IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received December 28, 2019, accepted January 3, 2020, date of publication January 10, 2020, date of current version January 27, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2965579

Optimal Path Finding With Beetle Antennae Search Algorithm by Using Ant Colony Optimization Initialization and Different Searching Strategies

XIANGYUAN JIANG[1,2], ZONGYUAN LIN[2], TIANHAO HE[3], (Member, IEEE), XIAOJING MA[1], SILE MA[1], AND SHUAI LI[4], (Senior Member, IEEE)

[1]Institute of Marine Science and Technology, Shandong University, Qingdao 266237, China
[2]College of Control Science and Engineering, China University of Petroleum, Qingdao 266580, China
[3]University of California at San Diego, San Diego, CA 92093, USA
[4]College of Engineering, Swansea University, Swansea SA1 8EN, U.K.

Corresponding authors: Sile Ma (masile@sdu.edu.cn) and Shuai Li (shuaili@ieee.org)

This work was supported in part by the Fundamental Research Funds of Shandong University, the Youth Innovation Group Project of Shandong University (2020QNQT016) and the National Natural Science Foundation of China (61503413).

ABSTRACT Intelligent algorithm acts as one of the most important solutions to path planning problem. In order to solve the problems of poor real-time and low accuracy of the heuristic optimization algorithm in 3D path planning, this paper proposes a novel heuristic intelligent algorithm derived from the Beetle Antennae Search (BAS) algorithm. The algorithm proposed in this paper has the advantages of wide search range and high search accuracy, and can still maintain a low time complexity when multiple mechanisms are introduced. This paper combines the BAS algorithm with three non-trivial mechanisms proposed to solve the problems of low search efficiency and poor convergence accuracy in 3D path planning. The algorithm contains three non-trivial mechanisms, including local fast search, aco initial path generation, and searching information orientation. At first, local fast search mechanism presents a specific bounded area and add fast iterative exploration to speed up the convergence of path finding. Then aco initial path generation mechanism is initialized by Ant Colony Optimization (ACO) as a pruning basis. The initialization of the ACO algorithm can quickly obtain an effective path. Using the exploration trend of this path, the algorithm can quickly obtain a locally optimal path. Thirdly, searching information orientation mechanism is employed for BAS algorithm to guarantee the stability of the path finding, thereby avoiding blind exploration and reducing wasted computing resources. Simulation results show that the algorithm proposed in this paper has higher search accuracy and exploration speed than other intelligent algorithms, and improves the adaptability of the path planning algorithms in different environments. The effectiveness of the proposed algorithm is verified in simulation.

INDEX TERMS Beetle antennae search, optimal path finding, bio-inspired optimization, search orientation.

I. INTRODUCTION

Path planning [1] or dynamical motion tracking [2]–[4] have received more and more attentions of researchers and achieved rapid development due to its wide applications [5], [6] in recent decades, by following new intelligent optimization algorithms. Path planning in three-dimensional space can be summarized as the following problem: according to a certain requirement, find a collision-free path in the space that meets the requirements from the starting point to the target point. Path planning research is closely related to the practical application of mobile robots and drones. Not only in the industrial field, robots have strong application prospects in medical services, special explosion protection, field exploration, deep-sea exploration, and home services. All of these fields require robots to have more effective path

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

TABLE 1. The state of the art algorithms for path planning.

Intuitive Algorithms	Breadth First Search[7]
	Depth First Search[8]
	Dijkstra Algorithm[9]
Heuristic Algorithms	Genetic Algorithm[16]
	Particle Swarm Optimization[17]
	Ant Colony Optimization[18]
	Artificial Fish Swarm[19]
	Fruit Fly Optimization[20]
	Artificial Neural Network[26]
	A* Algorithm[27]
	Artificial Potential Field Method[28]
	Simulated Annealing[29]
Coverage Algorithms	Rapidly Exploring Random Tree[32]
	Backtracking Spiral Algorithm[33]
	Boustrophedon Decomposition[34]

planning techniques, which also challenges traditional path planning methods. In Table.1, the currently widely used intelligent algorithms for path planning are classified by group. According to the features of intelligent algorithms, they are clustered into three groups: intuitive algorithms, heuristic algorithms and coverage algorithms.

In terms of intuitive algorithms, Breadth-first search [7] and Depth-first search [8] are the most general graph-base search algorithms. Both of them start from the source point and find the optimal path by searching each vertex. Dijkstra algorithm [9] that appears later is also affected by them. Many heuristic algorithms [10]–[12] are presented via inspiration from biological behavior, revolutionary process, and even physical progress, etc., which could be used widely for different applications [13]. Wu *et al* added the BAS algorithm to the pre-training of the Artificial Neural Network (ANN). Compared with the gradient algorithm, the improved pre-training algorithm can effectively avoid local optimization and improve the speed of training. Wu *et al* also proposed the Obstacle Avoidance BAS (OABAS) algorithm for UAVs path planning, and the improved algorithm solved the contradiction between the high computational complexity of bio-heuristic algorithm and real-time planning, which significantly improves the path planning efficiency [14].

For path planning, the heuristic algorithms based on searching strategies [15] gradually show their advantages. Most of them are computationally intensive, and can quickly find an effective path and obtain the optimal path through continuous exploration. There are an amount of intelligent algorithms such as Genetic algorithm [16], Particle Swarm Optimization [17], Ant Colony Optimization [18], Artificial Fish Swarm algorithm [19] and Fruit Fly Optimization algorithm [20], which obtain near-global search results via group advantage. These different search strategies explore environment under heuristic guidance to move in the most promising direction. Li [21] proposed a discrete Fruit Fly Optimization algorithm and applied it to 3D path planning. Compared with other intelligent algorithms, it shortens the path exploration time and improves the search efficiency. Wang *et al.* [22] added the search mechanism of BAS algorithm to the iterative model of particle velocity, which PSO algorithm. Asif [23] summarizes the development and application of the ACO algorithm in detail, and also the ACO algorithm in detail, and also mentions the cases where ACO can be combined with other heuristic algorithms. Jaradat *et al.* [24] proposed a hybrid elite ant system for the nurse roster problem. This system uses external memory to retain the diversity of search results and has good performance in processing numerical datasets. This algorithm can be well applied to combinatorial optimization problems. Chen *et al.* [25] proposed a dynamic heterogeneous ant colony algorithm based on entropy, which uses entropy to measure diversity while enhancing the adaptability of the algorithm, and uses heterogeneous colonies to improve the convergence speed of the algorithm. This algorithm can effectively solve the large-scale traveling salesman problem.

Artificial neural network [26], A* algorithm [27], and Artificial potential field [28] and simulated annealing [29] is dominated by probability optimization, and the optimal solution is gradually obtained through multiple trains. Wu *et al.* [30] introduce BAS algorithm and fallback mechanism to achieve obstacle avoidance ability. The time complexity of this algorithm is extremely low and it can be used for short-term collision-free path planning of UAVs.

There are also other path planning algorithms depending on coverage or sampling strategies. Coverage algorithms [31] continue to link from one point to another, achieving optimization by covering the entire search space. LaValle [32] constructs a Rapidly-exploring Random Tree as a randomized data set for a broad class of path planning problems, which works well to handle non-holonomic constraints and high degrees of freedom. In coarse gain grid, spiral fill paths [33] are used as a coverage strategy to assure the completeness of path planning. Boustrophedon cellular decomposition [34] is presented for the purpose of driving the robot across each obstacle in the environment. It is able to search the optimal path accurately by not considering the dimensional constraints, while not judging whether the solution of path panning exists or not.

The Beetle Antennae Search algorithm (BAS) [11] is also a heuristic search algorithm, which is inspired by the predation behavior of beetles. The algorithm has good global search ability and convergence. Depending on the adaptive step size, it can effectively jump out of the local optimal value at the early stage of exploration and quickly converge at the end of the search. Therefore, it is very suitable for solving the discrete path optimization problem. Correspondingly, the BAS algorithm is also affected by the initial search step size and completely random search orientation, and requires complex initialization parameter tuning.

