

Research Article

A hybrid artificial bee colony algorithm with genetic augmented exploration mechanism toward safe and smooth path planning for mobile robot

Fan Ye^a, Peng Duan^{a,b,*}, Leilei Meng^a, Lingyan Xue^a

^a School of Computer Science, Liaocheng University, Liaocheng 252059, China

^b Shandong Provincial Key Laboratory of Ubiquitous Intelligent Computing, Jinan 250022, China

ARTICLE INFO

Article history:

Received 27 September 2024

Revised 20 November 2024

Accepted 4 December 2024

Available online 26 December 2024

Keywords:

Mobile robot

Path planning

Hybrid artificial bee colony with genetic

augmented exploration mechanism

Objective-guided optimization strategy

Dual exploration restart strategy

ABSTRACT

Path planning is important for mobile robot to ensure safe and efficient navigation. This paper proposes a hybrid artificial bee colony with genetic augmented exploration mechanism (HABC-GA) that enables mobile robot to achieve safe and smooth path planning. Considering the characteristics of path planning problem, a mathematical model is constructed to balance three objectives: path length, path safety, and path smoothness. In the employed bee phase, a genetic augmented exploration mechanism is designed, which encompasses redesigned path crossover, adaptive obstacle-aware mutation, and dynamic elite selection operators. In the onlooker bee phase, an objective-guided optimization strategy is investigated to improve local search ability. In the scout bee phase, a dual exploration restart strategy is developed to increase the activity of individuals in the population, in which stagnant individuals in the evolution are replaced by more promising ones. Finally, the proposed HABC-GA is compared with five efficient algorithms in 24 instances of six representative environments. Simulation results demonstrate the effectiveness and high performance of HABC-GA in obtaining safe and smooth paths.

© 2024 The Author(s). Published by Elsevier B.V. on behalf of Shandong University. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Mobile robots are autonomous devices capable of performing particular tasks in intricate environments without human intervention. Currently, mobile robots are extensively utilized across multiple sectors, including manufacturing [1], military [2], agriculture [3], disaster relief [4], navigation [5], and emergency response [6]. Indeed, path planning is a fundamental component and critical technology in robotic systems [7]. Specifically, it involves determining a collision-free route from the start point to the target point within a given workspace. Efficient path planning methods can reduce capital investment while minimizing robot wear and tear. Although path planning has made steady progress in the past few decades, it remains an open problem in many applications.

Nowadays, numerous algorithms are developed to determine the optimal path for mobile robot, such as cell decomposition (CD) [8], artificial potential field algorithm (APF) [9,10], and roadmap method (RM) [11]. Specifically, the CD partitions the robot's configuration space into areas designated as free and those containing obstacles. It utilizes a connectivity graph to navigate through these cells, aiming to establish a collision-free path

from the starting point to the destination. The method usually tends to focus on whether the unit is occupied by obstacles, without strict constraints on the safe distance from the obstacle, so the generated path may be very close to obstacle edges. The APF algorithm generates a virtual force field in the configuration space of robot where the target point creates an attractive force and obstacles generate repulsive forces to prevent collisions. The movement of robot is directed by the combined influence of these forces. However, during path planning, the algorithm can cause the robot to become trapped near obstacles, preventing further movement toward the target. As a result, the path generated may be unsafe or incomplete, leading to task failure. The RM constructs a graph representing the free configuration space of robot by identifying key points and connecting them to collision-free paths. Once a roadmap is established, the robot navigates from the start location to the destination by traversing the chart to find the shortest path. Although RM effectively captures the connectivity of the environment, it often prioritizes path feasibility over path quality. This approach can lead to suboptimal path smoothness and safety, as it may not adequately consider proximity to obstacles or minimize sharp turns between points. Overall, although these classic methods have been successfully applied in path planning to generate the shortest possible path, they generalize imperfectly without consistent performance and computational efficiency in other complex environments.

* Corresponding author.

E-mail address: duanpeng@lcu.edu.cn (P. Duan).

Recently, researchers increasingly turned to metaheuristic algorithms inspired by biological and natural processes, as they have greater flexibility and adaptability in complex environments. In contrast to the classical algorithms, metaheuristic algorithms including particle swarm optimization algorithm (PSO) [12], ant colony optimization algorithm (ACO) [13], genetic algorithm (GA) [14–17], and firefly algorithm (FA) [18] are not limited by strict assumptions about environmental structure or precise robot motion capability. These algorithms balance path feasibility, optimality, and efficiency by searching for near-optimal solutions rather than exact solutions. Among the various metaheuristic algorithms, the artificial bee colony algorithm (ABC) [19] is extensively utilized across numerous fields, including path planning, due to its good exploration ability, few parameters, and simple structure. However, existing research often lacks deep search mechanism, which leads to incomplete exploration of the solution space. In this study, we proposed a HABC-GA to address safe and smooth path planning for mobile robot. Considering the excellent global search capability of GA, we redesign the evolutionary operators of GA based on the specific characteristics of the path planning problem. Specifically, we developed a path crossover operator to enhance solution diversity, an adaptive obstacle-aware mutation operator for local optimization, and a dynamic elite selection operator to retain high-quality solutions for further exploration. These improvements significantly enhance the algorithm's ability to explore the solution space effectively. The contributions of this study are as follows:

- (1) A mathematical model that balances path length, path safety, and path smoothness is formulated.
- (2) A genetic augmented exploration mechanism is designed to improve algorithm's global search ability.
- (3) An objective-guided optimization strategy is investigated to enhance algorithm's local search ability.
- (4) A dual exploration restart strategy is developed to increase the activity of individuals in the population.

The structure of this paper is outlined as follows. Section 2 covers the pertinent research work, followed by Section 3, which explains the path planning problem addressed in this study. Section 4 presents the proposed HABC-GA, while Section 5 describes the simulation results. Finally, Section 6 offers conclusions and future works.

2. Related works

In recent years, many researchers have employed various heuristic algorithms to solve path planning problems. A commonly used one is the PSO algorithm. For example, Ajeil et al. [20] designed a PSO-MFB that combines local search and obstacle detection modules for planning collision-free paths that meet the shortest distance and path smoothness criteria. Song et al. [12] developed an improved PSO algorithm to address the problem of local traps and premature convergence, combined with continuous high-order Bezier curves to plan the smooth path of mobile robots. Yuan et al. [21] presented an improved PSO that incorporates differential evolution, which improves convergence speed and achieves safe path planning by introducing adaptive adjustment of weights and acceleration coefficients. Similarly, Zhang et al. [22] introduced an adaptive PSO that dynamically adjusts inertia weights and acceleration coefficients based on the group's state for smooth path planning in environments with obstacles. Besides, Lin et al. [23] proposed a hybrid PSO-SA algorithm for AGV path planning, achieving smoother, collision-free paths with faster convergence and reduced runtime. These PSO variants have demonstrated convincing performance in solving path planning problem by optimizing algorithm parameters or enhancing global

search to avoid premature convergence. However, this makes algorithm performance sensitive to parameters. In different scenarios, it is necessary to constantly adjust parameters, which increases the complexity of the algorithm.

Currently, ACO algorithm is widely used to solve multi-objective path planning problems. For instance, Su et al. [24] developed an improved ACO to address path redundancy and local optima in mobile robot path planning, optimizing both path length and path smoothness. Miao et al. [25] enhanced ACO by introducing adaptive mechanisms and multi-objective optimization to achieve global optimal paths with improved safety performance and stability for indoor robot path planning. Additionally, Hou et al. [26] suggested an enhanced ACO with a communication mechanism, which can integrate historical path information to achieve the shortest path planning. Wu et al. [27] proposed an improved adaptive ACO, which significantly improves the efficiency of path planning and shortens path length by introducing direction guidance, improving heuristic functions, state transition rules, and pheromone distribution. Moreover, Cui et al. [28] proposed a multi-strategy adaptive ACO to alleviate the problems of insufficient convergence and low efficiency of the ant colony algorithm through four design improvements, to achieve shortest path planning. The aforementioned studies reveal that ACO has shown remarkable effectiveness in addressing path planning problem. However, the excessive dependence of the ACO on the existing pheromone concentration on the path can easily lead to local optima. Although researchers have enhanced ACO's exploration ability through various improvements, it still faces challenges in balancing exploration and exploitation capabilities.

In addition to PSO and ACO algorithms, the ABC algorithm has been applied by researchers to solve multi-objective path planning problems due to its simple structure and minimal parameters. Within this framework, the three types of bees, employed, onlooker, and scout, collaborate effectively to strike a balance between exploration and exploitation. For example, Contreras-Cruz et al. [29] integrated ABC with evolutionary programming to achieve shortest path planning for mobile robot. Xu et al. [30] implemented a coevolutionary framework in ABC, creating an advanced version that enhances convergence speed for optimal path planning. Kamil et al. [31] developed an adaptive dimension-constrained ABC for path planning of mobile robots, which reduces unnecessary iterations and computation time through dynamic parameter control and improves path planning efficiency. In addition, Cui et al. [32] introduced an improved ABC that uses reinforcement learning to optimize exploration and search strategies, achieving optimal path planning. Similarly, Cui et al. [33] developed an IMOABC algorithm that leverages advanced multi-objective strategies to enhance solution quality and demonstrates superior performance in solving complex path planning problem for mobile robot. Li et al. [34] presented a node mobility ABC algorithm that improves initial path generation and neighborhood search, providing faster convergence speed and better global search capability for robot path planning. In summary, for the above literatures, we found that researchers have improved convergence speed of algorithms and path planning efficiency by incorporating co-evolutionary frameworks and rules, or using dynamic parameter control. However, the above algorithms lack a deep search mechanism, which is crucial for exploring the solution space as much as possible and further improving the performance of solution.

3. Path planning

3.1. Problem definition of path planning

Let $X \subset R^d$ denote the configuration space of a mobile robot, where d is the dimensionality of the configuration space, where

$d \in \mathbb{N}$, $d \geq 2$. Define $X_{\text{obs}} \subset X$ as the obstacle region and $X_{\text{free}} \subset X \setminus X_{\text{obs}}$ as the corresponding free space. A path planning problem can thus be defined by the tuple $(X, x_{\text{start}}, x_{\text{target}})$, where $x_{\text{start}} \in X_{\text{free}}$ is the start state and $x_{\text{target}} \in X_{\text{free}}$ represents the target state, respectively. Assume the robot is a point agent whose path is represented by a continuous function with bounded variation, $\sigma(\tau): [0, 1] \rightarrow X$, where $\sigma(\tau)$ indicates an intermediate state along the path.

Problem1 (Feasible Path Planning). Given a path planning problem $(X, x_{\text{start}}, x_{\text{target}})$, a feasible path exists if the robot can move from the start state x_{start} to the target state x_{target} such that

$$\sigma(\tau): [0, 1] \rightarrow X_{\text{free}}, \sigma(0) = x_{\text{start}}, \sigma(1) = x_{\text{target}}, \forall \tau \in [0, 1], \sigma(\tau) \in X_{\text{free}} \quad (1)$$

If no such path exists, report failure.

