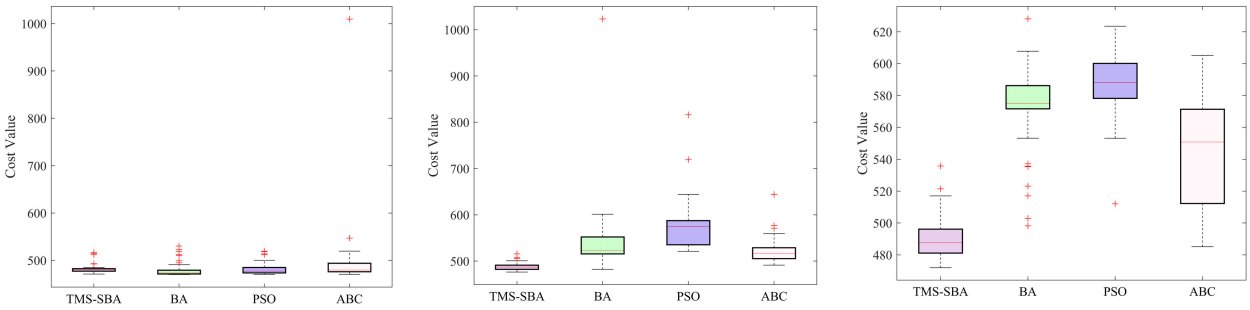
In order to make it easier for us to observe the behavior of the variables more precisely, Fig.10 displays the change curves of the fitness values of the four algorithms with each iteration. It is clear that the four algorithms function well in scenario 1's comparatively straightforward context. The quality of the solution, however, degrades as the scene's complexity rises due to PSO's shortcomings in global search. The population of BA is also vulnerable to early convergence because of the existence of super bats. Although ABC can find the average path, the convergence happens slowly. Finally, the TMS-SBA algorithm has more advantages in the search space since it is stored in a spherical coordinate system and is directly bound by the turning angle in the cost function. Additionally, TMS-SBA avoids the scenario of premature convergence to the precocity solution like BA and increases the diversity of the population by truncating the mean stability strategy, making it perform well in most situations. Fig.11 shows the box diagram of the four algorithms in 50 experiments. It can be seen that TMS-SBA produces the fewest outliers, is more stable overall, and performs well in most cases.

The effects of each variable on the functionality of the TMS-SBA algorithm should be discussed. The algorithm proposed in this paper is based on multiple variables, in which the spatial dimension, the number of iterations, and the number of populations all have an impact on the optimal path of the algorithm. Therefore, we carried out experiments to



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

FIGURE 10. The convergence curves of the four algorithms.



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

FIGURE 11. Boxplot with 50 iterations of four algorithms.

analyze the influence of each parameter on the algorithm. We chose BA for comparative analysis in the comparison experiment because TMS-SBA is an improvement over BA, and we let the two algorithms choose the same parameters and other algorithms for auxiliary comparison. Scenario 2, of moderate complexity, is used as an example for clarity.

1) NUMBER OF ITERATIONS

Table 3 shows the impact of different iterations on the overall change. As can be seen from the data in Fig.10 and table, TMS-SBA can find a better solution after about 50 iterations, while BA and PSO perform generally, because they are easy to fall into local optimals and not easy to get rid of, and the convergence speed of the ABC algorithm is slow. In summary, the TMS-SBA algorithm has a significant advantage in convergence speed.

2) NUMBER OF POPULATION

Table 4 shows the influence of different population numbers on the algorithm, and the number of iterations is set to 200. It can be seen that with the increase in population, the ability of TMS-SBA and BA to find the optimal solution becomes stronger, but the required iteration time also increases. For BA, with a population size of 50, it is easy to fall into the limitation of local search solutions, resulting in the illegal finding of appropriate paths. For TMS-SBA, the phase strategy can make it jump out of the influence of a larger fitness value and make the search for solutions more stable. In this paper, if the impact of time cost is taken into account, better results can be obtained faster if the population is set at around 200, and 200 to 500 is more appropriate if a solution of higher quality is to be obtained stably.