In this paper, we focus on the application of improved heuristic optimization algorithms to multi-objective requirements in 3D path planning. We propose a local fast search mechanism to reduce the impact of initial search step size and random optimization orientation. Compared with the original BAS algorithm, it enhances the global optimization capability of the BAS algorithm and avoids falling into local extreme values at the end of the search. Then, based on the efficient exploration ability of ACO algorithm, a hybrid mechanism that uses ACO algorithm to initialize the path is proposed. It combines with the iterative search of BAS algorithm, which not only enhances the search speed and convergence speed of ACO algorithm, but also extremely enhanced the real-time nature of the BAS algorithm. Finally, based on the swarm intelligence algorithm, a searching information orientation mechanism based on swarm information exchange is proposed, which is used to improve the stability of the algorithm while considering the search speed and search accuracy. It can be used to solve the multi-requirement path planning problem.

The main contributions of the path planning optimization algorithm proposed in this paper are as follows,

(a) This paper applies the BAS algorithm and its improved algorithms to the study of 3D path planning, expands the application of the BAS algorithm, and increases the diversity of path planning methods.
(b) ACO initialization in the structure of BAS strengthens both the completeness and the reach ability of path planning, as it imitates both the individual foraging behavior and swarm cooperation strategy well.
(c) Local fast search mechanism and the limitation of searching information orientation constitute optimum solution seeking for path planning.

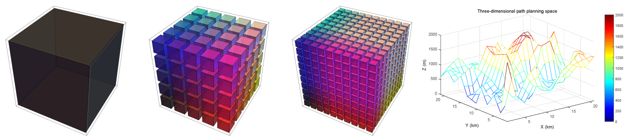
The next organization of this paper are as follows. In Section II, this paper will present preliminary information about both path planning and beetle antennae search. In Section III, this paper introduces the main structure and the proposes three complementary mechanism based on BAS, respectively, e.g. LFS-BAS and ACO-BAS, and SIO-BAS. In Section IV, we design experiments to validate the functions of the proposed algorithm. In Section V, a conclusion is drawn.

II. PRELIMINARY

In this section, we introduce the preliminary of both path planning and Beetle Antennae Search algorithm to serve for the coming optimization design in shortest path finding.

A. THE PATH PLANNING PROBLEM

The following methods are usually used to solve the path planning problem, such as using a continuous configuration space generated by the grid for graphic search, or random incremental search by random sampling. We pick up graph-based environment to build configuration space. In terms of 3D space as shown in Fig.1, at first, a complete 3D space is shown in Fig.1a. Then we use the grid method to



(a)3D space. (b)Coarse division. (c)Fine division. (d)The generated model.

FIGURE 1. 3D space model generated for path planning.

coarsely divide the space into limited space blocks as Fig.1b. Finally, in Fig.1c, as the space block becomes smaller and smaller, we obtain a fine space model that can describe the 3D terrain as Fig.1d. We take the coordinates of the cartesian coordinate system as $x-axis$, $y-axis$ and $z-axis$ respectively, the upper limits of their values are I, J and K, the contour points of the obstacle can be expressed as follows,

$$E_{point} = \{(i, j, k)|i \in [0, I], j \in [0, J], k \in [0, K]\} \tag{1}$$

where (i, j, k) denotes the grid point after space meshing, it is the effective value point in the path planning, and I, J, K represent the maximum discrete number along each coordinate. The X coordinates can be integrated into the iteration, which can effectively reduce the calculation matrix dimension. The lack of an X coordinate also avoids the occurrence of a backtracking point. Each set of Y and Z coordinates constitutes a path point. For example,

$$\boldsymbol{x} = [Y_1, Z_1, \cdots, Y_J, Z_K] \tag{2}$$

where $\boldsymbol{x}$ denotes a valid search path, and coordinate grid are represented by $Y \in [1, J]$, $Z \in [1, K]$.

The fitness evaluation function mainly considers weighted length of the path, so the distance of the path is the main evaluation basis. The fitness function is shown in detail as follows.

$$f = \sum_{i=2}^{J}(\sqrt{1 + (Y_i - Y_{i-1})^2 + (Z_i - Z_{i-1})^2} + Z_i) \tag{3}$$

Equation (3) shows that the path with the smaller fitness function value is better.

B. BEETLE ANTENNAE SEARCH ALGORITHM

The BAS algorithm is a meta heuristic search algorithm and it is inspired by the predation behavior of the beetle. Fig.2 shows BAS algorithm in a schematic diagram. As shown in the figure, we can compare a triangle to a beetle, the two corners on the long side of the triangle as the antennae, the line between the triangles as the route and distance of the movement, and the light blue area indicates the odor concentration. Then, we can visually get a conceptual picture of the beetle predation behavior. During the predation process, a beetle receives the pheromone scattered in the air through the odor sensors on the two antennae. When the distance between the beetle of the prey changes, the concentration of the pheromone received accepted by the beetle also changes. Therefore, when the concentration of pheromone felt by the

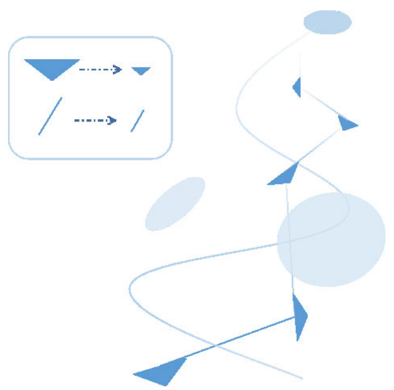


FIGURE 2. The searching process of BAS for optimal solution with bio-inspired behavior.

two antennae are different, the beetle will advance to the side with the high concentration of pheromone according to the habit, and by continuously moving and detecting the concentration of the pheromone, the beetle can approach the prey continuously, and finally arrives at the source point.

The mathematical model of the BAS algorithm is as follows,

1) Establish the fitness function $f(\boldsymbol{x}^t)$, where variable $\boldsymbol{x}^t = [\boldsymbol{x}^1, \cdots, \boldsymbol{x}^n]$, t denotes the number of iterations.
2) Randomize the orientation of the beetle,
$$\overrightarrow{\boldsymbol{b}} = \frac{rand(k,1)}{||rand(k,1)||} \tag{4}$$
where $rand(,)$ denotes the random orientation of the beetle, it is a matrix of $k \times 1$ dimensions, and parameter k is according to the dimensions of independent variable. $\overrightarrow{\boldsymbol{b}}$ is an unit vector.
3) The coordinates of the two antennae obtained in each iteration,
$$\begin{aligned} \boldsymbol{x}_r &= \boldsymbol{x}^{t-1} + d^{t-1}\overrightarrow{\boldsymbol{b}} \\ \boldsymbol{x}_l &= \boldsymbol{x}^{t-1} - d^{t-1}\overrightarrow{\boldsymbol{b}} \end{aligned} \tag{5}$$
where $\boldsymbol{x}_r$ and $\boldsymbol{x}_l$ denotes the coordinate of right antenna and left antenna respectively. Parameter d is the sensing length of antennae which can be decided by step δ.
$$\delta^t = \lambda_1 \delta^{t-1} \tag{6}$$
where λ_1 is a constant, it determines the decay speed of the search step.
$$d_0^t = \delta^t / c \tag{7}$$
where c is also a constant, it will affect the changes of $\boldsymbol{x}_l$ and $\boldsymbol{x}_r$.
4) By solving the above coordinates, it will get an iterative formula of the independent variable,
$$\boldsymbol{x}^t = \boldsymbol{x}^{t-1} - \delta\overrightarrow{\boldsymbol{b}}\, sign(f(\boldsymbol{x}_l) - f(\boldsymbol{x}_r)) \tag{8}$$
where the coordinate update of $\boldsymbol{x}^t$ is related to the coordinates of the previous step $\boldsymbol{x}^{t-1}$. Where sign(.) represents a sign function, it can help to compare the value of the fitness function on both the left and the right. $f(\boldsymbol{x}_l)$ and $f(\boldsymbol{x}_r)$ represent the fitness values of the left and right sides of $\boldsymbol{x}$ respectively, their connection with f_{bst} is shown in Fig.9a. The search step size δ is an attenuation variable associated with the number of iterations t. A suitable step attenuation strategy helps the algorithm converge without falling into extreme values too early.

