sensors

MDPI

Article

Real-Time Dynamic Path Planning of Mobile Robots: A Novel Hybrid Heuristic Optimization Algorithm

Qing Wu [1], Zeyu Chen [1], Lei Wang [1], Hao Lin [1], Zijing Jiang [1], Shuai Li [2] and Dechao Chen [1,*]

[1] School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China; wuqing@hdu.edu.cn (Q.W.); zeno_chen@163.com (Z.C.); 181050059@hdu.edu.cn (L.W.); 171050026@hdu.edu.cn (H.L.); jzj@hdu.edu.cn (Z.J.)

[2] Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong 999077, China; shuaili@polyu.edu.hk

* Correspondence: chdchao@hdu.edu.cn; Tel.: +86-13777481992

Received: 29 October 2019; Accepted: 26 December 2019; Published: 28 December 2019

Abstract: Mobile robots are becoming more and more widely used in industry and life, so the navigation of robots in dynamic environments has become an urgent problem to be solved. Dynamic path planning has, therefore, received more attention. This paper proposes a real-time dynamic path planning method for mobile robots that can avoid both static and dynamic obstacles. The proposed intelligent optimization method can not only get a better path but also has outstanding advantages in planning time. The algorithm used in the proposed method is a hybrid algorithm based on the beetle antennae search (BAS) algorithm and the artificial potential field (APF) algorithm, termed the BAS-APF method. By establishing a potential field, the convergence speed is accelerated, and the defect that the APF is easily trapped in the local minimum value is also avoided. At the same time, by setting a security scope to make the path closer to the available path in the real environment, the effectiveness and superiority of the proposed method are verified through simulative results.

Keywords: hybrid optimization algorithm; mobile robot; real-time path planning; dynamic obstacle avoidance; beetle antennae search algorithm (BAS)

1. Introduction

In recent decades, path planning has had important applications in many areas, such as mobile robots [1–6], unmanned aerial vehicle (UVA) [7,8], game artificial intelligence (AI) automatic pathfinding [9], etc. [10–13]. In particular, mobile robots have been put into practical use in many industries. For example, [14] introduced the practical application of sweeping robots for daily household cleaning, [15] proposed a robotic automatic battery sorting system for improving battery sorting efficiency, and [16] introduced many applications of mobile robots in agriculture: tilling, seeding, harvesting, etc.

For mobile robots, path planning consists of finding a feasible path to the target point in the workspace [17–23]. There are many related research results for this research direction [24–28]. For example, some traditional methods are: artificial potential field (APF), probabilistic roadmap method (PRM), and rapidly-exploring random trees (RRT) [29–31]. The paper presented in [32] used a hybrid algorithm of A* and RRT for indoor navigation of unmanned aerial vehicle. In [33], they proposed a bidirectional RRT based on potentially guidance to quickly plan paths in a messy environment. Most of these methods have certain advantages in terms of speed, but the path obtained lacks optimization. Some heuristic algorithms, such as A*, D*, are widely used in industry [34–36]. Xin et al. [37] proposed an improved A* algorithm, which extends the neighborhood propagation of the standard A* algorithm. In [38], the modified A* algorithm is used to plan the

path of the mobile robot. The connection between the path points of the A* algorithm is mainly modified. But these methods are mainly used in globally-known environments. There are also some intelligent optimization algorithms, such as ACO algorithm, genetic algorithm, and PSO algorithm, which also has corresponding results [39–42]. In [43], by combining artificial bee colony algorithm and evolutionary programming algorithm, they proposed a new path planning algorithm applied to path planning in two-dimensional static environment. In [44], they designed a non-dominated sorting genetic algorithm for multi-objective path planning in static environments. In [45], a heuristic PSO algorithm is proposed, which improves the PSO planning deficiency to a certain extent but only verifies the effectiveness of the algorithm in static environment. However, due to the characteristics of the group intelligence algorithm, the heuristic PSO algorithm is not particularly ideal in terms of time.

As can be seen from the above introduction, it is still necessary to study real-time dynamic path planning. It can also be seen that the research trend of path planning in recent years is also realized by a mixture of various algorithms to obtain better results.