TABLE 5. Influence of path node n on cost value in scenario 2.

algorithm		n =5	n=10	n=15
TMS-SBA	Mean	**494.7242**	**494.9166**	**496.7216**
	Worst	518.5188	**515.8917**	**522.0972**
	Optimal	481.2717	**476.2464**	**489.8841**
BA	Mean	516.4399	543.0228	616.2755
	Worst	580.0036	673.8124	854.2098
	Optimal	**474.8243**	521.8671	583.9849
PSO	Mean	513.3674	581.2557	875.0894
	Worst	518.0221	815.9874	2408.7629
	Optimal	475.6404	530.1165	597.2584
ABC	Mean	508.4973	511.0236	637.8574
	Worst	**515.6773**	647.3129	895.7127
	Optimal	488.6156	498.2166	550.1766

TABLE 6. The information of complex environment.

Threat center	(200,4500,50)	Threat radius	60
	(650,750,25)		70
	(500,550,0)		50
	(350,200,70)		70
	(700,550,0)		70
	(450,400,0)		60
	(250,350,10)		40

TABLE 7. Comparison results of four algorithms in complex environment (n = 10).

Scenario		TMS-SBA	BA	PSO	ABC
complex	Mean	**514.2963**	595.8998	644.1671	618.7789
	Std	**17.5402**	66.4799	127.8272	91.4022
	Worst	**559.4135**	619.5787	1041.1891	836.2991
	Optimal	**492.9784**	532.2682	570.1322	584.9538
	CPU processing time	**149.298**	195.720	180.471	173.824

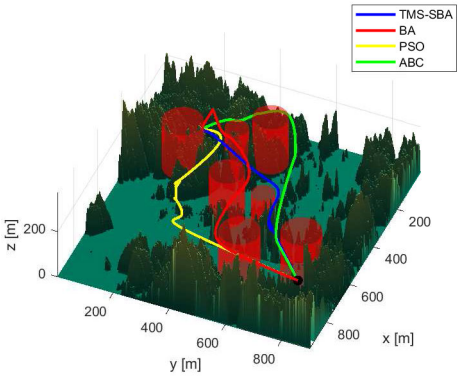


FIGURE 12. Results of the 3D view comparison in complex environment.

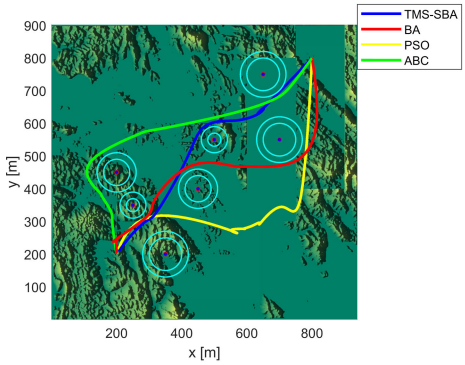


FIGURE 13. Results of the overlooking view comparison in complex environment.