III. THE PATH PLANNING DESIGN

The proposed path planning algorithm with BAS consists of three main parts: local fast search, ant colony initialization and path correction with searching information orientation. Each aforementioned part could be seen as a complementary to the BAS, individually, in other words, each one accompanying with BAS works well to improve the task of path planning.

A. LOCAL FAST SEARCH

The proposed local fast search introduces an important mechanism to boost the searching efficiency of BAS through adaptive search step size and random orientation. The local fast search embedded into BAS is named as Local Fast Search - Beetle Antennae Search (LFS-BAS), and its pseudo code is shown in Algorithm. 1.

When the BAS algorithm is used to deal with high-dimensional multi-extremum problems, although the large step size can effectively search large-scale, it will miss the possibility of searching for the global optimal solution. Meanwhile, the decay rate of the step size will also significantly affect the global search accuracy for the optimal solution. Therefore, on the premise of ensuring that the algorithm can still jump out of the local optimum, and in order to reduce the impact of the step size of the BAS algorithm on path planning and improve search accuracy, we have designed an LFS that can quickly search in the potential global optimal solution area. The main idea of LFS is to lock the step size in the current iteration process and derive a local search step size. This local search step size only exists in the LFS, and the adaptive attenuation of the step size is completed in the LFS. Hence, a new variable δ_η is added for BAS to get better search results. At the same time, it does not affect the search step δ of the algorithm at all.
$$\begin{aligned} \delta_\eta^0 &= \delta^t \\ \delta_\eta^i &= \lambda^{b/(a+\gamma i/m)} \cdot \delta_\eta^{i-1},\ i \in [0, m] \end{aligned} \tag{9}$$
where tth denotes the current number of iterations of the main program, ith denotes the number of iterations of the LFS, a, b, and γ are all constants, and their size affects the decay rate of δ_η. m denotes the maximum number of LFS iteration, and the choice of its size needs to balance search accuracy with search speed, $\lambda \in [0, 1]$. δ_η is an independent variable, which is initialized by the δ and then attenuated independently.

Algorithm 1 Local Fast Search BAS Algorithm

Input: Establish an objective function $f(\boldsymbol{x}^t)$, where variable $\boldsymbol{x}^t = [x_1, \cdots, x_k]^{\mathrm{T}}$, initialize the parameters $\boldsymbol{x}^0$, d^0, δ^0, M.
Output: $\boldsymbol{x}^*_{\mathrm{bst}}, f^*_{\mathrm{bst}}$.
while *($\boldsymbol{x}_j^0$= Invalid)* **do**
  Adjust the boundaries of the C_{ini} space accounding to (12);
  Randomly generate an initial path $\boldsymbol{x}_j^0$ in new C_{ini} space;
  $\boldsymbol{x}_j^0$.
while *($t < T_{max}$) or (stop criterion)* **do**
  Generate the direction vector unit $\overrightarrow{\boldsymbol{b}}$ according to (4);
  Search in variable space with two kinds of antennae according to (5);
  Update the state variable $\boldsymbol{x}^t$ according to (8);
  if $f(\boldsymbol{x}^t) < f_{bst}$ **then**
    $f_{bst} = f(\boldsymbol{x}^t), \boldsymbol{x}_{\mathrm{bst}} = \boldsymbol{x}^t$.
  if $f(\boldsymbol{x}^t) = f_{bst}$ **then**
    while *($i < M_{max}$)* **do**
      $\delta^i = \delta^t, \boldsymbol{x}^i = \boldsymbol{x}^t, f_\eta = f(\boldsymbol{x}^t)$;
      Generate the direction vector unit $\overrightarrow{\boldsymbol{b}}$ according to (4);
      Search in variable space with two kinds of antennae according to (10);
      Update the state variable $\boldsymbol{x}^i$ according to (11);
      if $f(\boldsymbol{x}^i) < f_\eta$ **then**
        $f_{\mathrm{new}} = f(\boldsymbol{x}^i)$;
      Update step size δ^i according to (9).
    $f^n_{gbst} = f_\eta$.
  Update step size δ and sensing diameter d with decreasing functions (6) and (7) respectively.
  f^*_{bst}=min($f_{bst}, f^1_{gbst}, \cdots, f^n_{gbst}$), corresponding $\boldsymbol{x}^*_{bst}$.
return $\boldsymbol{x}^*_{bst}, f^*_{bst}$.

Similarly, in the section of local fast search, the iteration of $\boldsymbol{x}$ needs to be redesigned as follows,

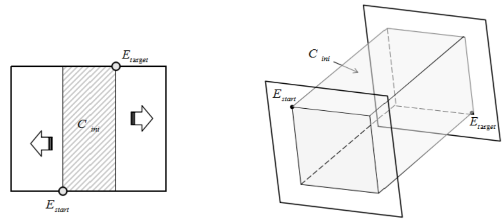
$$\begin{aligned} \boldsymbol{x}_r &= \boldsymbol{x}^{i-1}_{bst} + \delta^{i-1}_\eta \overrightarrow{\boldsymbol{b}} \\ \boldsymbol{x}_l &= \boldsymbol{x}^{i-1}_{bst} - \delta^{i-1}_\eta \overrightarrow{\boldsymbol{b}} \end{aligned} \tag{10}$$

where $\boldsymbol{x}_{bst}$ denotes the optimal value in the LFS. This part still retains the core heuristic search of the BAS algorithm.

$$\boldsymbol{x}^i_{bst} = \boldsymbol{x}^{i-1}_{bst} - \delta^{i-1}_\eta \overrightarrow{\boldsymbol{b}}\,\mathrm{sign}(f(\boldsymbol{x}_l) - f(\boldsymbol{x}_r)) \tag{11}$$

where $\boldsymbol{x}_{bst}$ is an iterative model of the LFS, which is similar to the iteration of the BAS algorithm, mainly to preserve the core search procedure of the algorithm.

To speed up generation of the initial effective path, we propose a constraint path generation method, which based on the



(a)2D effective expansion space. (b)3D effective expansion space.

FIGURE 3. Effective extension space for initial path selection.

Informed method [35] and Connectivity Region method. The constraint path generation method applied to the BAS algorithm is to establish a gradually expanding solution space C_{ini}. The space C_{ini} for the initial effective path is related to the initial point and the target point. It give bounded area around the line from the initial to the target point, and its illustration is shown in Fig.3, respectively. Then it is evident that the optimal path will be included in this bounded area with high possibility. To reduce the calculation of the algorithm, the lower bound of the Y coordinate in C_{ini} space is denoted as c_l, and the upper bound is denoted as c_u. The linear expansion rule of C_{ini} space is as follows,

$$\begin{aligned} c_l' &= c_l - c' \\ c_u' &= c_u + c' \end{aligned} \tag{12}$$

where the size of c' depends on the complexity of the search environment, the c' in this paper we set is 0.0007, which can balance the calculation speed and accuracy.

The flow of the LFS-BAS algorithm is as follows:

1) Initialize parameters, specify path start point and target point.
2) Generate an initial random path until the path $\boldsymbol{x}$ is valid, and calculate the fitness value $f(\boldsymbol{x})$ of the initial path. Stored as optimal path $\boldsymbol{x}_{bst}$, optimal fitness value f_{bst}, global optimal path $\boldsymbol{x}_{gbst}$, global optimal fitness value f_{gbst}.
3) Through BAS explores a new valid path, and calculate the fitness value of the path. By logically judging the fitness function value of the path $f(\boldsymbol{x})$ with optimal fitness value f_{bst}, and retains the path with a smaller value.
4) If the search for the path fitness value is larger than f_{bst}, skip this step and continue execution; If the search for the path fitness value is smaller, then go to the local fast search. After searching for a better path, it is stored as a global optimal path $\boldsymbol{x}_{gbst}$, and the corresponding value also stored as global optimal fitness value f_{gbst}.
5) Join the path rationality check. Calculate the coordinate difference between the previous path point and the next path point $(Y_i - Y_{i-1})^2$, compare the coordinate difference wether satisfied the reasonable evaluation step θ^t, if it is not satisfied, the following replacement

is required, where $i \in [2, 21]$,

$$Y_i = Y_{i-1}$$
$$Z_i = Z_{i-1} \tag{13}$$

6) Reassigning the global optimal path and the global optimal fitness value, then the iteration ends. The search step size δ and the reasonable evaluation step size θ are attenuated, then determine if the maximum number of iterations is reached. If yes, exit the iteration; if not, go to step 3 and continue the iterative operation.

$$\theta^t = \lambda_2 \theta^{t-1} \tag{14}$$

where λ_2 is a constant and given by experience.