The structure of this paper is organized as follows. Section 2 includes the definitions and formula descriptions for real-time path planning problems. In Section 3, we introduce the design ideas and implementation steps of the proposed beetle antennae search (BAS)-APF method in stages. Demonstration of the effectiveness of the algorithm through some numerical simulations for maps with dynamic obstacles is shown in Section 4. Finally, the conclusions are presented in Section 5. The main contributions of this paper are listed below.

- A novel hybrid intelligent optimization method, named BAS-APF, is proposed and applied to mobile robot dynamic path planning. The proposed method is divided into two phases. First, a feasible path from the current point to the end point is initialized based on the proposed algorithm. Then, path tracking is performed, and the path is optimized during path tracking without affecting the real-time performance of the algorithm.
- The proposed method has a good performance in both the planning time and the path length. And the advantages in planning speed are outstanding.
- The proposed method is simulative in our pre-set simulation environment maps and the real map collected by sensors, and its effectiveness and superiority are verified by comparison with other algorithms.

2. Problem Formulation

This section describes the definition of the path planning problem discussed in this article, some of the symbols used, and a brief introduction to the design basis of the proposed method. The method we proposed is called the BAS-APF. Therefore, as a background, as well as a brief introduction to the BAS algorithm and the APF algorithm, given below.

BAS is a relatively novel intelligent optimization algorithm. The design idea is inspired by the foraging behavior of beetles in nature. By optimizing the beetle's foraging process and the concept of pheromone, an optimization algorithm with faster optimization is proposed [46].

APF is a classic obstacle avoidance method. The basic idea is to abstract the motion of the robot in the surrounding environment into the motion in the artificial potential field and guide the robot movement through the force of the potential field. But this method easily falls into local extremum.

The above is a brief introduction to the basic algorithm of the proposed method. The following is a brief introduction and definition of the problem to be solved in this paper. The specific discussion in this paper is the real-time path planning problem in dynamic environment. For this problem, we need to plan a safe path from the start point to the end point as quickly as possible. The next points are taken into consideration for the hybrid BAS-APF path planning method.

- x_{sta} and x_{tar} represents the start and target points of the plan, respectively.
- Express the configuration space in this paper as a set of $Z \subset \mathbb{R}^n, n \in \mathbb{N}$ and $n \geq 2$, where n is the dimension of the configuration space. $Z_{\text{obs}} \subset Z$ is a set of obstacle areas in our configuration

space. $Z_{\text{fre}} \subset Z$ is a set of passable areas in our configuration space that can be obtained according to Formula (1). Another thing to note is that the dynamic obstacles used in this paper, the value of Z will change in real time:

$$Z_{\text{fre}} = Z \setminus Z_{\text{obs}}. \tag{1}$$

- The path obtained by the planning defined as **P** is a set of points $\{p_1, p_2, p_3 ... p_{\max}\}$, where p_i is the i-th path point.
- Path planning can be defined as: setting the function to be optimized $f(\cdot)$ of the sampling point, and let $f(\cdot) \to 0^+$ during the planning process.
- The path optimization process can be defined as: setting the cost function $g(\cdot)$ of the path and obtaining min.$g(\cdot)$ during each optimization process.

In general, we evaluate the path based on some criteria, such as path length, planning time, etc. Therefore, we only need to minimize $g(\cdot)$, while ensuring the planned time t. In addition, this article is mainly focused on two-dimensional environment, but this method is applicable to the case of higher dimensions.

3. Methodology

In this paper, we proposed a novel hybrid BAS-APF method for real-time path planning. This method can be applied to mobile robot path planning in both static and dynamic environments. The entire method flow is shown in Figure 1.

The proposed method is used to generate a safe path from a preset starting point to a target point. Our approach aims to solve the real-time path planning problem of mobile robots in dynamic environments. The map information of the environment needs to be processed after being collected from the sensor. Some of the map files used in this paper are in our own preset simulation map, and the other part is obtained after the real environment is collected and processed by the sensor. Below, we have a detailed introduction to the proposed method.