Problem2 (Optimal Path Planning). Let Σ denote the set of all feasible paths, and $f(\cdot)$ represent an objective function to be minimized over the path. The optimal path with respect to this objective function is formulated as

$$\sigma^* = \underset{\sigma \in \Sigma}{\operatorname{argmin}} \{f(\sigma) \mid \sigma(0) = x_{\text{start}}, \sigma(1) = x_{\text{target}}, \forall \tau \in [0, 1], \sigma(\tau) \in X_{\text{free}}\} \quad (2)$$

3.2. Path metrics

3.2.1. Path length

Path length directly affects the navigation time required for a mobile robot. Let the path ζ be defined as $\zeta = [S, p_1, p_2, \dots, p_i, \dots, p_{n-1}, T]$, where each p_i corresponds to a coordinate point (x_i, y_i) in the two-dimensional space, and $n+1$ denotes the total number of points along the path. Path length is calculated by summing the lengths of each segment along the trajectory, which is shown as follows:

$$\text{Length}(\zeta) = \sum_{i=0}^{n-1} \text{dis}(p_i, p_{i+1}) \quad (3)$$

$$\text{dis}(p_i, p_{i+1}) = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \quad (4)$$

where $\text{dis}(p_i, p_{i+1})$ is the Euclidean distance between p_i and p_{i+1} .

3.2.2. Path safety

Path safety measures the safe distance between a path and obstacles. A safe path ensures mobile robot to move with a high speed. Path safety is determined by measuring the minimum distance between the path and obstacles. In this study, to minimize this objective, the calculation formula of path safety is presented as follows:

$$\text{Safety}(\zeta) = \frac{1}{\text{mindis}} \quad (5)$$

$$\text{mindis} = \min_{1 \leq i \leq n-1} \min_{1 \leq j \leq m} \{\text{dis}(\overline{P_i P_{i+1}}, \text{Obs}_j)\} \quad (6)$$

where mindis describes the closest distance between the path ζ and obstacles.

3.2.3. Path smoothness

Path smoothness is closely associated with the energy consumption of mobile robot. In this study, the smoothness of a feasible path is quantified by computing the angle α_i formed between two adjacent segments. Hence, the path smoothness could be determined by the maximum complementary angle of all angles along the path, which is calculated as follows:

$$\text{Smoothness}(\zeta) = \max_{i=1, \dots, n-1} \{\pi - \alpha_i\} \quad (7)$$

$$\alpha_i = \sec \frac{(x_i - x_{i-1})(x_{i+1} - x_i) + (y_i - y_{i-1})(y_{i+1} - y_i)}{\text{dis}(p_{i-1}, p_i) \times \text{dis}(p_i, p_{i+1})} \quad (8)$$

where α_i refers to the value of the i th angle that ranges from 0 to π .

3.3. Objective function

To obtain safe and smooth paths, in this study, path metrics discussed in Section 3.2 are given corresponding weights. The optimization objective function of path ζ is formulated as follows:

$$f(\zeta) = w_1 \times \text{Length}(\zeta) + w_2 \times \text{Safety}(\zeta) + w_3 \times \text{Smoothness}(\zeta) \quad (9)$$

where w_1 , w_2 , and w_3 are the weight coefficients of path length, path safety, and path smoothness, respectively.

4. The proposed HABA-GA

This section provides the proposed HABC-GA, including its framework, genetic augmented exploration mechanism, objective-guided optimization strategy, and dual-exploration restart strategy.

4.1. The framework of HABC-GA

The HABC-GA consists of four phases, i.e., the initialization phase, the employed bee phase, the onlooker bee phase, and the scout bee phase. Algorithm 1 presents the framework of HABC-GA. In the initialization phase, RRT* [35] is used to generate the initialization population. RRT* is chosen not only for its widespread application in solving path planning problems but also for its extensive exploration of the configuration space, which facilitates the generation of diverse solutions. However, this algorithm also has two notable drawbacks: its slow convergence speed and its tendency to produce tortuous paths. In this study, these tortuous paths contribute to enriching the diversity of the initial population. It should be noted that, to enhance initialization efficiency, two termination conditions are employed during each execution of RRT*: (1) successfully finding a new solution, or (2) exceeding a single execution time of 0.01 s. Through iterative application of this process, RRT* generates a predetermined number of initial solutions. Subsequently, the objective value for each individual is calculated. In the employed bee phase, a genetic augmented exploration mechanism is applied to population, resulting in a new population, for which the objective values are recalculated. In the onlooker bee phase, an objective-guided optimization strategy is adopted to enhance the performance of each individual in population. Then, the objective values are recalculated. In the scout bee phase, a dual exploration restart strategy is applied on population to obtain a more active population. Finally, the optimal solution is obtained. Fig. 1 presents a graphical explanation of the proposed HABC-GA.

4.2. Genetic augmented exploration mechanism

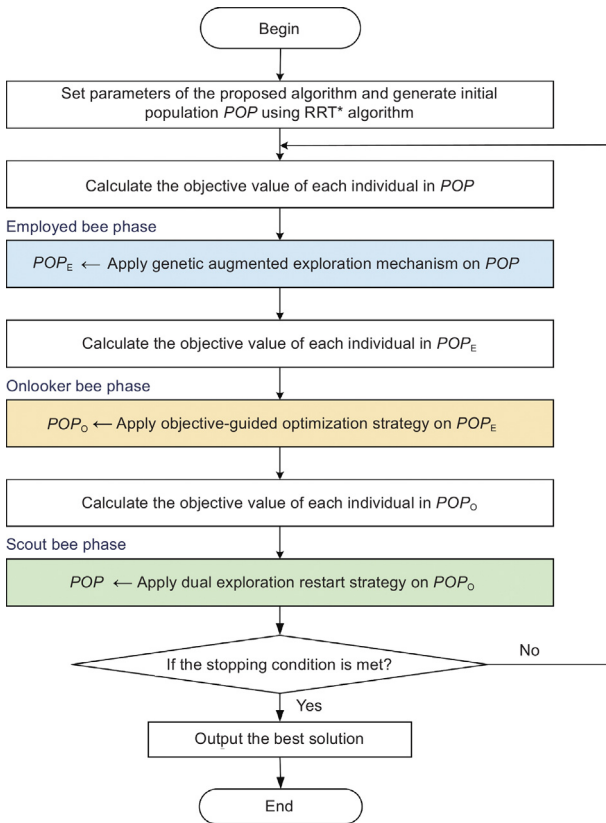
The genetic augmented exploration mechanism is designed to enhance the global exploration ability of the proposed algorithm. Inspired by genetic evolution, this mechanism consists of three operators, i.e., path crossover operator, adaptive obstacle-aware mutation operator, and dynamic elite selection operator. This mechanism begins by conducting path crossover operator on each individual to form a temporary crossover population. Afterward, an adaptive obstacle-aware mutation operator is applied to each individual of crossover population to form temporary mutated

Algorithm 1: Framework of the proposed HABC-GA**Output:** *best solution*

- ```

initialization phase
1. $POP \leftarrow$ generate initial population using RRT*;
2. while the stopping condition is not satisfied do
employed bee phase
3. $ObjectiveValue_{POP} \leftarrow$ calculate the objective value of each individual in POP ;
4. $POP_E \leftarrow$ apply genetic augmented exploration mechanism on POP ;
5. $ObjectiveValue_{POPE} \leftarrow$ calculate the objective value of each individual in POP_E ;
onlooker bee phase
6. $POP_O \leftarrow$ apply objective-guided optimization strategy on POP_E ;
7. $ObjectiveValue_{POPO} \leftarrow$ calculate the objective value of each individual in POP_O ;
scout bee phase
8. $POP \leftarrow$ apply dual exploration restart strategy on POP_O ;
9. endwhile

```

**Fig. 1.** The flowchart of the proposed HABC-GA.

population. Finally, a dynamic elite selection operator is applied to the combined mutated, crossover, and current populations to select the most promising individuals, which form the new population for next phase. Algorithm 2 provides pseudocode for the genetic augmented exploration mechanism.

#### 4.2.1. Path crossover operator

The purpose of the path crossover operator is to create offspring that obtain advantageous features from the parent generation. In this operator, a new feasible path is generated by exchanging path segments between two different paths. One is

the current solution, and the other is selected from the top 50% of solutions ranked in ascending order of their objective values. In the worst case, the time complexity of the path crossover operator is  $O(n)$ , where  $n$  is the number of the path segments. Algorithm 3 presents the pseudocode for the operator.

#### 4.2.2. Adaptive obstacle-aware mutation operator

The adaptive obstacle-aware mutation operator aims to promote the local search ability of proposed algorithm. To minimize the computational cost associated with collision detection, this study employs the single-point mutation operator. In traditional single-point mutation operator, mutation point is replaced by randomly generated point. On the other hand, traditional single-point mutation operators may make the path infeasible. The adaptive obstacle-aware mutation operator is achieved by threshold control based on the minimum distance between path and obstacles. A new mutation point is added into path only if both the new point and the new path segments created by new point and two adjacent points are feasible. The time complexity of the adaptive obstacle-aware mutation operator is  $O(1)$ . The pseudocode for this operator is presented in Algorithm 4.

#### 4.2.3. Dynamic elite selection operator

The dynamic elite selection operator aims to select high-performance individuals. In this operator, two variable-sized populations are used to store solutions with better objective values and diversity, i.e., elite population  $POP_e$  and differential population  $POP_{1-\varepsilon}$ . In addition, a dynamic ratio  $\varepsilon$  based on the runtime is employed to adjust the size of elite population and differential population. Individual differences are calculated using Eq. (10).

$$DE(M, \zeta_i) = |M(\zeta_i) - \text{median}(M(POP))| \quad (10)$$

where  $M(\bullet)$  denotes metric values of a special path,  $\text{median}(M(POP))$  denotes the median of the metric values of all paths. In the early stage, the operator tends to select individuals with significant differences to maintain the diversity of solutions. As the algorithm runs, it tends to select individuals with better objective values to accelerate convergence toward optimal solution. The time complexity of the dynamic elite selection operator is  $O(1)$ . Algorithm 5 provides the pseudocode for this operator.

#### 4.3. Objective-guided optimization strategy

The objective-guided optimization strategy aims to enhance the algorithm's local search ability. The strategy begins by

**Algorithm 2:** Genetic augmented exploration mechanism**Input:** current population:  $POP$ , objective value of individuals in  $POP$ :  $ObjectiveValue_{POP}$ **Output:** new population:  $POP_{new}$ 

1. **for** each solution  $\zeta_i$  in  $POP$  **do**
2.      $\zeta_r \leftarrow$  select a solution from the top 50% of  $POP$  ranked by objective value;
3.      $\zeta_{i\_new} \leftarrow PathCrossoverOperator(\zeta_i, \zeta_r)$ ;
4.     store  $\zeta_{i\_new}$  into  $POP_{cross}$ ;
5. **endfor**
6. **for** each solution  $\zeta_j$  in  $POP_{cross}$  **do**
7.      $\zeta_{j\_new} \leftarrow AdaptiveObstacleAwareMutationOperator(\zeta_j)$ ;
8.     store  $\zeta_{j\_new}$  into  $POP_{mutated}$ ;
9. **endfor**
10.  $POP_{new} \leftarrow DynamicEliteSelectionOperator(POP_{mutated}, POP_{cross}, POP)$ ;

**Algorithm 3:** Path crossover operator**Input:** current population:  $POP$ , two current solutions:  $\zeta_i = [S, p^1, \dots, p^k, \dots, T]$  and  $\zeta_r = [S, p^1, \dots, p^l, \dots, T]$ **Output:** new solution:  $\zeta_{i\_new}$ 