7) Give the optimal path and optimal fitness evaluation function value.

B. ANT COLONY INITIALIZATION

We use C-space method [36] to generate effective path points in the whole space, and find that exploring the potential optimal path of the entire space from the effective path can greatly reduce the probability of get the invalid path. Considering ACO algorithm, we found that it obtains the relative optimal path through the population search at the beginning, but then it needs to accumulate pheromone through a large number of repeated paths. Therefore, we try to generate an initial path based on the ACO algorithm, and then perform a fast search through the BAS algorithm. Finally, we obtained a hybrid algorithm with two advantages, namely the Ant Colony Optimization - Beetle Antennae Search (ACO-BAS), pseudo code as shown in Algorithm.2.

The ACO-BAS algorithm proposed in this paper can be divided into two parts, that is, using the ACO algorithm to initialize the path, and using this path to continue to find potential solutions through BAS. Its flow is as follows,

1) Perform parameter initialization, set the population size n, set the start point and target point coordinates, and determine the ant colony search direction.
2) Each ant in the population calculates the heuristic information value of each point in the next region according to the heuristic function.

$$H = S \times D \times M \tag{15}$$

where S denotes the point valid or not, D denotes the distance between the point and the current point, and M denotes the height information of the point.

3) Each ant in the population moves to a randomly selected path point. Iterative 21 times to complete a population path generation. Then select the path with the smallest fitness function value as the initial path.
4) Stored the initial path and its fitness value as optimal path and optimal fitness value.
5) Generate a random direction, use the BAS algorithm to search for potential paths, calculate the fitness value of the potential path, and save the better path as the optimal path.

Algorithm 2 ACO-BAS Algorithm for 3D Path Planning

Input: Establish the fitness function $f(\boldsymbol{x}^t)$, where variable $\boldsymbol{x}^t$=[$\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{2i}$], $i \in [2, 21]$, initialize the parameters $\boldsymbol{x}^0$, δ^0, c, θ^0, n.
Output: $\boldsymbol{x}_{bst}, f_{bst}$.
while $(t < T_{max})$ **do**
  Set the start point and target point, population size n, determine the ACO population search direction;
  for j=1:21 **do**
    Determine the next path point $[Y_j, Z_j]$ based on the heuristic function $H = S * D * M$;
    The number of generated paths is n, calculate their fitness values;
    Store the path corresponding to the minimum fitness value $\boldsymbol{x} = [Y_1, Z_1, \cdots, Y_{21}, Z_{21}]$.
  Generate orientation vector $\overrightarrow{\boldsymbol{b}}$ according to (4);
  Acquire the fitness of the left path and right path fitness functions using two antennae according to (5);
  Update centroid coordinates $\boldsymbol{x}^t$ according to (8);
  for i=2:21 **do**
    if $(Y_i - Y_{i-1})^2 > \theta^2$ **then**
      Update path information according to formula (13);
    Return a smooth $\boldsymbol{x}$.
  if $f(\boldsymbol{x}^t) < f_{bst}$ **then**
    f_{bst}=$f(\boldsymbol{x}^t)$, $\boldsymbol{x}_{bst}$=$\boldsymbol{x}^t$.
  Update the sensing diameter d , attenuation spacing θ and step size δ according to (14) and (6), respectively.
return $\boldsymbol{x}_{bst}, f_{bst}$.

6) Perform path rationality verification. Calculate the distance of adjacent path points. If it is greater than the reasonable value, adjust according to (13).
7) Returns the optimal path and optimal fitness value.

C. PATH CORRECTION WITH SEARCHING INFORMATION ORIENTATION

In this subsection, searching information orientation is employed to correct and bias the path planning, named as swarm intelligence optimization - Beetle Antennae Search (SIO-BAS), by using multiple beetles with fusion information in path exploration. They exchange the path information with each other to obtain the better path information, and use this information as an influencing factor in the next direction of exploration. For the sake of comparison and explanation, this paper refers to the BAS group without information sharing behavior.

The SIO-BAS algorithm performs the expansion of the search particle based on the generalized BAS [37]. The WPT method normalizes the parameters of different value ranges to the same range, which simplifies the difficulty and complexity of parameter tuning. This number of expansion is not only a superposition of multiple repeated searches, but a fusion of information sharing behavior which similar to the PSO algorithm. This allows the information of the best individuals to be widely disseminated in the group, that improving the stability of the whole search. Its pseudo code is shown in Algorithm.3.

The searching information orientation is designed as follows,

$$\begin{aligned} \overrightarrow{b}^t &= \frac{\overrightarrow{b}^{t*}}{||\overrightarrow{b}^{t*}||} \\ \overrightarrow{b}^{t*} &= \eta_1 \overrightarrow{b}^t_1 + \eta_2 \overrightarrow{b}^t_2 + \eta_3 \overrightarrow{b}^t_3 \end{aligned} \tag{16}$$

where $\overrightarrow{b}^t$ denotes the search direction of each individual in tth search, and η_1, η_2, and η_3 denote the composition ratios of the three direction vectors, respectively. The mathematical meanings they represent are explained in detail below. Where $\overrightarrow{b}^t_1$ is the same as equation (4), and denotes a randomly generated unit vector for reducing the probability of falling into a local optimal solution. Where $\overrightarrow{b}^t_2$ denotes the difference value unit vector between the current group optimal path and the individual current path information, indicating the current information of the optimal direction, equation (17) is its mathematical expression representation. Where $\overrightarrow{b}^t_3$ denotes the direction vector of the dominant side in the previous generation of search information guidance. Recording the historical search of orientation can enhance the search and optimization of beetle in local areas. It is used to indicate fluctuations in the search path on the left and right, and to help the convergence of search results. In this way, it is able to obtain search direction information of the same order of magnitude and with certain directivity.

$$\overrightarrow{b}^t_2 = \frac{\boldsymbol{x}^{t-1}_{gbst} - \boldsymbol{x}^{t-1}}{||\boldsymbol{x}^{t-1}_{gbst} - \boldsymbol{x}^{t-1}||} \tag{17}$$

where $\boldsymbol{x}^{t-1}_{gbst}$ represents the optimal group path information and $\boldsymbol{x}^{t-1}$ denotes the individual current path information.

$$\overrightarrow{b}^t_3 = \frac{sign(f(\boldsymbol{x}_l) - f(\boldsymbol{x}_r))(\boldsymbol{x}_r - \boldsymbol{x}_l)}{||sign(f(\boldsymbol{x}_l) - f(\boldsymbol{x}_r))(\boldsymbol{x}_r - \boldsymbol{x}_l)||} \tag{18}$$

where $sign(f(\boldsymbol{x}_l) - f(\boldsymbol{x}_r))$ is used to indicate the optimal direction in the left and right sides of search beetle.