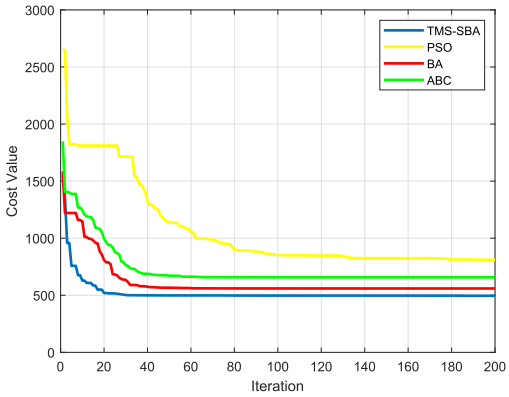


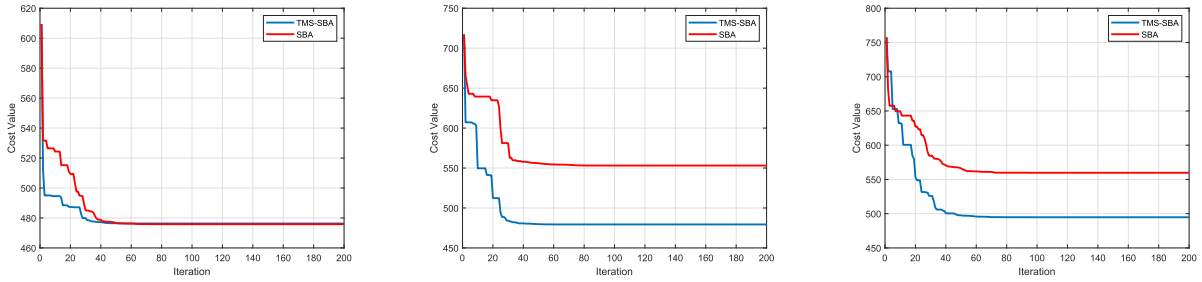
FIGURE 14. Convergence curves of the four algorithms.

3) NUMBER OF PATH NODES

Table 5 shows the effect of different path nodes n on the algorithm. What you need to know is that the dimension of the search space depends on n, which is $d = 3n$. From table 5, as you can see, the four algorithms have basically the same search ability in low-dimensional space. Because BA, PSO, and ABC have weak searching abilities in high-dimensional space, with the increase of path nodes, the optimal path is often unable to be generated or only a poor path can be generated. On the other hand, due to its own coding characteristics, the convergence rate of TMS-SBA is almost not affected, and it performs well in high-dimensional space.

Similarly, this paper analyzes algorithm performance in complex environments. The data cluster is employed in the search space to imitate the height of a mountain, and the threat area is set in accordance with the topography. Table 6 displays the related values. The range of height constraints for UAVs expands as the complexity of the terrain does. Table 7 shows the results after 50 independent runs of the four algorithms, in which the number of iterations is set to 200, the population is set to 500, $N = 10$, the start node $S = (200, 200, 0)$, and the target node $T = (800, 800, 150)$. The flight height at this time is $h_{min} = 100$ and $h_{max} = 300$. As you can see, TMS-SBA can generate a good flight path while avoiding the impact of threat areas and mountain heights. Due to the limitation of the flight height, it is more difficult for PSO, BA, and ABC to generate the optimal path, and the path cannot be generated, which can also be seen from the data of the standard deviation. Fig.12 and Fig.13 show the three-dimensional image and top view of the UAV route generated by four algorithms in a complex environment. It can be seen that the path of TMS-SBA is more stable and smooth than that of other traditional evolutionary algorithms. Fig.14 shows the convergence curves of the four functions. The convergence rate of TMS-SBA is fast enough.

Finally, it is necessary to discuss the influence of the truncated mean stability strategy on the algorithm. We run TMS-SBA and SBA without a truncated mean stability strategy in parallel in the same environment. The conditions of other parameter variables are unchanged, and the initialization paths of the two algorithms are the same. Fig.15 shows the convergence curves of the two algorithms in the same environment. It can be seen that, compared with SBA algorithm, TMS-SBA algorithm can generate optimal fitness value path in most cases, while SBA algorithm is easy to fall into local optimal value, resulting in unstable results. TMS-SBA may occasionally be affected by local solutions, but in



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

FIGURE 15. The convergence curves of the two algorithms.

the process of iteration, bats with the worst fitness values are replaced by bats with an average fitness value, which increases the diversity of the population and thus can avoid falling into a local optimal state in most cases.

VI. CONCLUSION

In this study, we propose an improved TMS-SBA algorithm to solve the path planning problem of UAV, so that the UAV can obtain a safe and non-collision optimal path. The algorithm is based on a combination of the UAV's own motion space and search space. Meanwhile, a new truncated mean stability strategy is designed in this paper, which maintains the diversity of the population in the iterative process, improves the convergence performance, and prevents the algorithm from falling into the local optimal solution. This paper tested the performance of the algorithm in different threat areas and flight altitudes through four different scenarios and compared it with the performance of the classical evolutionary algorithms such as PSO, BA, and ABC. Experiments show that TMS-SBA can find the best flight path location information in most cases, and the performance is significantly better than the traditional evolutionary algorithm when there are more path nodes.