1.  $[p^i_{k-1}, p^i_{k+2}] \leftarrow$  select two intermediate nodes on  $\zeta_i$ ;
2.  $[p^r_{l-1}, p^r_{l+2}] \leftarrow$  select two intermediate nodes on  $\zeta_r$ ;
3.  $Dis \leftarrow []$ ; # initialize list to store distances between selected nodes;
4. **for** each pair nodes  $(p^i_{k_v}, p^r_{l_w})$  in  $[(p^i_{k-1}, p^r_{l-1}), (p^i_{k-1}, p^r_{l+1}), (p^i_{k+2}, p^r_{l-1}), (p^i_{k+2}, p^r_{l+2})]$  **do**
5.     **if**  $CollisionFree(p^i_{k_v}, p^r_{l_w})$  **then**
6.          $distance \leftarrow CalculateDistance(p^i_{k_v}, p^r_{l_w})$ ; # Compute Euclidean distance between  $p^i_{k_v}$  and  $p^r_{l_w}$ ;
7.         append  $(distance, p^i_{k_v}, p^r_{l_w})$  to  $Dis$ ;
8.     **endif**
9. **endfor**
10. **if**  $Dis$  is not empty **then**
11.      $[min\_distance, p^i_{k\_v\_opt}, p^r_{l\_w\_opt}] \leftarrow FindMinDistance(Dis)$ ; # Select feasible pair points with shortest distance;
12.      $\zeta_{i\_new}^1 \leftarrow [S, p^1, \dots, p^i_{k\_v\_opt-1}, p^i_{k\_v\_opt}, p^r_{l\_w\_opt}, p^r_{l\_w\_opt+1}, \dots, T]$ ;
13.      $\zeta_{i\_new}^2 \leftarrow [S, p^1, \dots, p^r_{l\_w\_opt-1}, p^r_{l\_w\_opt}, p^i_{k\_v\_opt}, p^i_{k\_v\_opt+1}, \dots, T]$ ;
14.      $\zeta_{i\_new} \leftarrow$  select a path with the minimum objective value between  $\zeta_{i\_new}^1$  and  $\zeta_{i\_new}^2$ ;
15. **else**
16.      $\zeta_{i\_new} \leftarrow \zeta_i$ ;
17. **endif**

**Algorithm 4:** Adaptive obstacle-aware mutation operator**Input:** current solution:  $\zeta_i = [S, p^1, \dots, p^k, \dots, T]$ **Output:** new solution:  $\zeta_{i\_new}$ 

1.  $p^j_k \leftarrow$  select an intermediate point on  $\zeta_i$  randomly;
2.  $dis_{i\_obs} \leftarrow$  calculate the distance between  $p^j_k$  and the nearest obstacle;
3.  $p^j_{new} \leftarrow$  generate a new point with a distance of  $dis_{i\_obs} * 0.5$  around point  $p^j_k$ ;
4. **if**  $CollisionFree(p^j_{k-1}, p^j_{new}) \ \&\& \ CollisionFree(p^j_{k+1}, p^j_{new})$  **then**
5.      $\zeta_{i\_new} \leftarrow [S, p^1, \dots, p^j_{k-1}, p^j_{new}, p^j_{k+1}, \dots, T]$ ;
6. **else**
7.      $\zeta_{i\_new} \leftarrow \zeta_i$ ;
8. **endif**

---

Algorithm 5. Dynamic elite selection operator

---

**Input:**  $POP$ ,  $POP_{cross}$ ,  $POP_{mutated}$ , deadline for running:  $t_{max}$ , current run time:  $t$ , population size:  $p_{size}$

**Output:** new population:  $POP_{new}$

1.  $POP_{merge} \leftarrow POP \cup POP_{cross} \cup POP_{mutated}$ ;
  2. calculate the objective values of  $POP_{merge}$  and sort individuals in ascending order based on objective values;
  3.  $\varepsilon = t/t_{max}$ ;
  4.  $num\_elite = \lfloor p_{size} \times \varepsilon \rfloor$ ; #  $\lfloor \cdot \rfloor$  is used to round up a float number.
  5.  $POP_{\varepsilon} \leftarrow$  select the top  $num\_elite$  individuals with objective values ranking from  $POP_{merge}$ ;
  6.  $rand \leftarrow$  generate a random integer number between 1 and 3;
  7. **if**  $rand == 1$  **then**
  8.  $POP_{1-\varepsilon} \leftarrow$  select  $(p_{size} - num\_elite)$  individuals with larger  $DE$  values on length metric from  $[POP_{merge} - POP_{\varepsilon}]$ ;
  8. **elseif**  $rand == 2$  **then**
  9.  $POP_{1-\varepsilon} \leftarrow$  select  $(p_{size} - num\_elite)$  individuals with larger  $DE$  values on safety metric from  $[POP_{merge} - POP_{\varepsilon}]$ ;
  9. **else**
  10.  $POP_{1-\varepsilon} \leftarrow$  select  $(p_{size} - num\_elite)$  individuals with larger  $DE$  values on smoothness metric from  $[POP_{merge} - POP_{\varepsilon}]$ ;
  11. **endif**
  12.  $POP_{new} \leftarrow POP_{\varepsilon} \cup POP_{1-\varepsilon}$ ;
- 

normalizing the metric values of each solution in population, i.e., path length, path safety, and path smoothness values. Normalization is conducted using the following formula:

$$Norm\_M_v^i = \frac{M_v(\zeta_i) - \min(M_v(POP))}{\max(M_v(POP)) - \min(M_v(POP))} \quad (11)$$

where  $Norm\_M_v^i$  represents the normalized value of the  $i$ th solution for a specific path metric,  $M_v(\zeta_i)$  represents the metric value of the  $i$ th solution.  $\min(M_v(POP))$  and  $\max(M_v(POP))$  denote the minimum and maximum metric values of all solutions in  $POP$ , respectively. Afterward, the strategy identifies the metric that most significantly affects the current solution's objective value. Based on the identified metric, this strategy applies the corresponding metric optimization operator, i.e., path length, path safety, or path smoothness as described in [36]. If the optimized solution achieves a better objective value, it is replaced by the optimized one. In the worst case, the time complexity of the objective-guided optimization strategy is  $O(p_{size} \times (n^2 + n + 1))$ , where  $n$  denotes the number of path nodes, and  $p_{size}$  denotes the population size. Algorithm 6 provides the pseudocode for the objective-guided optimization strategy.

#### 4.4. The dual exploration restart strategy

The dual exploration restart strategy aims to increase the activity of individuals in the population. The strategy sorts individuals in ascending order based on their objective values and monitors stagnation value, which represents the number of consecutive evolutionary failures. If an individual's stagnation value exceeds a predefined threshold, this strategy replaces the individual with a new one. Specifically, individuals in the population are divided into two groups: the top 50% of individuals and the remaining 50% of individuals, according to their objective values. Different operations are applied to the stagnant evolution individuals of different categories. Algorithm 7 demonstrates pseudocode for dual exploration restart strategy. Lines 4 and 5 describe the operation conducted on the top 50% of individuals, while line 7 provides the operation conducted on remaining 50%

of individuals.

## 5. Simulation results

To verify the performance of the proposed HABC-GA algorithm as discussed in Section 4, extensive computational experiments were performed with five efficient algorithms proposed in recent years, i.e., SPSFA [37], PSO-MFB [20], PSO-SA [23], IAACO [25], and IMOABC [33]. All algorithms were implemented in MATLAB R2019a and executed on a PC with a 3.60 GHz CPU, 16.0 GB RAM. To ensure a fair comparison, all compared algorithms used the same termination condition with a running time of 50 s. Each algorithm was independently run 30 times, and the results were collected for performance comparison.

The relative percentage increase (RPI) was used to assess the performance of the proposed algorithm, given as follows:

$$RPI = \frac{f_c - f_b}{f_b} \times 100\% \quad (12)$$

where  $f_c$  represents the value obtained through the specified algorithm,  $f_b$  is the optimal value from all algorithms.

### 5.1. Experimental instances

To comprehensively assess the performance of the proposed algorithm, six representative maps are used for testing and shown in Fig. 2. Map A represents an irregular environment characterized by a chaotic distribution of obstacles, while Map B depicts a realistic outdoor street setting. Map C represents a forest environment with maze feature sourced from Nathan Sturtevant's Moving AI Lab [38]. Map D represents an indoor environment. Laser\_Map E and Laser\_Map F are high-resolution laser radar maps. Each map is standardized to a size of  $500 \times 500$  pixels. For each map, coordinate points for prospective paths are selected from four directions, i.e., main diagonal (MD), secondary diagonal (SD), horizontal axis (HA), and vertical axis (VA). The selection of MD, SD, HA, and VA setups was based on their ability to represent

**Algorithm 6:** Objective-guided optimization strategy

**Input:** current population:  $POP$ , objective value of each individual:  $ObjectiveValue_{POP}$ , each individual corresponds to optimization metric values, i.e., path length, path safety, and path smoothness values:  $[M_{length}, M_{safety}, M_{smoothness}]$

**Output:** new population:  $POP_{new}$

1.  $[Norm\_M_{length}, Norm\_M_{safety}, Norm\_M_{smoothness}] \leftarrow$  normalize each metric value using Eq. (8) based on  $[M_{length}, M_{safety}, M_{smoothness}]$ ;
2. **for** each solution  $\zeta_i$  in  $POP$  **do**
3.  $id \leftarrow$  calculate the index of the minimum value in  $[Norm\_M_{length}^i, Norm\_M_{safety}^i, Norm\_M_{smoothness}^i]$ ;
4. **if**  $id == 1$  **then**
5.  $\zeta_{i\_new} \leftarrow PathLengthOperator(\zeta_i)$ ;
6.  $Objective\_value\_i \leftarrow$  calculate the objective value of  $\zeta_{i\_new}$ ;
7. **if**  $Objective\_value\_i < ObjectiveValue_{POP}^i$  **then**
8.  $\zeta_i \leftarrow \zeta_{i\_new}$ ;
9. **end**
10. **elseif**  $index == 2$  **then**
11.  $\zeta_{i\_new} \leftarrow PathSafetyOperator(\zeta_i)$ ;
12. execute lines 6-9;
13. **else**
14.  $\zeta_{i\_new} \leftarrow PathSmoothnessOperator(\zeta_i)$ ;
15. execute lines 6-9;
16. **endif**
17. store  $\zeta_{i\_new}$  into  $POP_{new}$ ;
18. **endfor**

**Algorithm 7:** Dual exploration restart strategy

**Input:** current population:  $POP$ , population objective function values:  $ObjectiveValue_{POP}$ , individual stagnation value in the population:  $Stag\_Value$ , threshold for stagnant evolution:  $stag_{max}$ , population size:  $p_{size}$

**Output:** new population:  $POP_{new}$

1.  $ID \leftarrow$  sort individuals of  $POP$  in ascending order corresponding objective value and assign corresponding numbers to the sorted population;
2. **for** each solution  $\zeta_i$  in  $POP$  **do**
3. **if**  $Stag\_Value(\zeta_i) > stag_{max}$  **then**
4. **if**  $ID_i < p_{size} \times 0.5$  **then** # Top 50% individuals
5.  $\zeta_{i\_new} \leftarrow AdaptiveObstacleAwareMutationOperator(\zeta_i)$ ;
6. **else** # Remaining 50% individuals
7.  $\zeta_{i\_new} \leftarrow$  select the solution with the best objective value among the five solutions generated using the RRT\*;
8. **endif**
9.  $\zeta_i \leftarrow \zeta_{i\_new}$ ;
10. **endif**
11. **endfor**
12.  $POP_{new} \leftarrow POP$ ;

a variety of directional and spatial challenges encountered in real-world path planning scenarios. Table 1 lists the 24 environmental instances, with the first column giving the environmental maps, the second column providing the instances, and the last two columns presenting the start and target points.