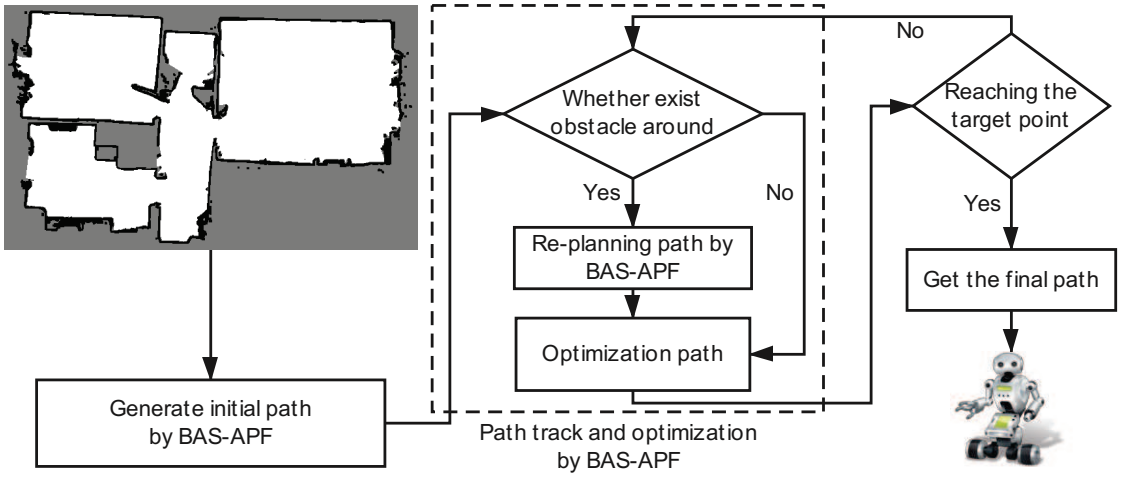


Figure 1. The schematic diagram of the proposed beetle antennae search (BAS)-artificial potential field (APF) method applied to mobile robot path planning in both static and dynamic environments.

3.1. Proposed BAS-APF Method

Path planning in a dynamic environment places high demands on the real-time and security of the planning algorithm. For these two requirements, we first avoid collisions by setting up a layered detector and a well-designed cost function. At the same time, in order to meet the requirements of real-time, the proposed method first gives the initial path, and then gradually optimizes and updates the path during the tracking process of the mobile robot.

The proposed method can be divided into two phases as a whole. The first phase is to generate the initial path, and the second phase is to trace the trajectory of the initial path and optimize the iterative update.

3.1.1. Initial Path Generation

First, according to the pre-setting starting point $\mathbf{x}_{\text{sta}}$, the detection range τ, the direction vector $\mathbf{d}$ and the Formulas (2) and (3), select two candidate points $\mathbf{x}_{\text{l}}, \mathbf{x}_{\text{r}}$ within the perimeter detection range. $\mathbf{x}_{\text{sta}}$ is the initial position of the longicorn in BAS algorithm. $\mathbf{d}$ is the direction that beetle will expore.

$$\mathbf{d} = \frac{\text{rands}\,(k,1)}{\|\text{rands}\,(k,1)\|_2}, \tag{2}$$

where rands$(\cdot)$ is a random function, each time a k-dimensional direction vector $\mathbf{d}$ is randomly selected, and k is the dimension of the space to be planned.

$$\begin{aligned} \mathbf{x}_{\text{l}} &= \mathbf{x} + \mathbf{d}\tau\eta^{i-1}, \\ \mathbf{x}_{\text{r}} &= \mathbf{x} - \mathbf{d}\tau\eta^{i-1}, \end{aligned} \tag{3}$$

where $\mathbf{x}_{\text{l}}$ and $\mathbf{x}_{\text{r}}$ are the coordinates of the candidate points. τ is the size of the detection range, and η is the attenuation rate of the detection range. As the number of iterations increases, the detection range of the agent will gradually shrink, which can also reduce the possibility of falling into local extremum to some extent.