Algorithm 3 SIO-BAS Algorithm

Input: Establish an objective function $f(\boldsymbol{x}^t)$, where variable $\boldsymbol{x}^t = [x_1, \cdots, x_k]^{\mathrm{T}}$, initialize the parameters $\boldsymbol{x}^0$, d^0, δ^0, M, C_{ini}, pop.
Output: $\boldsymbol{x}^*_{\mathrm{bst}}, f^*_{\mathrm{bst}}$.
while *($j < pop$)* **do**
  if *($\boldsymbol{x}^0_j$= Invalid)* **then**
    Adjust the boundaries of the C_{ini} space accounding to (12);
    Randomly generate an initial path $\boldsymbol{x}^0_j$ in new C_{ini} space;
    $\boldsymbol{x}^0_j$
  $\boldsymbol{x}^0_1, \cdots, \boldsymbol{x}^0_{pop}$, and initialize j.
while *($t < T_{max}$) or (stop criterion)* **do**
  while *($j < pop$)* **do**
    Generate the direction vector unit $\overrightarrow{b}$ according to (16);
    Search in variable space with two kinds of antennae according to (5);
    Update the state variable $\boldsymbol{x}^t_j$ according to (8);
    for *i=2:21* **do**
      if $(Y_i - Y_{i-1})^2 > \theta^2$ **then**
        Update path information according to formula (13);
      Return a smooth $\boldsymbol{x}_j$.
    if $f(\boldsymbol{x}^t_j) < f_{jbst}$ **then**
      $f_{jbst} = f(\boldsymbol{x}^t)$, $\boldsymbol{x}_{\mathrm{jbst}} = \boldsymbol{x}^t_j$
    Get the optimal path information in the population, update $\overrightarrow{b}^t_2$ according to formula (18).
  Update step size δ and a reasonable evaluation step size θ with decreasing functions (6) and (14), respectively;
  f^*_{bst}=min($f_{bst}, \cdots, f_{popbst}$), corresponding $\boldsymbol{x}^*_{bst}$.
return $\boldsymbol{x}^*_{bst}, f^*_{bst}$.

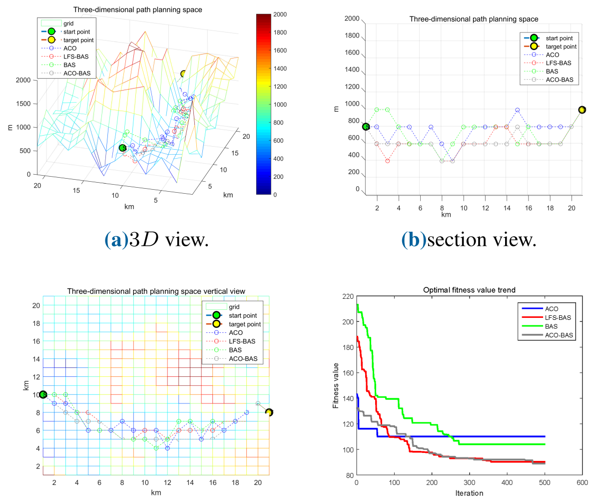
IV. EXPERIMENTAL VALIDATION

A. EXPERIMENTS WITH PROPOSED ALGORITHMS IN 3-D SITUATION

The experiments in this part are the path planning experiments of ACO, BAS, LFS-BAS and ACO-BAS algorithms in 3D space.

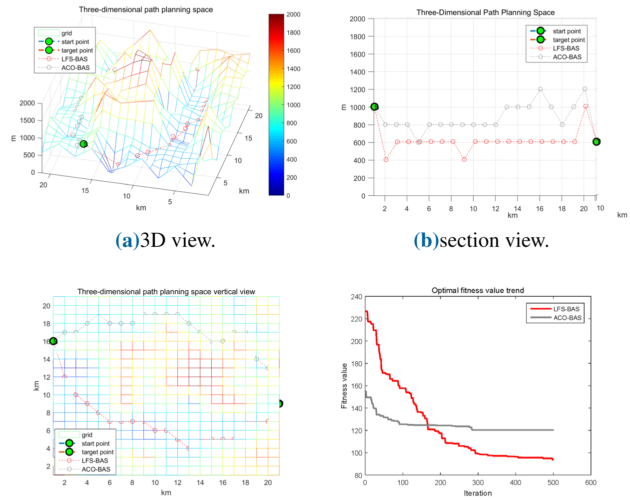
In this part, the path planning of the four algorithms in 3D space is tested separately. The final result is shown by fig.4. The maximum number of iterations of each algorithm is 500, and the population size of ACO is 20, The initial step size of the BAS-related algorithm is set to 8, the reasonable evaluation step size is set to 21, and the two attenuation coefficients λ_1 and λ_2 are both 0.998, c set 2, and μ set 0.998. After 500 iterations, the step size is finally attenuated to 2.94, and the reasonable evaluation step is attenuated to 7.71. The optimal path of the experiment is along the curved area on the right side of the canyon. Meanwhile, we also conducted three experiments on different iterations of the four algorithms. The data is recorded 10 times for each iteration, as shown in Table.3. After averaging 10 calculations, we recorded the data with significant changes to Table.2.

Experimental results show that our proposed LFS-BAS algorithm and ACO-BAS algorithm can efficiently complete the path planning work within a limited number of iterations. All four algorithms can search for the optimal path, but the



(a)3D view. (b)section view.
(c)top view. (d)optimal fitness value trend.

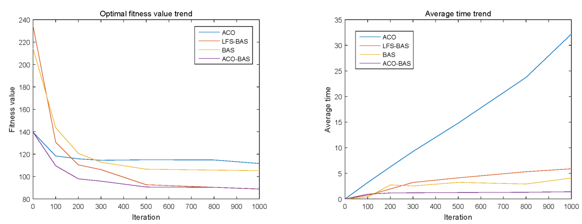
FIGURE 4. **The comparison of ACO, LFS-BAS, BAS and ACO-BAS algorithms in 3D path planning, the optimal fitness values are 110.0917, 90.3271, 103.9980 and 88.9594 respectively, and the corresponding iteration time are 14.2730s, 4.6728s, 2.7311s and 1.2537s.**



(a)3D view. (b)section view.
(c)top view. (d)optimal fitness value trend.

FIGURE 6. **The comparison of LFS-BAS and ACO-BAS algorithms in 3D path planning, the optimal fitness values are 93.6522 and 120.2932 respectively, and the corresponding iteration time are 2.6312s and 0.9764s.**

final fitness value of ACO and BAS algorithm is still larger than the other two algorithms. We guess that the path they get is more unstable and less smooth, and it is necessary to filter out the path points with large fluctuations. The comparison of the data in the table shows that the LFS-BAS algorithm and the ACO-BAS algorithm have a significant improvement in the final fitness value, 19.97% and 21.75% have been improved respectively, and the iterative calculation time has also shortened 34.30% and 80.07%.



(a)Optimal fitness value trend. (b)Average time trend.

FIGURE 5. **Trend figure of optimal fitness values and iteration time of ACO, LFS-BAS, BAS and ACO-BAS with increasing number of iterations.**

After data processing, the algorithm fitness trend under different iteration times is shown in Fig.5. As shown in the Fig.5a, the ACO algorithm basically finds the optimal path of the minimum fitness value after 200 iterations, while other BAS-related algorithms must find the optimal path after 500 iterations. However, the number of iterations can only reflect a certain degree of computational complexity. As can be seen from Fig.5b, the time after the 500 iterations of the BAS-related algorithm is still much less than the time taken by the ACO after 200 iterations. Compared with the ACO algorithm, the main reason that the calculation time is not significantly increased with the increased iterations is that the search path is almost all valid paths, which reduces the repeated exploration. And because the paths after the left and right exploration are almost the same, a lot of calculation time is saved. The analysis results show that BAS-related algorithms have extremely efficient computing ability. Therefore, for BAS-related algorithms, a larger number of iterations can obtain a better path without too much computation, which is beneficial for further optimization of path planning with non-convex bounded constraint [38].

In order to further test the search ability of LFS-BAS and ACO-BAS algorithm, this part adds another experiment. This experiment examines whether the algorithm will fall into the local optimal solution on the left side of the canyon by changing the path start point. The results are shown in Fig.6. The ACO-BAS algorithm may fall into the local optimal solution, but the LFS-BAS can still get the optimal path. Experimental results show that if the initial path generated by the ACO-BAS algorithm is too close to the local optimal solution, the possibility of subsequent searches jumping out of the local optimal will be greatly reduced. Nevertheless, the search speed of the ACO-BAS algorithm is much faster than the other three algorithms.