The algorithm parameters are compared and analyzed in this paper. As the population size grows, the algorithm's iteration time inevitably grows, and as the number of path nodes rises, the algorithm's stability also declines. TMS-SBA can perform stable and efficient computation in high-dimensional space. In terms of function optimization, TMS-SBA can produce better results with a lower standard deviation when compared to ABC, BA, and PSO. The flight scenario set forth in this paper can effectively make the UAV avoid the threat area. However, because the design in this paper cannot go over the top of the threat area in the face of a dense mountain and jungle environment, the path cost must be generated. At the same time, in contrast to other methods, the algorithm's general applicability isn't as great because it was created to take into account the features of the UAV configuration space. Therefore, further improvement is needed in the future work. In the following work, we will also extend our research to the problem of multi-UAV collaborative route planning.

REFERENCES

[1] E. Besada-Portas, L. de la Torre, J. M. de la Cruz, and B. de Andrés-Toro, "Evolutionary trajectory planner for multiple UAVs in realistic scenarios," *IEEE Trans. Robot.*, vol. 26, no. 4, pp. 619–634, Aug. 2010, doi: 10.1109/TRO.2010.2048610.

[2] Z. Ma, H. Qiu, H. Wang, L. Yang, L. Huang, and R. Qiu, "A algorithm path planning and minimum snap trajectory generation for mobile robot," in *Proc. 4th Int. Conf. Robot., Control Autom. Eng. (RCAE)*, Nov. 2021, pp. 284–288, doi: 10.1109/RCAE53607.2021.9638900.

[3] Y. Wu and X. Qu, "Path planning for taxi of carrier aircraft launching," *Sci. China Technol. Sci.*, vol. 56, no. 6, pp. 1561–1570, 2013, doi: 10.1007/s11431-013-5222-5.

[4] Y. B. Chen, J. Q. Yu, X. L. Su, and G. C. Luo, "Path planning for multi-UAV formation," *J. Intell. Robot. Syst.*, vol. 77, no. 1, pp. 229–246, 2015, doi: 10.1007/s10846-014-0077-y.

[5] X. Chen and J. Zhang, "The three-dimension path planning of UAV based on improved artificial potential field in dynamic environment," in *Proc. 5th Int. Conf. Intell. Hum.-Mach. Syst. Cybern.*, vol. 2, Aug. 2013, pp. 144–147, doi: 10.1109/IHMSC.2013.181.

[6] M. Radmanesh, M. Kumar, A. Nemati, and M. Sarim, "Dynamic optimal UAV trajectory planning in the national airspace system via mixed integer linear programming," *Proc. Inst. Mech. Eng., G, J. Aerosp. Eng.*, vol. 230, no. 9, pp. 1668–1682, Jul. 2016, doi: 10.1177/0954410015609361.

[7] Y. Zhao, Z. Zheng, and Y. Liu, "Survey on computational-intelligence-based UAV path planning," *Knowl.-Based Syst.*, vol. 158, pp. 54–64, Dec. 2018, doi: 10.1016/j.knosys.2018.05.033.

[8] G. J. Woeginger, "Exact algorithms for np-hard problems: A survey," in *Combinatorial Optimizationa Eureka, You Shrink!*. Berlin, Germany: Springer, 2003, pp. 185–207, doi: 10.1007/3-540-36478-1_17.

[9] Z. Cui and X. Gao, "Theory and applications of swarm intelligence," *Neural Comput. Appl.*, vol. 21, no. 2, pp. 205–206, Mar. 2012, doi: 10.1007/s00521-011-0523-8.

[10] N. H. Tran, A. D. Nguyen, and T. N. Nguyen, "A genetic algorithm application in planning path using B-Spline model for autonomous underwater vehicle (AUV)," *Appl. Mech. Mater.*, vol. 902, pp. 54–64, Sep. 2020, doi: 10.4028/www.scientific.net/AMM.902.54.

[11] 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, doi: 10.1016/j.isatra.2019.08.018.