## 5.2. Weight coefficients selection

In multi-objective path planning problems, different objectives have different ranges of variation. After preliminary experiments, we found that the metric values of path length, path safety, and

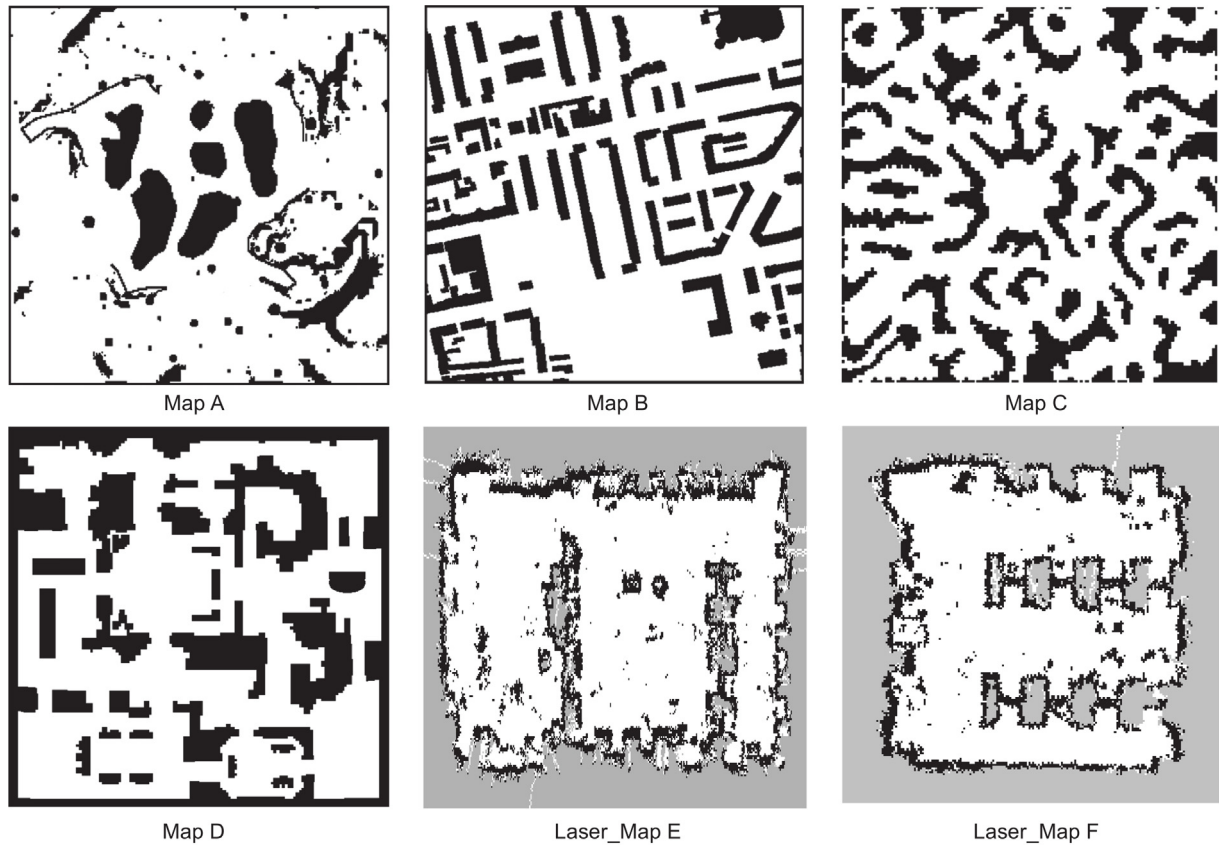


Fig. 2. Six tested environmental maps used to evaluate the compared algorithms.

**Table 1**  
Tested environmental instances.

| Tested environments | Instances     | Start point | Target point |
|---------------------|---------------|-------------|--------------|
| Map A               | Inst_map_A_MD | (30, 30)    | (470, 460)   |
|                     | Inst_map_A_SD | (445, 45)   | (75,480)     |
|                     | Inst_map_A_HA | (235, 25)   | (250, 485)   |
|                     | Inst_map_A_VA | (25, 265)   | (465, 280)   |
| Map B               | Inst_map_B_MD | (15, 15)    | (445, 455)   |
|                     | Inst_map_B_SD | (415, 15)   | (55, 470)    |
|                     | Inst_map_B_HA | (245, 20)   | (255, 485)   |
|                     | Inst_map_B_VA | (15, 235)   | (480, 265)   |
| Map C               | Inst_map_C_MD | (17, 61)    | (443, 479)   |
|                     | Inst_map_C_SD | (477, 67)   | (23, 440)    |
|                     | Inst_map_C_HA | (248, 18)   | (216, 478)   |
|                     | Inst_map_C_VA | (18, 265)   | (451, 238)   |
| Map D               | Inst_map_D_MD | (40, 40)    | (455, 450)   |
|                     | Inst_map_D_SD | (455, 45)   | (48, 458)    |
|                     | Inst_map_D_HA | (228, 24)   | (235, 470)   |
|                     | Inst_map_D_VA | (37, 245)   | (459, 259)   |
| Laser_Map E         | Inst_map_E_MD | (75, 60)    | (400, 440)   |
|                     | Inst_map_E_SD | (370, 80)   | (105, 440)   |
|                     | Inst_map_E_HA | (220, 60)   | (220, 445)   |
|                     | Inst_map_E_VA | (105, 275)  | (380, 275)   |
| Laser_Map F         | Inst_map_F_MD | (90, 110)   | (320, 415)   |
|                     | Inst_map_F_SD | (400, 90)   | (115, 409)   |
|                     | Inst_map_F_HA | (244, 90)   | (230, 415)   |
|                     | Inst_map_F_VA | (100, 285)  | (420, 280)   |

path smoothness are in different numerical orders of magnitude, i.e.,  $10^2$ ,  $10^0$ , and  $10^{-1}$ , respectively. The magnitude difference between these metrics may lead to an imbalance in optimization. To address this problem and prioritize smoother and safer paths, a weighted method is employed to normalize the metrics to a similar order of magnitude. The procedure for determining the weight coefficients is as follows:

Step 1: assign weight coefficients  $w_1$ ,  $w_2$ , and  $w_3$  for path length, path safety, and path smoothness, respectively.

Step 2: execute the following loop.

**for**  $i = 1$  to 3 **do**

    set  $w_i$  to 1, and set the other two coefficients to 0;

    repeat simulation experiment 30 times and output corresponding optimal metric value  $opt\_metric$ ;

**endfor**

**Table 2**  
Weight coefficients setting and standardization coefficient values.

| Metric            | Weight coefficient setting |                          |                           |                                    |
|-------------------|----------------------------|--------------------------|---------------------------|------------------------------------|
|                   | $w_1 = 1, w_2 = w_3 = 0$   | $w_2 = 1, w_3 = w_1 = 0$ | $w_3 = 1, w_1 = w_2 = 0,$ | $w_1 = 0.002, w_2 = 7.65, w_3 = 1$ |
| $otp\_length$     | 689.78                     | 0                        | 0                         | 1.38                               |
| $otp\_safety$     | 0                          | 0.20                     | 0                         | 1.53                               |
| $otp\_smoothness$ | 0                          | 0                        | 1.53                      | 1.53                               |

**Table 3**  
Levels of four parameters.

| Parameter   | Level |      |      |       |
|-------------|-------|------|------|-------|
|             | 1     | 2    | 3    | 4     |
| $pop\_size$ | 10.0  | 20.0 | 50.0 | 100.0 |
| $p_c$       | 0.4   | 0.5  | 0.6  | 0.7   |
| $p_m$       | 0.6   | 0.7  | 0.8  | 0.9   |
| $stag\_max$ | 5.0   | 10.0 | 15.0 | 20.0  |

**Table 4**  
Parameter combinations and their corresponding response values.

| Parameter combinations | Level       |       |       |             | RV    |
|------------------------|-------------|-------|-------|-------------|-------|
|                        | $pop\_size$ | $p_c$ | $p_m$ | $stag\_max$ |       |
| 1                      | 1           | 1     | 1     | 1           | 7.236 |
| 2                      | 1           | 2     | 2     | 2           | 7.159 |
| 3                      | 1           | 3     | 3     | 3           | 7.007 |
| 4                      | 1           | 4     | 4     | 4           | 7.354 |
| 5                      | 2           | 1     | 2     | 3           | 6.971 |
| 6                      | 2           | 2     | 1     | 4           | 6.999 |
| 7                      | 2           | 3     | 4     | 1           | 6.768 |
| 8                      | 2           | 4     | 3     | 2           | 6.820 |
| 9                      | 3           | 1     | 3     | 4           | 6.940 |
| 10                     | 3           | 2     | 4     | 3           | 6.909 |
| 11                     | 3           | 3     | 1     | 2           | 7.069 |
| 12                     | 3           | 4     | 2     | 1           | 6.969 |
| 13                     | 4           | 1     | 4     | 2           | 6.900 |
| 14                     | 4           | 2     | 3     | 1           | 7.057 |
| 15                     | 4           | 3     | 2     | 4           | 6.979 |
| 16                     | 4           | 4     | 1     | 3           | 6.852 |

Step 3: set the metric coefficient of the intermediate order of magnitude to 1, i.e.,  $w_3 = 1$ .

Step 4: calculate the other two metric weight coefficients, i.e.,  $w_1$  and  $w_2$ , using the following formula:

$$w_1 = \frac{otp\_smoothness}{otp\_length}, w_2 = \frac{otp\_smoothness}{otp\_safety} \quad (13)$$

This procedure ensures that each metric is standardized to the same order of magnitude. Table 2 provides the setting of weight coefficients and normalized coefficient values. From the last column of Table 2, it can be seen that the standardized values of the weight coefficients are almost equal.

### 5.3. Main parameters analysis

In this study, four main parameters were calibrated, i.e., population size ( $pop\_size$ ), crossover rate ( $p_c$ ), mutation rate ( $p_m$ ), and threshold for stagnant evolution ( $stag\_max$ ). Table 3 presents the levels of four parameters. The design of experiments (DOE) is used to construct a set of orthogonal arrays  $L_{16} (4^4)$  to combine parameters. For each parameter combination, the average objective value obtained from 30 independent runs of the proposed algorithm is used as the response variable (RV). Table 4 provides detailed parameter combinations and corresponding RV values for each parameter combination. Based on Table 4, the factor level trend of the four parameters is shown in Fig. 3, and the significance ranks of the four parameters are given in Table 5. It can be concluded that HABC-GA performs best where  $pop\_size = 20$ ,  $p_c = 0.6$ ,  $p_m = 0.8$ , and  $stag\_max = 15$ .

**Table 5**  
Ranking of four parameters.

| Level      | $pop\_size$ | $p_c$ | $p_m$ | $stag\_max$ |
|------------|-------------|-------|-------|-------------|
| 1          | 7.19        | 7.01  | 7.04  | 7.01        |
| 2          | 6.89        | 7.03  | 7.02  | 6.99        |
| 3          | 6.97        | 6.96  | 6.96  | 6.94        |
| 4          | 6.95        | 7.00  | 6.98  | 7.07        |
| Range      | 2.2         | 0.4   | 0.8   | 0.4         |
| Importance | 1           | 4     | 3     | 2           |

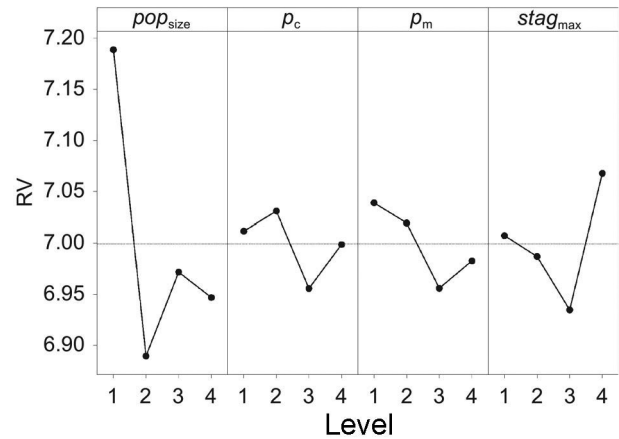


Fig. 3. Factor level trend of the four parameters.