Then, according to Formula (12), the cost function of current point, we calculate $f(\mathbf{x}_{\text{l}})$ and $f(\mathbf{x}_{\text{r}})$ to select a better candidate point, respectively. And according to the Formula (4) and (5), the mobile robot moves one step in the direction of the better point. $\mathbf{x}_{\text{nex}}$ is the position that the robot will move next, and its formula is as follows:

$$\mathbf{x}_{\text{nex}} = \mathbf{x} + s\chi^n \text{sign}(f(\mathbf{x}_{\text{l}}) - f(\mathbf{x}_{\text{r}}))\mathbf{d}, \tag{4}$$

where s represents the current step size, χ represents the step decay rate, and sign$(\cdot)$ is a symbolic function that extracts the sign of a real number.

$$\text{sign}\,(x) = \begin{cases} 1, & x > 0, \\ 0, & x = 0, \\ -1, & x < 0. \end{cases} \tag{5}$$

Add the coordinates to the alternate path after getting $\mathbf{x}_{\text{nex}}$. The candidate path is evaluated according to the evaluation Formula (6). If the evaluation value is better, the current path is replaced; otherwise, the candidate point is re-selected. Iterate through the above process until the target point is reached:

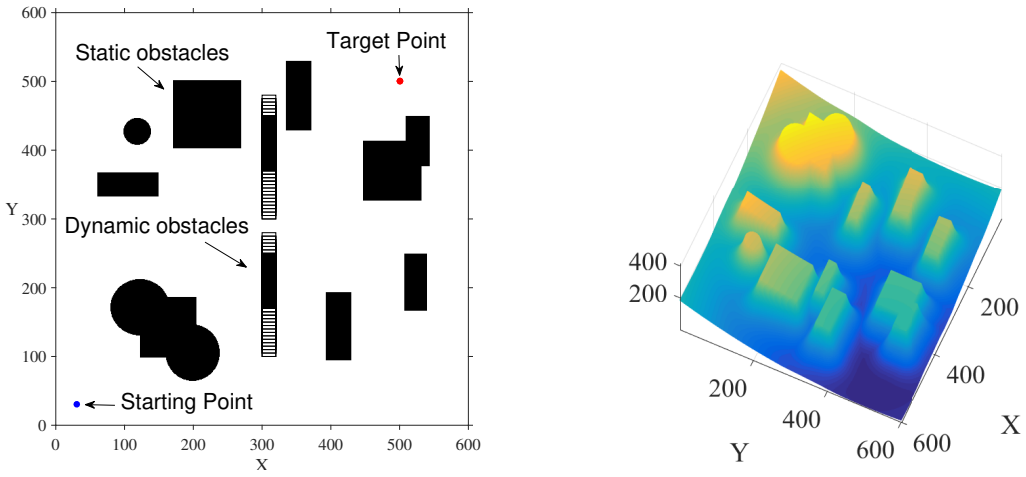
$$g(p) = \sum_{i=1}^{N} (p_i - p_{i-1})^2, \tag{6}$$

where N is the number of path points, and i is the i-th path point. The above steps can already achieve path planning from the start point to the end point, but it may fall into local extremes, resulting in slow planning. Therefore, we speed up planning by designing appropriate cost functions and introducing artificial potential fields. A spline curve is also introduced to fit the path to achieve a smooth path effect.

3.1.2. Add Artificial Potential Field Function

Artificial potential field (APF) is a trajectory planning method based on artificial space field. The basic idea is as follows: First, the robot is regarded as a point in the space to be planned, and the artificial potential field fills the entire space to be planned. The construction method of the artificial

potential field can cause the robot to be attracted to the target point away from the obstacle space. If the potential field is constructed reasonably, the artificial potential field will reach a global minimum at the end point. However, it is difficult to construct such a potential field, and even if such a potential field is constructed, it is not common to all environments. The implementation of the entire artificial potential field can be seen more intuitively through Figure 2.



(**a**) Original map (**b**) Artificial potential field

Figure 2. Original map and artificial potential field (APF) visualization.