In order to explore the effectiveness of searching information orientation mechanism on group optimization, this paper conducted a comparative experiment with non-information sharing groups. In this experiment, SIO-BAS represents a group with searching information orientation mechanism, and Multi-BAS represents a group composed of individuals without searching information orientation. In addition, the parameter settings of the two algorithm are the same, the population number is 5, and the maximum number of iterations is 200. The only difference between them is whether there is searching information orientation mechanism. The experiment was repeated 10 times, the optimal path information of the experiment and the results of each experiment

TABLE 2. Comparision of optimization results from ACO, BAS, LFS-BAS and ACO-BAS.

name	iteration	ave_fitness	fitness ratio	ave_time(s)	time ratio
ACO	200	115.82	100%	6.27	100%
ACO	500	112.98	97.55%	14.94	238.28%
BAS	500	106.56	92.00%	3.23	52.00%
LFS-BAS	500	92.7	80.03%	4.12	65.70%
ACO-BAS	500	90.64	78.25%	1.25	19.93%

TABLE 3. Comparison details between ACO, LFS-BAS, BAS and ACO-BAS.

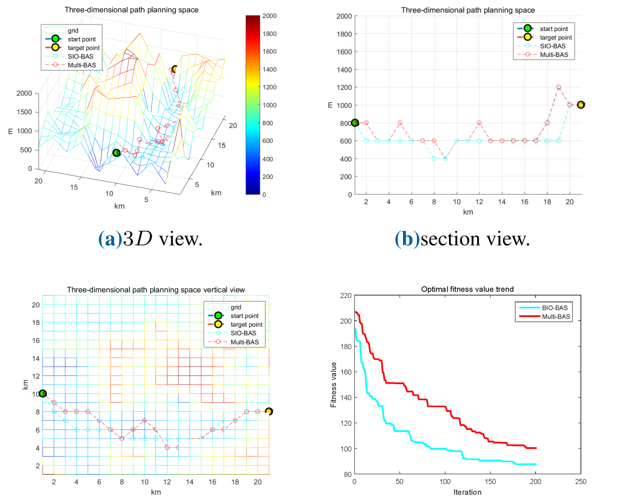
Ite	$BestFitness$				$ElapsedTime(s)$			
	ACO	LFS-BAS	BAS	ACO-BAS	ACO	LFS-BAS	BAS	ACO-BAS
200	111.93	113.48	128.16	94.60	6.28	1.62	1.13	1.95
	112.19	107.88	114.21	118.96	6.06	2.31	2.59	1.12
	118.53	112.23	120.56	102.46	6.02	1.99	6.49	1.10
	116.52	115.01	121.23	103.12	6.04	1.65	3.34	1.01
	114.63	118.25	118.15	90.19	6.02	1.37	1.76	1.13
	114.03	117.75	124.44	98.25	6.11	1.42	3.05	1.03
	112.50	101.30	116.01	94.61	6.37	2.86	1.64	1.13
	116.29	101.56	120.33	95.19	6.12	2.31	2.14	1.15
	115.98	100.61	114.01	90.98	5.96	2.93	3.17	1.12
	113.00	116.55	130.20	91.97	6.00	1.13	2.17	1.16
500	109.85	88.95	105.34	87.63	14.78	5.40	6.45	1.27
	110.72	90.72	113.39	91.09	15.04	4.49	3.53	1.23
	111.29	91.19	95.86	89.23	14.74	4.02	2.43	1.23
	115.30	92.50	106.39	90.22	14.76	4.79	1.07	1.23
	113.82	93.50	104.26	88.43	14.82	6.14	3.42	1.25
	117.43	92.90	114.14	91.76	15.21	4.19	1.80	1.26
	115.83	93.42	105.47	95.02	14.92	5.69	3.62	1.29
	111.51	90.95	103.76	92.06	14.95	5.26	2.50	1.24
	109.05	103.11	109.33	92.13	14.85	5.93	5.11	1.25
	114.96	90.04	108.04	89.22	15.33	5.29	2.43	1.25
800	112.89	88.03	113.42	89.22	25.31	6.13	3.71	1.40
	116.83	89.05	106.83	88.13	24.50	5.78	3.87	1.34
	114.50	89.86	107.34	90.11	24.23	4.95	1.45	1.31
	116.20	90.64	107.02	88.59	24.01	5.01	1.53	1.29
	113.25	91.53	103.78	91.99	23.85	4.96	1.77	1.32
	113.51	88.22	101.70	87.82	23.82	5.22	2.69	1.31
	117.26	88.49	112.94	95.84	23.42	5.15	8.06	1.30
	115.87	89.79	97.18	92.80	24.23	4.61	0.98	1.31
	114.12	98.56	103.81	88.22	23.71	5.90	2.87	1.32
	113.76	90.19	104.52	91.36	23.55	5.06	2.81	1.30

TABLE 4. Comparision of optimization results from SIO-BAS, Multi-BAS, and ACO.

name	ite	pop	bestfitness	ave_fitness	std_fitness	ave_time(s)
SIO-BAS	200	5	89.93	100.44	22.03	4.87
Multi-BAS	200	5	103.75	117.31	31.71	3.39
ACO	200	20	111.56	115.82	10.93	6.27

were recorded, and the average time of the experiment was obtained. Fig.7 shows the experimental results from the 3D view, the sectional view and the top view respectively, and finally gives the trend of the fitness value of the respective optimal paths. The detailed result analysis is given in Table.4.

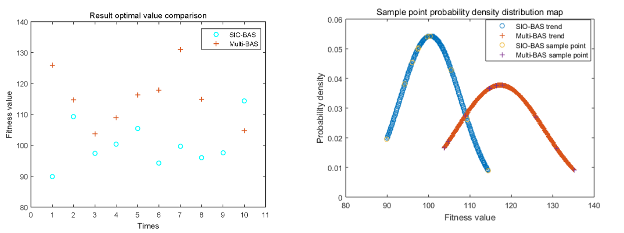
It can be seen from the experimental results in Table.4 that the fitness value of the optimal path obtained by algorithm SIO-BAS is 89.93, which is 13.32% higher than that obtained by algorithm Multi-BAS under the same conditions, indicating that the algorithm SIO-BAS has higher precision and shorter path. The average value of the path fitness value obtained by algorithm SIO-BAS is 100.44, which is lower than the path obtained by algorithm Multi-BAS, and the standard deviation obtained by algorithm SIO-BAS after 10



(a)3D view. **(b)**section view.

(c)top view. **(d)**optimal fitness value trend.

FIGURE 7. **The comparison of SIO-BAS and Multi-BAS algorithms in 3D path planning, the optimal fitness values are 89.9321 and 103.7597 respectively, and the corresponding iteration time are 4.8782s and 3.3959s. The population is 5 and the maximum number of iterations is 200.**

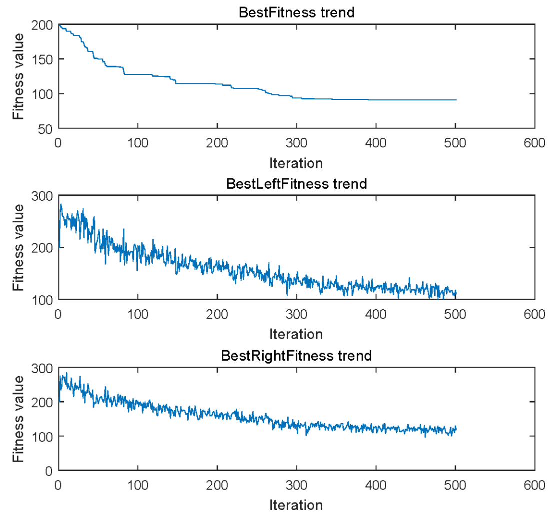


(a) The final fitness value. **(b)**Probability density map.

FIGURE 8. **Comparison of the optimal values of SIO-BAS and Multi-BAS in 10 times, and their sample point probability density distribution map. Their average estimates are 100.4449 and 117.3193, their estimates of the standard deviation in the normal distribution are 7.3452 and 10.5721.**

experiments is 22.03, which is 30.53% lower than that of algorithm Multi-BAS, indicating that it has better stability. Moreover, Fig.8 shows their discreteness more intuitively. In terms of computation time, the algorithm SIO-BAS is 43.66% larger than the algorithm Multi-BAS, but considering that the ACO algorithm calculates the time at 6.27s in 200 iterations (the population is 20, the average fitness value is 115.82), the both of calculation results are within the expected range. Experimental results show that the SIO-BAS algorithm with searching information orientation mechanism has better stability and search accuracy. This means that when we are solving offline optimization problems with less real-time performance, we can use the searching information orientation mechanism in the group to enhance the stability of optimization, and our future work may be invested in this aspect.