[12] S. Mirjalili, G.-Ge Wang, and L. D. S. Coelho, "Binary optimization using hybrid particle swarm optimization and gravitational search algorithm," *Neural Comput. Appl.*, vol. 25, no. 6, pp. 1423–1435, 2014, doi: 10.1007/s00521-014-1629-6.

[13] A. Lin, W. Sun, H. Yu, G. Wu, and H. Tang, "Global genetic learning particle swarm optimization with diversity enhancement by ring topology," *Swarm Evol. Comput.*, vol. 44, pp. 571–583, Feb. 2019, doi: 10.1016/j.swevo.2018.07.002.

[14] M. D. Phung, C. H. Quach, Q. Ha, and T. H. Dinh, "Enhanced discrete particle swarm optimization path planning for UAV vision-based surface inspection," *Autom. Construct.*, vol. 81, pp. 25–33, Sep. 2017, doi: 10.1016/j.autcon.2017.04.013.

[15] H. S. Lim, S. Fan, C. K. H. Chin, and S. Chai, "Performance evaluation of particle swarm intelligence based optimization techniques in a novel AUV path planner," in *Proc. IEEE/OES Auto. Underwater Vehicle Workshop (AUV)*, Nov. 2018, pp. 1–7, doi: 10.1109/AUV.2018.8729773.