### 5.4. Efficiency of the genetic augmented exploration mechanism

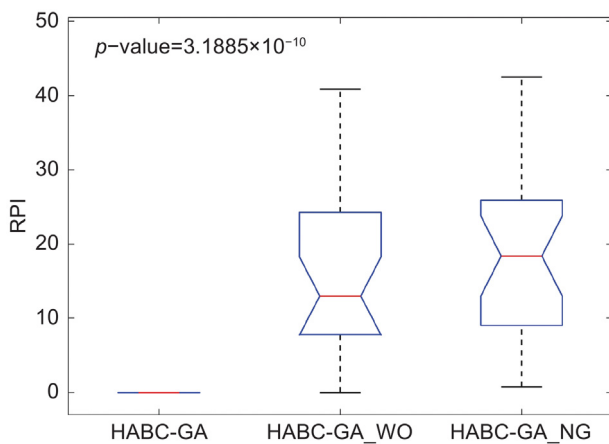
The genetic augmented exploration mechanism discussed in Section 4.2 is designed to enhance the algorithm's global search ability. To evaluate the effectiveness of this mechanism, we implemented three algorithms. The first algorithm is HABC-GA, which incorporated the proposed mechanism. The second algorithm is HABC-GA\_WO, which replaced the proposed mechanism with the genetic evolution operators described in Ref. [39]. The third algorithm is HABC-GA\_NG, which employed the original bee exploration strategy used by ABC in the employed bee phase. All other components of the three algorithms remain identical for a fair comparison.

Table 6 presents the results for the 24 tested instances evaluated using three algorithms. The first column lists the environment maps, the second column lists the instances, and the subsequent four columns present the average objective values along with the corresponding RPI values for each algorithm.

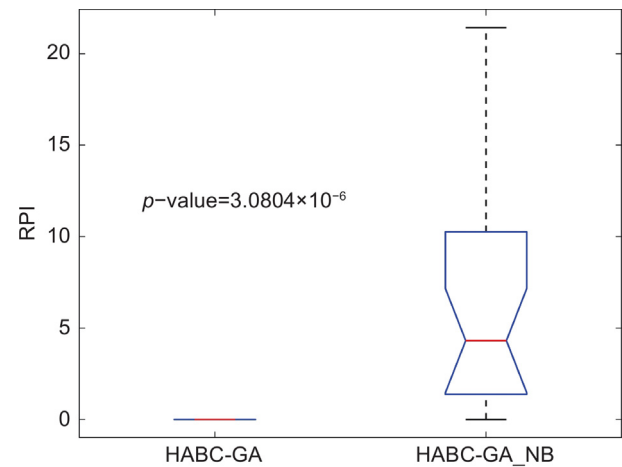
The comparison results show that (1) HABC-GA achieved better results in all 24 instances. (2) In the last two columns of Table 6, HABC-GA achieved an average PRI value of 0, significantly lower than the average RPI that of the second-best performance algorithm that of 15.42. This shows that the results of HABC-GA are obviously better than those of the second-best performance algorithm. Furthermore, a multifactor analysis of variance (ANOVA) was conducted to evaluate the statistical significance of the differences among the three compared algorithms. Fig. 4 presents the ANOVA result, with a  $p$ -value of  $3.1885 \times 10^{-10}$ , which is significantly lower than 0.05, confirming a substantial

**Table 6**  
Comparison results of HABC-GA, HABC-GA\_NG, and HABC-GA\_WO.

| Instances     | Algorithms   |            |            | RPI      |            |            |
|---------------|--------------|------------|------------|----------|------------|------------|
|               | HABC-GA      | HABC-GA_NG | HABC-GA_WO | HABC-GA  | HABC-GA_NG | HABC-GA_WO |
| Inst_map_A_MD | <b>6.49</b>  | 7.26       | 7.92       | <b>0</b> | 11.88      | 22.09      |
| Inst_map_A_SD | <b>4.28</b>  | 4.89       | 5.21       | <b>0</b> | 14.11      | 21.66      |
| Inst_map_A_HA | <b>5.06</b>  | 5.38       | 5.87       | <b>0</b> | 6.32       | 15.99      |
| Inst_map_A_VA | <b>3.04</b>  | 3.32       | 3.56       | <b>0</b> | 9.42       | 17.35      |
| Inst_map_B_MD | <b>6.35</b>  | 8.19       | 8.43       | <b>0</b> | 28.85      | 32.67      |
| Inst_map_B_SD | <b>6.86</b>  | 8.27       | 8.08       | <b>0</b> | 20.62      | 17.82      |
| Inst_map_B_HA | <b>6.74</b>  | 8.74       | 8.58       | <b>0</b> | 29.57      | 27.23      |
| Inst_map_B_VA | <b>2.30</b>  | 2.31       | 2.32       | <b>0</b> | 0.57       | 0.75       |
| Inst_map_C_MD | <b>4.93</b>  | 5.88       | 6.44       | <b>0</b> | 19.18      | 30.62      |
| Inst_map_C_SD | <b>6.38</b>  | 7.56       | 7.59       | <b>0</b> | 18.59      | 18.98      |
| Inst_map_C_HA | <b>10.96</b> | 13.79      | 13.62      | <b>0</b> | 25.82      | 24.29      |
| Inst_map_C_VA | <b>6.80</b>  | 8.49       | 8.48       | <b>0</b> | 24.90      | 24.80      |
| Inst_map_D_MD | <b>4.70</b>  | 6.62       | 6.69       | <b>0</b> | 40.85      | 42.48      |
| Inst_map_D_SD | <b>6.01</b>  | 7.80       | 7.64       | <b>0</b> | 29.82      | 27.08      |
| Inst_map_D_HA | <b>4.01</b>  | 4.47       | 4.53       | <b>0</b> | 11.55      | 13.00      |
| Inst_map_D_VA | <b>4.52</b>  | 4.85       | 4.79       | <b>0</b> | 7.14       | 6.00       |
| Inst_map_E_MD | <b>4.61</b>  | 5.70       | 5.86       | <b>0</b> | 23.66      | 27.02      |
| Inst_map_E_SD | <b>4.64</b>  | 5.03       | 5.00       | <b>0</b> | 8.45       | 7.82       |
| Inst_map_E_HA | <b>4.92</b>  | 5.71       | 5.95       | <b>0</b> | 16.00      | 20.83      |
| Inst_map_E_VA | <b>2.15</b>  | 2.15       | 2.22       | <b>0</b> | 0.00       | 2.93       |
| Inst_map_F_MD | <b>3.41</b>  | 3.55       | 3.67       | <b>0</b> | 4.13       | 7.78       |
| Inst_map_F_SD | <b>4.51</b>  | 4.92       | 5.16       | <b>0</b> | 9.02       | 14.41      |
| Inst_map_F_HA | <b>3.56</b>  | 3.58       | 3.69       | <b>0</b> | 0.70       | 3.84       |
| Inst_map_F_VA | <b>4.73</b>  | 5.16       | 5.22       | <b>0</b> | 8.98       | 10.21      |
| Means         | <b>5.08</b>  | 5.98       | 6.11       | <b>0</b> | 15.42      | 18.24      |



**Fig. 4.** ANOVA results for the three compared algorithms.



**Fig. 5.** ANOVA result for HABC-GA and HABC-GA\_NB.

difference between the HABC-GA and the other two algorithms. In general, the proposed genetic augmented exploration mechanism is effective.

**5.5. Efficiency of the objective-guided optimization strategy**

The objective-guided optimization strategy discussed in Section 4.3 enhances the algorithm’s local search capability and accelerates convergence. To verify the effectiveness of this strategy, we designed two algorithms. One is HABC-GA, which uses the objective-guided optimization strategy in the onlooker bee phase. The other is HABC-GA-NB, which replaces this strategy with a random mutation operator in the onlooker bee phase. All other components of both algorithms remain identical. Table 7 displays the comparison results of the two algorithms. From the data, it can be seen that (1) HABC-GA achieved the best results on the given 24 instances. (2) The last two columns of Table 7 show that the average RPI value of HABC-GA is 0, significantly lower than that of HABC-GA\_NB, which is 5.87. These results validate the effectiveness of the proposed objective-guided optimization

strategy. Fig. 5 shows the ANOVA results between HABC-GA\_NB and HABC-GA, presenting a  $p$ -value of  $3.0848 \times 10^{-6}$ , which is less than the threshold of 0.05, signifying a statistically significant difference between HABC-GA and HABC-GA\_NB.

**5.6. Efficiency of the dual exploration restart strategy**

The dual exploration restart strategy discussed in Section 4.4 is implemented to replace solutions that have experienced stagnation. To assess the effectiveness of this strategy, two algorithms are tested. One is HABC-GA, which uses a dual exploration restart strategy to replace stagnation solutions. The other is HABC-GA-ND, which replaces stagnant solutions with randomly generated new solutions. Table 8 shows the comparison results of the two compared algorithms. From this table, it can be concluded that (1) HABC-GA achieved the best results in all 24 instances. (2) The final row of Table 8 shows that the average RPI value of HABC-GA is notably lower than that of HABC-GA\_ND. The above results confirm the effectiveness of the proposed dual exploration restart