Fig.9 reflects the change in fitness values of the centroid and the left and right fitness values during the iteration. They are consistent with the number of iterations, indicating that the algorithm has good convergence. At the same time,



(a)f_{bst}, f_l and f_r.

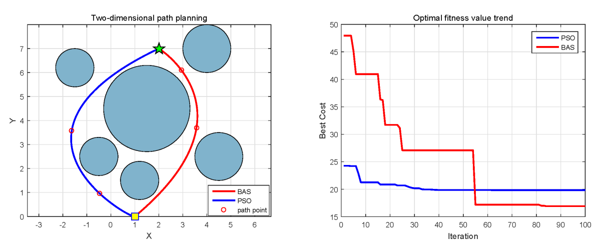
FIGURE 9. **The fitness iteration trend of f_{bst}, f_l and f_r in a certain path planning. The three exploration trends are the same, and the convergence is effective.**

the fitness values on the left and right sides jump near the optimal fitness, indicating that the algorithm has an effective search mechanism and is sensitive.

B. EXPERIMENTS WITH PROPOSED ALGORITHMS IN 2-D SITUATION

This experimental part of this paper includes two simulation experiments, namely the path planning experiment of PSO and BAS algorithm in 2D plane. The next experiment is Artificial Potential Field (APF) and BAS algorithm in 2D plane path planning. These two sets of experiments are mainly to verify the effectiveness of the algorithm in planar path planning.

In the 2D path planning experiment of BAS algorithm and PSO algorithm, this paper sets the blue circular area as the obstacle area, given the starting point and the target point. Because the test environment is simple, this paper uses two path points for path search, using the *spline*() function in Matlab to connect the path point with the start point and the target point. The experimental figure is shown in Fig.10.



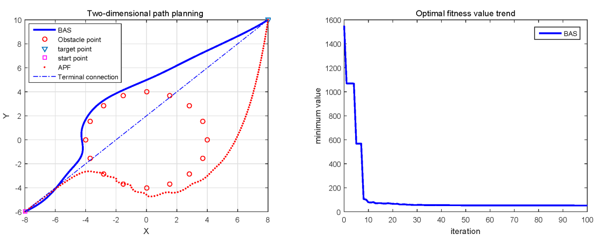
(a)2D path planning. **(b)**Optimal fitness value trend.

FIGURE 10. **BAS and PSO in 2D path planning and the optimal fitness value trend.**

From the Fig.10, we can see that both PSO and BAS algorithms use a strategy of tightening from the periphery to the middle, but in most cases of testing, the PSO algorithm can only search for the path on the left side of the graph due to inertia and group optimal constraints. The BAS algorithm can search on both the left and the right through the initial large search step, and can skip a certain range of obstacles in the process of tightening to the middle, so it can search for the optimal path on the right side multiple times. Because the direction $\overrightarrow{\boldsymbol{b}}$ of the BAS algorithm is random, it is not so efficient in the middle tightening process, so in this experiment we design the BAS algorithm associates the random direction with the PSO velocity formula. Application model is as follows,

$$\overrightarrow{\boldsymbol{b}} = m\overrightarrow{\boldsymbol{b}} + c_1(\boldsymbol{x}_{bst} - \boldsymbol{x})rand(\Delta \boldsymbol{X}) \tag{19}$$

where m, c_1 respectively represent constants, set to 1 and 1.5. $\Delta\boldsymbol{X}$ denotes the range of $\boldsymbol{x}$, $\boldsymbol{x}_{bst}$ denote the current optimal path.



(a) 2D path planning. **(b)** Optimal fitness value trend.

FIGURE 11. BAS and APF in 2D path planning and the optimal fitness value trend.

The path planning experiment of the BAS algorithm and the APF method is shown in Fig.11. In Fig.11a, it can be clearly seen that the artificial potential field method searches for a longer path, and because of the unbalanced between gravity and repulsive force, part of the path intersect with the obstacles, which is more dangerous in practical applications. The maximum number of iterations of the BAS is 100, the step size is 4, and the number of iterations of the APF is 89. Through experiment and code analysis, we found that the search direction of APF is very fixed, and it may be better if the search direction of each step is set to random. Since the path planning of APF is affected by the gravitation and repulsive force of each step, it cannot be compared with the fitness function of BAS. Therefore, only the optimal fitness value trend of BAS algorithm is shown Fig.11b.

Two sets of experiments show that compared with the PSO algorithm, the BAS algorithm can explore a wider range and is less affected by the current optimal solution, so it has better global optimization capabilities. Compared with the APF algorithm, the BAS algorithm is more flexible, it can effectively avoid collisions with obstacles, and can provide a variety of solutions.

V. CONCLUSION

This paper proposed an improved BAS algorithm to solve the problems of poor real-time and low accuracy of the heuristic optimization algorithm in 3D path planning. The improved BAS algorithm mainly relies on three non-trivial mechanisms, including local fast search, aco initial path generation, and searching information orientation. The LFS-BAS algorithm with LFS mechanism can effectively avoid the situation of missing the global optimal solution at the early stage of search and falling into the local optimal solution at the later stage, which can increase the diversity and the possibility of finding a better solution. The ACO-BAS algorithm initialized by ACO has an efficient search ability, can quickly obtain a better solution, and can be applied to scenarios with high requirements for real-time performance. The SIO-BAS algorithm with searching information orientation balances the search speed, search accuracy, and search stability. It can give the best solution as much as possible in the case of meeting multiple requirements. Through comparison experiments with traditional ACO algorithm, the search results of BAS-related algorithms are better than ACO algorithm, which verifies the effectiveness of the proposed algorithm in 3D path planning problems. In addition, we will continue to study the path planning problems of heuristic optimization algorithms in dynamic and complex three-dimensional space.

REFERENCES

[1] J. Barraquand and J.-C. Latombe, "Robot motion planning: A distributed representation approach," *Int. J. Robot. Res.*, vol. 10, no. 6, pp. 628–649, Dec. 1991.

[2] L. Jin, S. Li, H. M. La, X. Zhang, and B. Hu, "Dynamic task allocation in multi-robot coordination for moving target tracking: A distributed approach," *Automatica*, vol. 100, pp. 75–81, Feb. 2019.

[3] X. Jiang and S. Li, "Plume front tracking in unknown environments by estimation and control," *IEEE Trans. Ind. Informat.*, vol. 15, no. 2, pp. 911–921, Feb. 2019.

[4] L. Xiao, B. Liao, S. Li, Z. Zhang, L. Ding, and L. Jin, "Design and analysis of FTZNN applied to the real-time solution of a nonstationary Lyapunov equation and tracking control of a wheeled mobile manipulator," *IEEE Trans. Ind. Informat.*, vol. 14, no. 1, pp. 98–105, Jan. 2018.

[5] L. Zhang, Q. Zhang, B. Du, X. Huang, Y. Tang, and D. Tao, "Simultaneous spectral-spatial feature selection and extraction for hyperspectral images," 2019, *arXiv:1904.03982*. [Online]. Available: https://arxiv.org/abs/1904.03982

[6] J. Shao, B. Du, C. Wu, and L. Zhang, "Can we track targets from space? A hybrid Kernel correlation filter tracker for satellite video," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 11, pp. 8719–8731, Nov. 2019.

[7] A. Bundy and W. Lincoln, "Breadth-first search," in *Catalogue of Artificial Intelligence Tools*. Berlin, Germany: Springer, 1984, p. 13.

[8] R. Tarjan, "Depth-first search and linear graph algorithms," *SIAM J. Comput.*, vol. 1, no. 2, pp. 146–160, Jun. 1972.

[9] H. I. Kang, B. Lee, and K. Kim, "Path planning algorithm using the particle swarm optimization and the improved Dijkstra algorithm," in *Proc. IEEE Pacific-Asia Workshop Comput. Intell. Ind. Appl.*, Dec. 2008.