[16] E. Osaba, X. Yang, I. Fister, J. D. Ser, P. Lopez-Garcia, and A. J. Vazquez-Pardavila, "A discrete and improved bat algorithm for solving a medical goods distribution problem with pharmacological waste collection," *Swarm Evol. Comput.*, vol. 44, pp. 273–286, Feb. 2019, doi: 10.1016/j.swevo.2018.04.001.
[17] W. Lee and D. Kim, "Adaptive approach to regulate task distribution in swarm robotic systems," *Swarm Evol. Comput.*, vol. 44, pp. 1108–1118, Feb. 2019, doi: 10.1016/j.swevo.2018.11.005.
[18] F. Kang, J. Li, and Z. Ma, "Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions," *Inf. Sci.*, vol. 181, no. 16, pp. 3508–3531, 2011, doi: 10.1016/j.ins.2011.04.024.
[19] 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, doi: 10.1016/j.asoc.2014.09.046.
[20] C. Qu, W. Gai, M. Zhong, and J. Zhang, "A novel reinforcement learning based grey wolf optimizer algorithm for unmanned aerial vehicles (UAVs) path planning," *Appl. Soft Comput.*, vol. 89, Apr. 2020, Art. no. 106099, doi: 10.1016/j.asoc.2020.106099.
[21] D. H. Wolpert and W. G. Macready, "No free lunch theorems for search," Santa Fe Institute, Santa Fe, NM, USA, Tech. Rep. SFI-TR-95–02–010, 1995.
[22] V. Y. Pehlivanoglu, "A new vibrational genetic algorithm enhanced with a Voronoi diagram for path planning of autonomous UAV," *Aerosp. Sci. Technol.*, vol. 16, no. 1, pp. 47–55, Jan./Feb. 2012, doi: 10.1016/j.ast.2011.02.006.
[23] I. K. Ibraheem and F. H. Ajeil, "Path planning of an autonomous mobile robot using swarm based optimization techniques," *Al-Khwarizmi Eng. J.*, vol. 12, no. 4, pp. 12–25, Dec. 2017, doi: 10.22153/kej.2016.08.002.
[24] M. D. Phung and Q. P. Ha, "Safety-enhanced UAV path planning with spherical vector-based particle swarm optimization," *Appl. Soft Comput.*, vol. 107, Aug. 2021, Art. no. 107376, doi: 10.1016/j.asoc.2021.107376.
[25] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization*. Berlin, Germany: Springer, 2010, pp. 65–74, doi: 10.1007/978-3-642-12538-6_6.
[26] Z. Cui, F. Li, and W. Zhang, "Bat algorithm with principal component analysis," *Int. J. Mach. Learn. Cybern.*, vol. 10, no. 3, pp. 603–622, Mar. 2019, doi: 10.1007/s13042-018-0888-4.
[27] A. Baziar, A. Kavoosi-Fard, and J. Zare, "A novel self adaptive modification approach based on bat algorithm for optimal management of renewable MG," *J. Intell. Learn. Syst. Appl.*, vol. 5, no. 1, pp. 11–18, 2013, doi: 10.4236/jilsa.2013.51002.
[28] T.-K. Dao, T.-S. Pan, T.-T. Nguyen, and J.-S. Pan, "Parallel bat algorithm for optimizing makespan in job shop scheduling problems," *J. Intell. Manuf.*, vol. 29, no. 2, pp. 451–462, Feb. 2018, doi: 10.1007/s10845-015-1121-x.
[29] G.-G. Wang, H. E. Chu, and S. Mirjalili, "Three-dimensional path planning for UCAV using an improved bat algorithm," *Aerosp. Sci. Technol.*, vol. 49, pp. 231–238, Feb. 2016, doi: 10.1016/j.ast.2015.11.040.
[30] X. Zhou, F. Gao, X. Fang, and Z. Lan, "Improved bat algorithm for UAV path planning in three-dimensional space," *IEEE Access*, vol. 9, pp. 20100–20116, 2021, doi: 10.1109/ACCESS.2021.3054179.
[31] Y. Wang, K. Li, Y. Han, F. Ge, W. Xu, and L. Liu, "Tracking a dynamic invading target by UAV in oilfield inspection via an improved bat algorithm," *Appl. Soft Comput.*, vol. 90, May 2020, Art. no. 106150, doi: 10.1016/j.asoc.2020.106150.
[32] N. Lin, J. Tang, X. Li, and L. Zhao, "A novel improved bat algorithm in UAV path planning," *Comput., Mater. Continua*, vol. 61, no. 1, pp. 323–344, 2019.
[33] J. Shi, L. Tan, X. Lian, X. Lv, and T. Xu, "Multi-UAV task allocation method based on improved bat algorithm," in *Proc. Int. Conf. Artif. Intell. Secur.* Cham, Switzerland: Springer, 2021, pp. 205–213, doi: 10.1007/978-3-030-78609-0_18.
[34] X. S. Yang, "Bat algorithm for multi-objective optimisation," *Int. J. Bio-Inspired Comput.*, vol. 3, no. 5, pp. 267–274, 2012, doi: 10.1504/IJBIC.2011.042259.
[35] X. Ye, P. Wang, G. Xin, J. Jin, and Y. Huang, "Multi-scale quantum harmonic oscillator algorithm with truncated mean stabilization strategy for global numerical optimization problems," *IEEE Access*, vol. 7, pp. 18926–18939, 2019, doi: 10.1109/ACCESS.2019.2893200.

BUQIAN CHEN was born in 1998. He received the B.S. degree in information and computing science from Ludong University, in 2021. He is currently pursuing the M.S. degree with the Department of Mathematics, University of Shanghai for Science and Technology. His research interests include intelligent optimization algorithm and path planning.

JIN YANG received the B.S. degree in applied mathematics and the M.S. degree in operations research from Nanjing Normal University, in 2000 and 2003, respectively, and the Ph.D. degree in management science from the University of Shanghai for Science and Technology, in 2010. Since 2003, she has been working at the School of Science, University of Shanghai for Science and Technology. She has participated in several national and provincial-level projects in China and independently completed more than ten horizontal projects at top 500 companies. Her current research interests include intelligent optimization algorithms, optimization control, swarm intelligence, artificial intelligence, and global optimization.

HUIZHEN ZHANG received the Ph.D. degree in management science and engineering from the University of Shanghai for Science and Technology. She is currently an Associate Professor at the School of Management, University of Shanghai for Science and Technology. Her current research interests include operations research, swarm intelligence, hybrid algorithms, integer programming, and global optimization.

MENG YANG is currently pursuing the master's degree with the School of Science, University of Shanghai for Science and Technology. His research interests include recommendation systems, optimization algorithms, and artificial intelligence.