**Table 7**  
Comparison results between HABC-GA and HABC-GA\_NB.

| Tested environments | Instances     | Algorithms   |            | RPI      |            |
|---------------------|---------------|--------------|------------|----------|------------|
|                     |               | HABC-GA      | HABC-GA_NB | HABC-GA  | HABC-GA_NB |
| Map A               | Inst_map_A_MD | <b>6.49</b>  | 7.17       | <b>0</b> | 10.56      |
|                     | Inst_map_A_SD | <b>4.28</b>  | 4.47       | <b>0</b> | 4.29       |
|                     | Inst_map_A_HA | <b>5.06</b>  | 5.24       | <b>0</b> | 3.57       |
|                     | Inst_map_A_VA | <b>3.04</b>  | 3.20       | <b>0</b> | 5.30       |
| Map B               | Inst_map_B_MD | <b>6.35</b>  | 7.08       | <b>0</b> | 11.38      |
|                     | Inst_map_B_SD | <b>6.86</b>  | 6.96       | <b>0</b> | 1.40       |
|                     | Inst_map_B_HA | <b>6.74</b>  | 7.31       | <b>0</b> | 8.45       |
|                     | Inst_map_B_VA | <b>2.30</b>  | 2.30       | <b>0</b> | 0.03       |
| Map C               | Inst_map_C_MD | <b>4.93</b>  | 5.25       | <b>0</b> | 6.49       |
|                     | Inst_map_C_SD | <b>6.38</b>  | 6.52       | <b>0</b> | 2.20       |
|                     | Inst_map_C_HA | <b>10.96</b> | 11.85      | <b>0</b> | 8.14       |
|                     | Inst_map_C_VA | <b>6.80</b>  | 7.60       | <b>0</b> | 11.89      |
| Map D               | Inst_map_D_MD | <b>4.70</b>  | 5.70       | <b>0</b> | 21.42      |
|                     | Inst_map_D_SD | <b>6.01</b>  | 6.66       | <b>0</b> | 10.83      |
|                     | Inst_map_D_HA | <b>4.01</b>  | 4.06       | <b>0</b> | 1.41       |
|                     | Inst_map_D_VA | <b>4.52</b>  | 4.57       | <b>0</b> | 0.96       |
| Laser_Map E         | Inst_map_E_MD | <b>4.61</b>  | 5.07       | <b>0</b> | 9.96       |
|                     | Inst_map_E_SD | <b>4.64</b>  | 4.84       | <b>0</b> | 4.32       |
|                     | Inst_map_E_HA | <b>4.92</b>  | 5.09       | <b>0</b> | 3.36       |
|                     | Inst_map_E_VA | <b>2.15</b>  | 2.16       | <b>0</b> | 0.11       |
| Laser_Map F         | Inst_map_F_MD | <b>3.41</b>  | 3.45       | <b>0</b> | 1.33       |
|                     | Inst_map_F_SD | <b>4.51</b>  | 4.54       | <b>0</b> | 0.73       |
|                     | Inst_map_F_HA | <b>3.56</b>  | 3.56       | <b>0</b> | 0.00       |
|                     | Inst_map_F_VA | <b>4.73</b>  | 5.34       | <b>0</b> | 12.79      |
| Means               | -             | <b>5.08</b>  | 5.42       | <b>0</b> | 5.87       |

**Table 8**  
Comparison results between HABC-GA and HABC-GA\_ND.

| Tested environments | Instances     | Algorithms   |            | RPI      |            |
|---------------------|---------------|--------------|------------|----------|------------|
|                     |               | HABC-GA      | HABC-GA_ND | HABC-GA  | HABC-GA_ND |
| Map A               | Inst_map_A_MD | <b>6.49</b>  | 7.29       | <b>0</b> | 12.31      |
|                     | Inst_map_A_SD | <b>4.28</b>  | 4.41       | <b>0</b> | 3.03       |
|                     | Inst_map_A_HA | <b>5.06</b>  | 5.29       | <b>0</b> | 4.40       |
|                     | Inst_map_A_VA | <b>3.04</b>  | 3.04       | <b>0</b> | 0.13       |
| Map B               | Inst_map_B_MD | <b>6.35</b>  | 7.12       | <b>0</b> | 12.03      |
|                     | Inst_map_B_SD | <b>6.86</b>  | 7.38       | <b>0</b> | 7.58       |
|                     | Inst_map_B_HA | <b>6.74</b>  | 7.14       | <b>0</b> | 5.84       |
|                     | Inst_map_B_VA | <b>2.30</b>  | 2.30       | <b>0</b> | 0.00       |
| Map C               | Inst_map_C_MD | <b>4.93</b>  | 5.44       | <b>0</b> | 10.27      |
|                     | Inst_map_C_SD | <b>6.38</b>  | 6.43       | <b>0</b> | 0.91       |
|                     | Inst_map_C_HA | <b>10.96</b> | 11.79      | <b>0</b> | 7.56       |
|                     | Inst_map_C_VA | <b>6.80</b>  | 7.43       | <b>0</b> | 9.32       |
| Map D               | Inst_map_D_MD | <b>4.70</b>  | 5.58       | <b>0</b> | 18.79      |
|                     | Inst_map_D_SD | <b>6.01</b>  | 6.39       | <b>0</b> | 6.34       |
|                     | Inst_map_D_HA | <b>4.01</b>  | 4.15       | <b>0</b> | 3.51       |
|                     | Inst_map_D_VA | <b>4.52</b>  | 4.53       | <b>0</b> | 0.15       |
| Laser_Map E         | Inst_map_E_MD | <b>4.61</b>  | 4.79       | <b>0</b> | 3.87       |
|                     | Inst_map_E_SD | <b>4.64</b>  | 4.85       | <b>0</b> | 4.49       |
|                     | Inst_map_E_HA | <b>4.92</b>  | 5.09       | <b>0</b> | 3.28       |
|                     | Inst_map_E_VA | <b>2.15</b>  | 2.16       | <b>0</b> | 0.36       |
| Laser_Map F         | Inst_map_F_MD | <b>3.41</b>  | 3.46       | <b>0</b> | 1.45       |
|                     | Inst_map_F_SD | <b>4.51</b>  | 4.6        | <b>0</b> | 2.23       |
|                     | Inst_map_F_HA | <b>3.56</b>  | 3.58       | <b>0</b> | 0.67       |
|                     | Inst_map_F_VA | <b>4.73</b>  | 5.07       | <b>0</b> | 7.07       |
| Means               | -             | <b>5.08</b>  | 5.39       | <b>0</b> | 5.23       |

strategy.

Fig. 6 shows the ANOVA results of HABC-GA and HABC-GA\_ND, with a  $p$ -value of  $2.2816 \times 10^{-6}$ , indicating significant differences between the two algorithms.

To evaluate the impact of each component on the algorithm, we conducted an ablation experiment. Fig. 7 shows the Tukey test plots for the three components of HABC-GA obtained under a 95% confidence interval. The results show that the genetic augmented exploration mechanism is the most critical component for HABC-GA, followed by the objective-guided optimization strategy and the dual exploration restart strategy, in descending

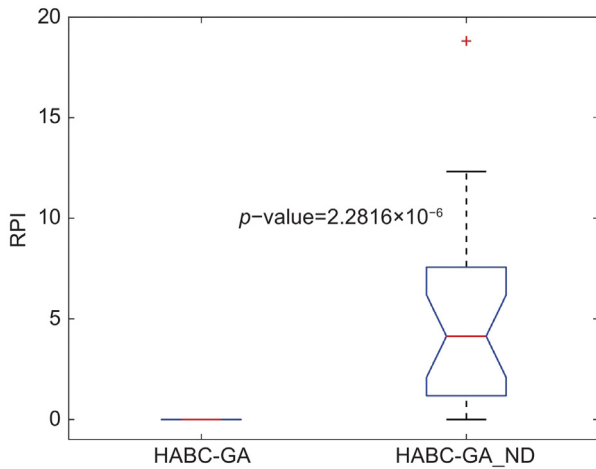
order of importance.

### 5.7. Comparison with other algorithms

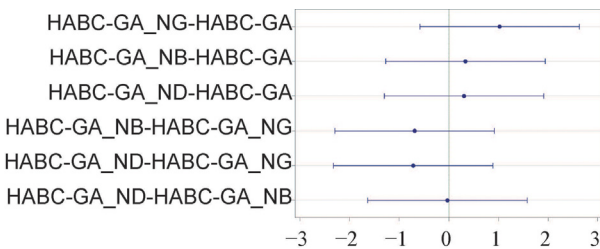
To further assess the effectiveness of the proposed HABC-GA, five efficient algorithms are compared. For each compared algorithm, the results from 30 independent runs are collected for detailed analysis. The comparison focused on two metrics for each test instance, i.e., the minimum objective value and the average objective value achieved by the algorithms. Table 9 summarizes the results for 24 test instances, where the first column lists test

**Table 9**  
Comparison results of compared algorithms on objective function values.

| Tested instances | SPSFA |       | PSO-SA |       | IAACO |       | PSO-MFB |       | IMOABC |             | HABC-GA |              |
|------------------|-------|-------|--------|-------|-------|-------|---------|-------|--------|-------------|---------|--------------|
|                  | Min   | Avg   | Min    | Avg   | Min   | Avg   | Min     | Avg   | Min    | Avg         | Min     | Avg          |
| Inst_map_A_MD    | 6.88  | 8.51  | 6.66   | 9.19  | 6.57  | 7.91  | 7.07    | 9.93  | 5.84†  | 7.49        | 5.88    | <b>6.49</b>  |
| Inst_map_A_SD    | 4.50  | 5.51  | 4.71   | 6.31  | 4.26  | 4.83  | 5.06    | 7.25  | 4.00†  | 4.76        | 4.09    | <b>4.29</b>  |
| Inst_map_A_HA    | 5.32  | 6.34  | 5.31   | 7.18  | 4.95  | 5.98  | 5.51    | 7.98  | 4.85   | 6.04        | 4.75†   | <b>5.06</b>  |
| Inst_map_A_VA    | 3.01  | 3.65  | 2.99   | 4.43  | 2.90  | 3.55  | 3.77    | 5.28  | 2.74†  | 3.26        | 2.96    | <b>3.04</b>  |
| Inst_map_B_MD    | 7.67  | 8.60  | 6.61   | 8.70  | 6.15  | 7.63  | 7.96    | 10.24 | 5.96   | 7.86        | 5.95†   | <b>6.35</b>  |
| Inst_map_B_SD    | 7.11  | 8.89  | 7.16   | 9.57  | 6.38  | 8.05  | 7.27    | 10.06 | 6.29†  | 7.82        | 6.39    | <b>6.86</b>  |
| Inst_map_B_HA    | 7.69  | 9.20  | 7.99   | 9.55  | 6.76  | 7.59  | 7.95    | 10.67 | 6.46   | 7.69        | 6.4†    | <b>6.74</b>  |
| Inst_map_B_VA    | 2.28  | 2.33  | 2.26   | 2.32  | 2.24  | 2.29  | 2.30    | 2.51  | 2.22   | <b>2.29</b> | 2.1†    | 2.30         |
| Inst_map_C_MD    | 4.69  | 6.95  | 5.52   | 7.27  | 4.53  | 5.60  | 5.10    | 6.55  | 4.42†  | 5.77        | 4.49    | <b>4.93</b>  |
| Inst_map_C_SD    | 6.69  | 7.96  | 7.22   | 8.06  | 6.26  | 6.91  | 6.73    | 7.64  | 6.06†  | 6.66        | 6.17    | <b>6.38</b>  |
| Inst_map_C_HA    | 12.62 | 14.07 | 12.74  | 14.73 | 11.24 | 12.68 | 12.35   | 14.22 | 10.76  | 12.12       | 10.46†  | <b>10.96</b> |
| Inst_map_C_VA    | 6.99  | 9.25  | 8.12   | 9.27  | 7.40  | 7.80  | 7.86    | 8.97  | 6.38   | 7.54        | 6.34†   | <b>6.80</b>  |
| Inst_map_D_MD    | 6.05  | 6.83  | 6.13   | 7.08  | 5.34  | 5.91  | 5.94    | 6.75  | 5.24   | 5.87        | 4.67†   | <b>4.70</b>  |
| Inst_map_D_SD    | 6.33  | 8.12  | 6.89   | 8.47  | 5.98  | 7.18  | 6.68    | 8.03  | 5.27†  | 7.00        | 5.30    | <b>6.01</b>  |
| Inst_map_D_HA    | 3.93  | 4.86  | 3.96   | 5.01  | 3.92  | 4.40  | 3.94    | 4.54  | 3.82†  | 4.39        | 3.87    | <b>4.01</b>  |
| Inst_map_D_VA    | 4.50  | 5.07  | 4.58   | 5.22  | 4.49  | 4.76  | 4.50    | 5.03  | 4.49†  | 4.73        | 4.49†   | <b>4.52</b>  |
| Inst_map_E_MD    | 5.07  | 6.11  | 5.40   | 6.42  | 4.80  | 5.39  | 5.79    | 7.58  | 4.59   | 5.45        | 4.13†   | <b>4.61</b>  |
| Inst_map_E_SD    | 4.31  | 5.15  | 4.57   | 5.31  | 4.20  | 4.63  | 4.77    | 6.02  | 4.160† | <b>4.40</b> | 4.33    | 4.64         |
| Inst_map_E_HA    | 5.28  | 5.97  | 5.42   | 6.29  | 4.80  | 5.52  | 5.53    | 6.90  | 4.78   | 5.38        | 4.7†    | <b>4.92</b>  |
| Inst_map_E_VA    | 2.11  | 2.31  | 2.14   | 2.45  | 2.12  | 2.22  | 2.35    | 2.87  | 2.11†  | 2.19        | 2.11†   | <b>2.15</b>  |
| Inst_map_F_MD    | 3.45  | 3.74  | 3.51   | 3.72  | 3.40  | 3.56  | 3.56    | 4.32  | 3.30†  | 3.60        | 3.32    | <b>3.41</b>  |
| Inst_map_F_HA    | 4.67  | 5.43  | 4.83   | 5.53  | 4.39  | 4.82  | 5.03    | 6.38  | 4.26†  | 4.73        | 4.28    | <b>4.51</b>  |
| Inst_map_F_VA    | 3.58  | 3.80  | 3.59   | 3.85  | 3.53  | 3.61  | 3.57    | 4.24  | 3.53†  | 3.63        | 3.54    | <b>3.56</b>  |
| Inst_map_F_VA    | 5.12  | 5.50  | 5.01   | 5.47  | 4.84  | 5.17  | 5.45    | 5.98  | 4.72   | 5.15        | 4.41†   | <b>4.73</b>  |
| Means            | 5.41  | 6.42  | 5.56   | 6.73  | 5.06  | 5.75  | 5.67    | 7.08  | 5.24   | 5.66        | 4.81†   | <b>5.08</b>  |