[10] Y. Zhang and S. Li, "PSA: A novel optimization algorithm based on survival rules of porcellio scaber," 2017, *arXiv:1709.09840*. [Online]. Available: https://arxiv.org/abs/1709.09840

[11] X. Jiang and S. Li, "BAS: Beetle antennae search algorithm for optimization problems," 2017, *arXiv:1710.10724*. [Online]. Available: https://arxiv.org/abs/1710.10724

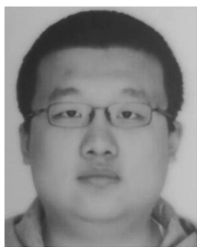
[12] A. T. Khan, S. Li, P. S. Stanimirovic, and Y. Zhang, "Model-free optimization using eagle perching optimizer," 2018, *arXiv:1807.02754*. [Online]. Available: https://arxiv.org/abs/1807.02754

[13] Q. Wu, Z. Ma, G. Xu, S. Li, and D. Chen, "A novel neural network classifier using beetle antennae search algorithm for pattern classification," *IEEE Access*, vol. 7, pp. 64686–64696, 2019.
[14] Q. Wu, X. Shen, Y. Jin, Z. Chen, S. Li, A. H. Khan, and D. Chen, "Intelligent beetle antennae search for UAV sensing and avoidance of obstacles," *Sensors*, vol. 19, no. 8, p. 1758, Apr. 2019.
[15] W. L. David and D. M. Rocke, "Heuristic search algorithms for the minimum volume ellipsoid," *J. Comput. Graph. Statist.*, vol. 2, no. 1, pp. 69–95, 1993.
[16] J. Grefenstette, "Optimization of control parameters for genetic algorithms," *IEEE Trans. Syst., Man, Cybern.*, vol. 16, no. 1, pp. 122–128, Jan. 1986.
[17] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of Machine Learning*. Boston, MA, USA: Springer, 2011. 760-766.
[18] M. Dorigo and G. Di Caro, "Ant colony optimization: A new meta-heuristic," in *Proc. Congr. Evol. Comput.*, Jul. 1999.
[19] M. Neshat, G. Sepidnam, M. Sargolzaei, and A. N. Toosi, "Artificial fish swarm algorithm: A survey of the state-of-the-art, hybridization, combinatorial and indicative applications," *Artif. Intell. Rev.*, vol. 42, no. 4, pp. 965–997, Dec. 2014.
[20] W.-T. Pan, "A new fruit fly optimization algorithm: Taking the financial distress model as an example," *Knowl.-Based Syst.*, vol. 26, pp. 69–74, Feb. 2012.
[21] M. Li, "Three-dimensional path planning of robots in virtual situations based on an improved fruit fly optimization algorithm," *Adv. Mech. Eng.*, vol. 6, Jan. 2014, Art. no. 314797.
[22] T. Wang, Y. Long, and Q. Liu, "Beetle swarm optimization algorithm: Theory and application," 2018, *arXiv:1808.00206*. [Online]. Available: https://arxiv.org/abs/1808.00206
[23] A. Akhtar, "Evolution of ant colony optimization algorithm–a brief literature review," 2019, *arXiv:1908.08007*. [Online]. Available: https://arxiv.org/abs/1908.08007
[24] G. M. Jaradat, A. Al-Badareen, M. Ayob, M. Al-Smadi, I. Al-Marashdeh, M. Ash-Shuqran, and E. Al-Odat, "Hybrid elitist-ant system for nurse-rostering problem," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 31, no. 3, pp. 378–384, Jul. 2019.
[25] J. Chen, X.-M. You, S. Liu, and J. Li, "Entropy-based dynamic heterogeneous ant colony optimization," *IEEE Access*, vol. 7, pp. 56317–56328, 2019.
[26] J. J. Hopfield, "Artificial neural networks," *IEEE Circuits Devices Mag.*, vol. 4, no. 5, pp. 3–10, Sep. 1988.
[27] J. Yao, C. Lin, X. Xie, A. J. Wang, and C.-C. Hung, "Path planning for virtual human motion using improved A∗ star algorithm," in *Proc. 7th Int. Conf. Inf. Technol., New Gener.*, Apr. 2010.
[28] S. S. Ge and Y. J. Cui, "Dynamic motion planning for mobile robots using potential field method," *Auton. Robots*, vol. 13, no. 3, pp. 207–222, 2002.
[29] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
[30] Q. Wu, H. Lin, Y. Jin, Z. Chen, S. Li, and D. Chen, "A new fallback beetle antennae search algorithm for path planning of mobile robots with collision-free capability," *Soft Comput.*, to be published.
[31] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robot. Auto. Syst.*, vol. 61, no. 12, pp. 1258–1276, Dec. 2013.
[32] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Dept. Comput. Sci., Iowa State Univ., Ames, IA, USA, Tech. Rep., 1998, pp. 1–11.
[33] E. Gonzalez, O. Alvarez, Y. Diaz, C. Parra, and C. Bustacara, "BSA: A complete coverage algorithm," in *Proc. IEEE Int. Conf. Robot. Automat.*, Apr. 2005.
[34] H. Choset, "Coverage of known spaces: The boustrophedon cellular decomposition," *Auton. Robots*, vol. 9, no. 3, pp. 247–253, 2000.
[35] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *Proc. 2014 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2014.
[36] B. K. Choi, D. H. Kim, and R. B. Jerard, "C-space approach to tool-path generation for die and mould machining," *Comput.-Aided Des.*, vol. 29, no. 9, pp. 657–669, Sep. 1997.
[37] X. Jiang and S. Li, "Beetle antennae search without parameter tuning (BAS-WPT) for multi-objective optimization," 2017, *arXiv:1711.02395*. [Online]. Available: https://arxiv.org/abs/1711.02395
[38] Z. Xie, L. Jin, X. Du, X. Xiao, H. Li, and S. Li, "On generalized RMP scheme for redundant robot manipulators aided with dynamic neural networks and nonconvex bound constraints," *IEEE Trans. Ind. Informat.*, vol. 15, no. 9, pp. 5172–5181, Sep. 2019.

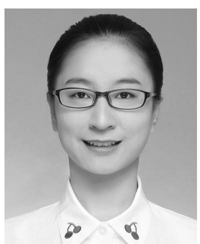
XIANGYUAN JIANG received the B.E. degree in biomedical engineering and the Ph.D. degree in control science and control engineering from Shandong University, China, in 2006 and 2012, respectively.

He is currently an Associate Professor with the Institute of Marine Science and Technology, Shandong University. He also worked with the College of Control Science and Engineering, China University of Petroleum, Qingdao, China. His current research interests include the adaptive control of source seeking, and distributed estimation and control.

ZONGYUAN LIN received the B.S. degree in automation from Qingdao University, Qingdao, China, in 2017. He is currently pursuing the M.S. degree in control science and engineering with the China University of Petroleum, Qingdao.



TIANHAO HE (Member, IEEE) is currently pursuing the B.S. degree in cognitive science with a specialization in machine learning and neural computation with the University of California at San Diego, San Diego, CA, USA.



XIAOJING MA received the B.S. and M.S. degrees from the School of Control Science and Engineering, Shandong University, Shandong, China, where she is currently pursuing the Ph.D. degree.

SILE MA received the B.E. degree in automation engineering from Beijing Technology and Business University, China, in 1985, and the Ph.D. degree in control science and control engineering from Shandong University, in 2014.

He was an Associate Professor with Department of Automation, Shandong Industry University, China. He is currently a Professor with the Institute of Marine Science and Technology, Shandong University, China. His current research interests include industrial process control, and distributed control for industrial manipulators.

SHUAI LI (Senior Member, IEEE) received the B.E. degree in precision mechanical engineering from the Hefei University of Technology, Hefei, China, in 2005, the M.E. degree in automatic control engineering from the University of Science and Technology of China, Hefei, in 2008, and the Ph.D. degree in electrical and computer engineering from the Stevens Institute of Technology, Hoboken, NJ, USA, in 2014. He is currently an Associate Professor (reader) with Swansea University, Wales, U.K., leading the Robotic Lab and conducting research on robot manipulation and impedance control, multirobot coordination, distributed control, intelligent optimization and control, and legged robots. He is the General Co-Chair of the 2018 International Conference on Advanced Robotics and Intelligent Control. He is the founding Editor-in-Chief of the *International Journal of Robotics and Control*.