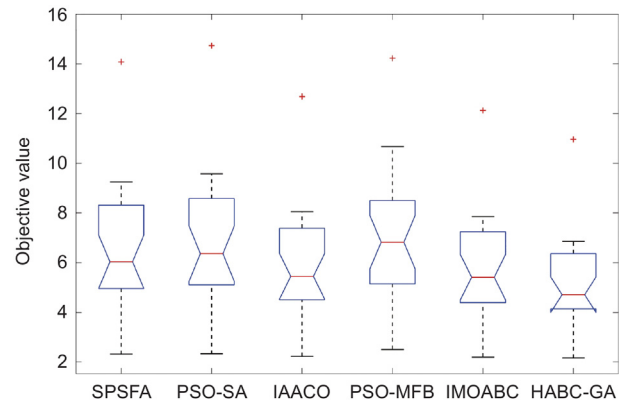


**Fig. 6.** ANOVA result for HABC-GA and HABC-GA\_ND.



**Fig. 7.** Tukey test and 95% LSD intervals among the components of HABC-GA.

instances and the subsequent 12 columns display the minimum objective values and average objective values achieved by the six algorithms. The value with the symbol “†” represents the best value among the minimum objective values obtained by all compared algorithms for each instance, while the bold value represents the best values among the average objective values obtained by all compared algorithms for each instance.



**Fig. 8.** ANOVA result of the six compared algorithms.

From **Table 9**, the key findings can be summarized as follows:  
 (1) The HABC-GA achieved the best average objective values in 22 out of 24 instances compared to the other algorithms.  
 (2) HABC-GA achieved the minimum objective value in 12 out of 24 instances. The optimal minimum objective values for the other 12 instances are obtained by the IMO-ABC algorithm. We can see that on these 12 instances, although the best minimum objective value obtained by HABC-GA is slightly higher than that of IMO-ABC, the average objective value obtained by HABC-GA is much lower than that of IMO-ABC. This phenomenon indicates that while obtaining similar results, the proposed HABC-GA is more stable. In addition, in terms of the average objective value of all instances, HABC-GA consistently outperforms other instances.  
 (3) As shown in the last row of the table, HABC-GA achieved an average target value of 5.082 out of 24 instances, a decrease of 10.18% compared to the second-best average value. **Fig. 8** shows the ANOVA results of the six compared algorithms. From the figure, it can be seen that the HABC-GA algorithm achieved the minimum median. In conclusion, The HABC-GA algorithm has higher performance in solving safe and smooth path planning.

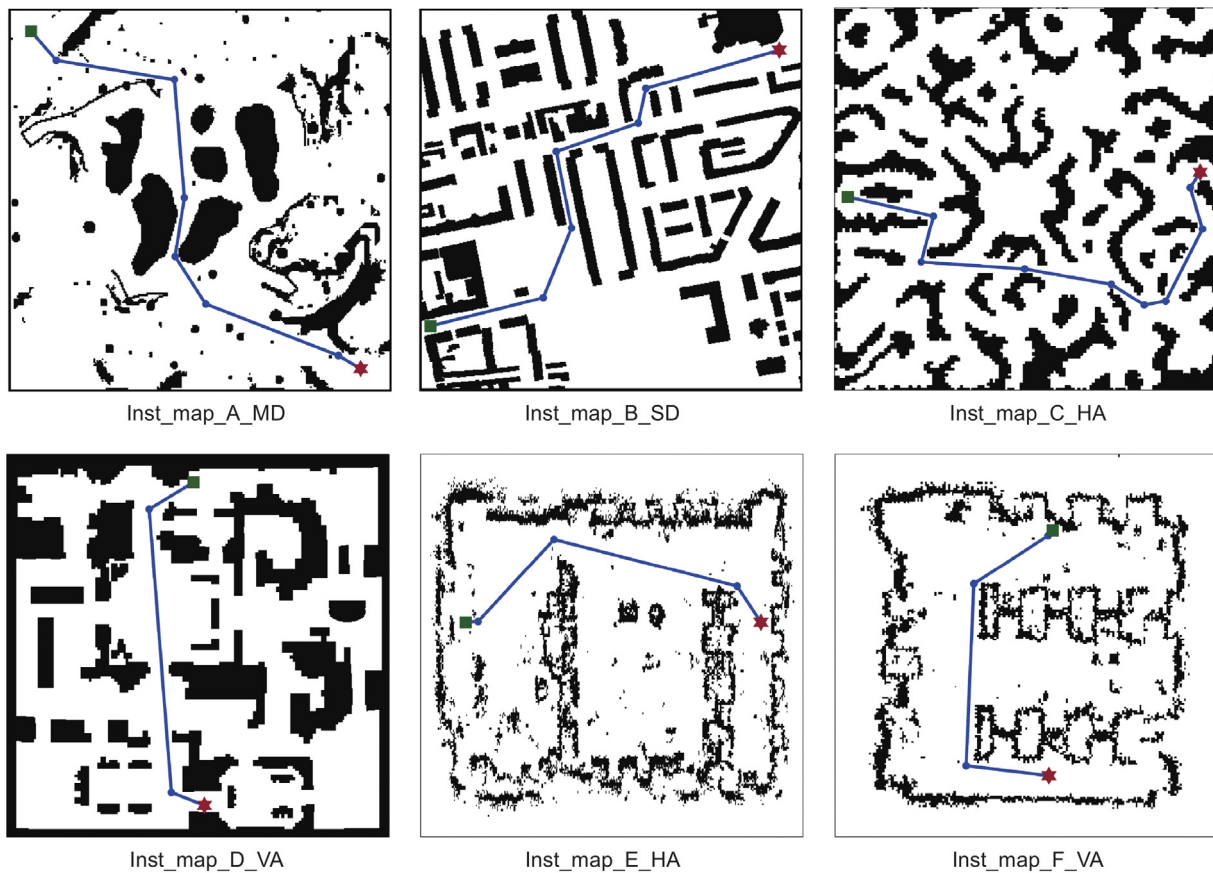


Fig. 9. The solutions with the best objective value obtained by HABC-GA on six instances.

To intuitively demonstrate the effectiveness of the algorithm, we present the optimal solutions for six representative instances: Inst\_map\_A\_MD, Inst\_map\_B\_SD, Inst\_map\_C\_HA, Inst\_map\_D\_VA, Inst\_map\_E\_HA, and Inst\_map\_F\_VA. As shown in Fig. 9, the start point is indicated by a green square, the target point is represented by a red hexagon, and the obtained path is depicted by a blue line.

To effectively illustrate the evolution process of the compared algorithms, Fig. 10 shows the convergence curves of the objective values of the compared algorithms on the six instances. All comparison algorithms have a cutoff condition of 50 s of runtime. As observed in Fig. 10, HABC-GA consistently achieves the majority of optimal objective values in the early stage, indicating that the utilization of RRT\* in the initialization phase successfully generates diverse solutions. Additionally, among all compared algorithms, HABC-GA demonstrates superior convergence speed, underscoring the effectiveness of the proposed genetic augmented exploration mechanism and the objective-guided optimization strategy. In the later stage, HABC-GA continues to produce the lowest objective function values, confirming its capability to provide high-quality solutions for the safe and smooth path planning problem.

## 6. Conclusions and future work

In this work, we proposed the HABC-GA to achieve safe and smooth path planning for mobile robot. First, the genetic augmented exploration mechanism is developed, including the path crossover, adaptive obstacle-aware mutation, and dynamic elite

selection operators, to enhance population quality and the algorithm's global exploration capability. In addition, the objective-guided optimization strategy is designed to optimize each individual while accelerating convergence. Moreover, we presented the dual exploration restart strategy, which selects the appropriate restart method based on individual objective values to update the stagnant solutions. Finally, experiments conducted on 24 instances in six different environments showed that HABC-GA outperforms the five efficient algorithms in achieving safe and smooth path planning.

The limitations of this research include: (1) Our main goal is to find a safe and smooth path, ignoring the adaptability of the path to dynamic environments. (2) We only considered path planning for a single robot and did not take into account path planning for multiple agents.

Future research will consider the following aspects: (1) The key to achieving safe and stable path planning in dynamic environments will be addressed. (2) Multi-agent path planning is considered as a future research direction.

## CRedit authorship contribution statement

**Fan Ye:** Writing – original draft, Validation, Software, Methodology, Conceptualization. **Peng Duan:** Writing – review & editing, Validation, Funding acquisition. **Leilei Meng:** Funding acquisition. **Lingyan Xue:** Validation, Writing – review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

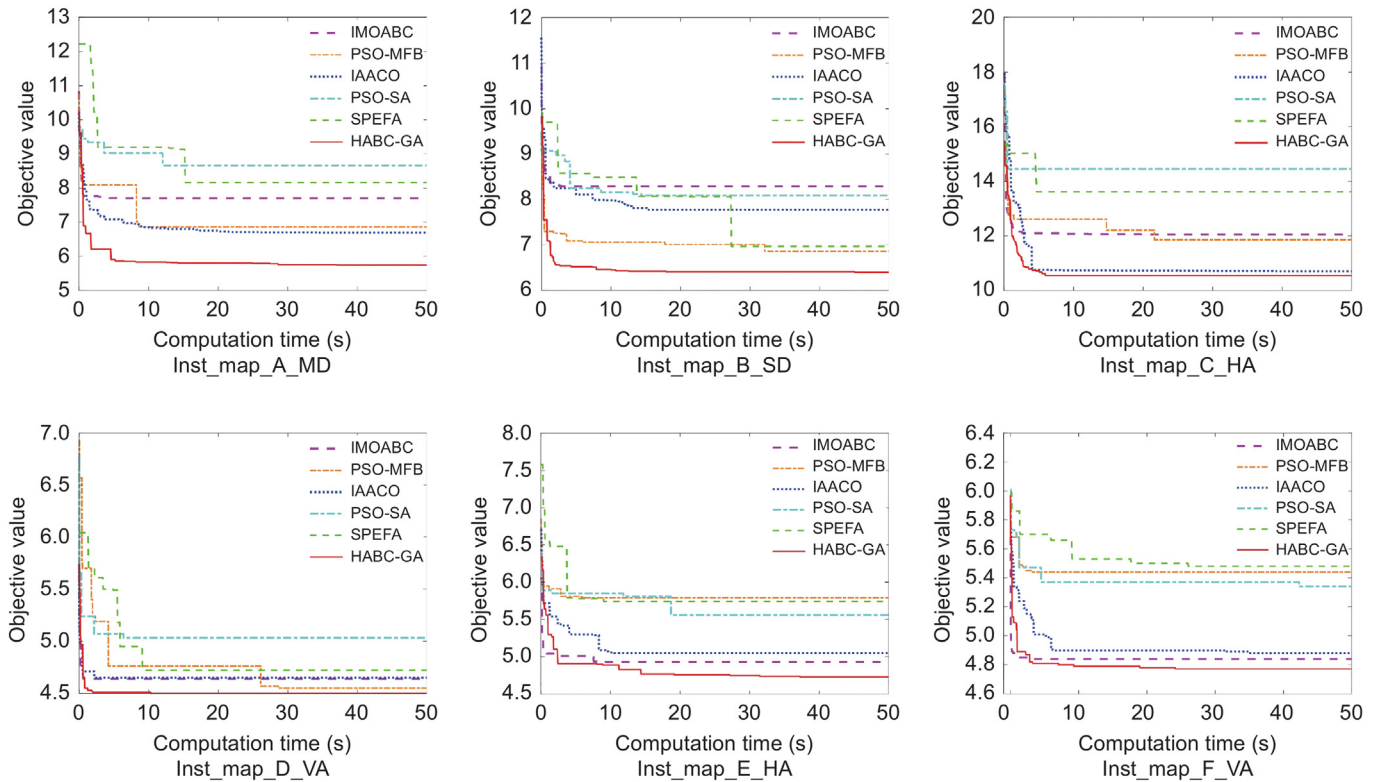


Fig. 10. Convergence curves of the compared algorithms on six instances.

## Acknowledgments

This work was supported by the Opening Fund of Shandong Key Laboratory of Ubiquitous Intelligent Computing, China, the National Natural Science Foundation of China (52205529), the Natural Science Foundation of Shandong Province, China (ZR2021QE195 and ZR2021MD090), and the Discipline with Strong Characteristics of Liaocheng University-Intelligent Science and Technology, China (319462208).

## References

- [1] I. Nielsen, Q.V. Dang, G. Bocewicz, Z. Banaszak, A methodology for implementation of mobile robot in adaptive manufacturing environments, *J. Intell. Manuf.* 28 (2017) 1171–1188.
- [2] R. Raj, A. Kos, A comprehensive study of mobile robot: history, developments, applications, and future research perspectives, *Appl. Sci.* 12 (14) (2022) 6951.
- [3] X. Zhang, Y. Guo, J. Yang, D. Li, Y. Wang, R. Zhao, Many-objective evolutionary algorithm based agricultural mobile robot route planning, *Comput. Electron. Agric.* 200 (2022) 107274.
- [4] A. Khan, S. Gupta, S.K. Gupta, Emerging UAV technology for disaster detection, mitigation, response, and preparedness, *J. Field Robotics* 39 (6) (2022) 905–955.
- [5] M. Zhang, Q. Zhang, R. Song, P.L. Rosin, W. Zhang, Ship landmark: an informative ship image annotation and its applications, *IEEE Trans. Intell. Transp. Syst.* 25 (11) (2024) 17778–17793.
- [6] M.B. Alatise, G.P. Hancke, A review on challenges of autonomous mobile robot and sensor fusion methods, *IEEE Access* 8 (2020) 39830–39846.
- [7] L. Liu, X. Wang, X. Yang, H. Liu, J. Li, P. Wang, Path planning techniques for mobile robots: review and prospect, *Expert Syst. Appl.* 227 (2023) 120254.
- [8] O.A.A. Salama, M.E.H. Eltaib, H.A. Mohamed, O. Salah, RCD: radial cell decomposition algorithm for mobile robot path planning, *IEEE Access* 9 (2021) 149982–149992.
- [9] U. Orozco-Rosas, O. Montiel, R. Sepúlveda, Mobile robot path planning using membrane evolutionary artificial potential field, *Appl. Soft Comput.* 77 (2019) 236–251.
- [10] J. Luo, Z. Wang, K. Pan, Reliable path planning algorithm based on improved artificial potential field method, *IEEE Access* 10 (2022) 108276–108284.
- [11] J.D. Marble, K.E. Bekris, Asymptotically near-optimal planning with probabilistic roadmap spanners, *IEEE Trans. Robot.* 29 (2) (2013) 432–444.
- [12] B. Song, Z. Wang, L. Zou, An improved PSO algorithm for smooth path planning of mobile robots using continuous high-degree bezier curve, *Appl. Soft Comput.* 100 (2021) 106960.
- [13] Q. Luo, H. Wang, Y. Zheng, J. He, Research on path planning of mobile robot based on improved ant colony algorithm, *Neural Comput. Appl.* 32 (6) (2020) 1555–1566.
- [14] P.G. Luan, N.T. Thinh, Hybrid genetic algorithm based smooth global-path planning for a mobile robot, *Mech. Des. Struct. Mach.* 51 (3) (2023) 1758–1774.
- [15] H. Qu, K. Xing, T. Alexander, An improved genetic algorithm with co-evolutionary strategy for global path planning of multiple mobile robots, *Neurocomputing* 120 (2013) 509–517.
- [16] C. Lamini, S. Benhlila, A. Elbekri, Genetic algorithm based approach for autonomous mobile robot path planning, *Procedia Comput. Sci.* 127 (2018) 180–189.
- [17] R. Sarkar, D. Barman, N. Chowdhury, Domain knowledge based genetic algorithms for mobile robot path planning having single and multiple targets, *J. King Saud Univ.-Comput. Inf. Sci.* 34 (7) (2022) 4269–4283.
- [18] L. Cheng, L. Zhong, X. Zhang, J. Xing, A staged adaptive firefly algorithm for UAV charging planning in wireless sensor networks, *Comput. Commun.* 161 (2020) 132–141.
- [19] H. Liu, B. Xu, D. Lu, G. Zhang, A path planning approach for crowd evacuation in buildings based on improved artificial bee colony algorithm, *Appl. Soft Comput.* 68 (2018) 360–376.
- [20] F.H. Ajeil, I.K. Ibraheem, M.A. Sahib, A.J. Humaidi, Multi-objective path planning of an autonomous mobile robot using hybrid PSO-MFB optimization algorithm, *Appl. Soft Comput.* 89 (2020) 106076.
- [21] Q. Yuan, R. Sun, X. Du, Path planning of mobile robots based on an improved particle swarm optimization algorithm, *Processes* 11 (1) (2022) 26.
- [22] G. Zhang, C. Li, M. Gao, L. Sheng, Global smooth path planning for mobile robots using a novel adaptive particle swarm optimization, in: *Proceedings of IEEE Chinese Control Conference (CCC)*, Guangzhou, China, 2019, pp. 2124–2129.

- [23] S. Lin, A. Liu, J. Wang, X. Kong, An intelligence-based hybrid PSO-SA for mobile robot path planning in warehouse, *J. Comput. Sci.* 67 (2023) 101938.
- [24] Q.G. Su, W.W. Yu, J. Liu, Mobile robot path planning based on improved ant colony algorithm, in: *Proceedings of IEEE Asia-Pacific Conference on Communications Technology and Computer Science (ACCTCS)*, Shenyang, China, 2021, pp. 220–224.
- [25] C. Miao, G. Chen, C. Yan, Y. Wu, Path planning optimization of indoor mobile robot based on adaptive ant colony algorithm, *Comput. Ind. Eng.* 156 (2021) 107230.
- [26] W. Hou, Z. Xiong, C. Wang, H. Chen, Enhanced ant colony algorithm with communication mechanism for mobile robot path planning, *Robot. Auton. Syst.* 148 (2022) 103949.
- [27] L. Wu, X. Huang, J. Cui, C. Liu, W. Xiao, Modified adaptive ant colony optimization algorithm and its application for solving path planning of mobile robot, *Expert Syst. Appl.* 215 (2023) 119410.
- [28] J. Cui, L. Wu, X. Huang, D. Xu, C. Liu, W. Xiao, Multi-strategy adaptable ant colony optimization algorithm and its application in robot path planning, *Knowl.-Based Syst.* 288 (2024) 111459.
- [29] M.A. Contreras-Cruz, V. Ayala-Ramirez, U.H. Hernandez-Belmonte, Mobile robot path planning using artificial bee colony and evolutionary programming, *Appl. Soft Comput.* 30 (2015) 319–328.
- [30] F. Xu, H. Li, C. Pun, H. Hu, Y. Li, Y. Song, H. Gao, A new global best guided artificial bee colony algorithm with application in robot path planning, *Appl. Soft Comput.* 88 (2020) 106037.
- [31] R.T. Kamil, M.J. Mohamed, B.K. Oleiwi, Path planning of mobile robot using improved artificial bee colony algorithm, *Eng. Technol. J.* 38 (2020) 1384–1395.
- [32] Y. Cui, W. Hu, A. Rahmani, A reinforcement learning based artificial bee colony algorithm with application in robot path planning, *Expert Syst. Appl.* 203 (2022) 117389.
- [33] Q. Cui, P. Liu, H. Du, H. Wang, X. Ma, Improved multi-objective artificial bee colony algorithm-based path planning for mobile robots, *Front. Neurobot.* 17 (2023) 1196683.
- [34] G. Li, C. Liu, W. Xiao, L. Tan, C. Li, T. Wang, Improved artificial bee colony algorithm to solve mobile robot path planning, in: *Proceedings of SPIE Third International Conference on Advanced Algorithms and Neural Networks (AANN)*, Qingdao, China, 2023, pp. 165–169.
- [35] X. Cui, C. Wang, Y. Xiong, L. Mei, S. Wu, More quickly-RRT\*: improved quick rapidly-exploring random tree star algorithm based on optimized sampling point with better initial solution and convergence rate, *Eng. Appl. Artif. Intell.* 133 (2024) 108246.
- [36] P. Duan, Z. Yu, K. Gao, L. Meng, Y. Han, F. Ye, Solving the multi-objective path planning problem for mobile robot using an improved NSGA-II algorithm, *Swarm Evol. Comput.* 87 (2024) 101576.
- [37] F. Li, X. Fan, Z. Hou, A firefly algorithm with self-adaptive population size for global path planning of mobile robot, *IEEE Access* 8 (2020) 168951–168964.
- [38] R. Stern, N. Sturtevant, A. Felner, S. Koenig, H. Ma, T. Walker, J. Li, D. Atzmon, L. Cohen, T. Kumar, Multi-agent pathfinding: definitions, variants, and benchmarks, in: *Proceedings of AAAI Twelfth International Symposium on Combinatorial Search (SoCS)*, California, USA, 2019, pp. 151–158.
- [39] Y. Xue, Mobile robot path planning with a non-dominated sorting genetic algorithm, *Appl. Sci.* 8 (11) (2018) 2253.