Next, we will explain the construction methods of the gravitational field and the repulsive field in the artificial potential field. The gravitational field needs to satisfy the increase in the distance between the current point and the target point, so the easiest way is to set a function that increases linearly with distance. However, this is a defect in the potential field; that is, when the current point is too far from the target point, the gravity is too large. Therefore, an additional quadratic form function is set, and the threshold is set to counteract the effect of the linearly increasing gravitational pull. The specific gravitational field Formula (7) is as follows.

$$U_{\text{att}} = \begin{cases} \frac{1}{2}\epsilon d_g^2, & \text{if} \quad d_g < d_{\text{gra}}, \\ \epsilon d_{\text{gra}} d_g - \frac{1}{2}\epsilon d_{\text{gra}}^2, & \text{if} \quad d_g > d_{\text{gra}}, \end{cases} \tag{7}$$

where ϵ is the gravitational factor that determines the gravitational pull of the current point, and d_g is the distance from the current point to the target point. d_{gra} is the distance threshold of the gravitational field. The calculation of gravitation is to obtain the gradient information of the gravitational field and take the negative gradient. See Formula (8) for specific gravity settings,

$$F_{\text{att}} = \begin{cases} -\epsilon d_g, & \text{if} \quad d_g < d_{\text{gra}}, \\ -\epsilon d_{\text{gra}} + \frac{\epsilon d_{\text{gra}}}{d_g}, & \text{if} \quad d_g > d_{\text{gra}}. \end{cases} \tag{8}$$

Then, we need to build a repulsive field. The repulsive field needs to ensure that the farther away from the obstacle, the smaller the repulsive force, and the distance threshold d_{rep} is set in order to prevent repelling interference from distant obstacles. When the current point distance obstacle is greater than the threshold, the obstacle does not generate a repulsive force to the current point. The specific repulsive field structure is shown in Formula (9).

$$U_{\text{rep}} = \begin{cases} \frac{1}{2}\mu(\frac{1}{d_r} - \frac{1}{d_{\text{rep}}})^2, & \text{if} \quad d_r < d_{\text{rep}}, \\ 0, & \text{if} \quad d_r > d_{\text{rep}}, \end{cases} \tag{9}$$

where μ is the repulsion factor, d_r is the distance from the current point to the obstacle, and d_{rep} is the distance threshold for the repulsion. Similarly, we can get a specific repulsion value by grading the repulsion field. The specific form is as in Formula (10),

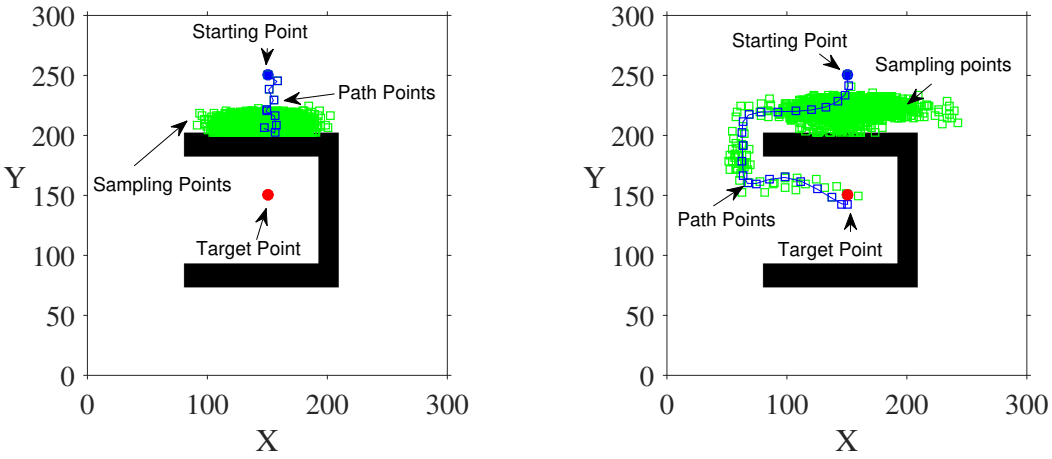
$$F_{\text{rep}} = \begin{cases} \mu(\frac{1}{d_r} - \frac{1}{d_{\text{rep}}})^2 \frac{1}{d_r^2}, & \text{if} \quad d_r < d_{\text{rep}}, \\ 0, & \text{if} \quad d_r > d_{\text{rep}}. \end{cases} \tag{10}$$

Finally, according to the Formula (11), we combined it with attraction and repulsive force to obtain a complete force to guide the robot to the target point.

$$F = F_{\text{att}} + F_{\text{rep}}. \tag{11}$$

3.1.3. Design Cost Function

In the path planning process, due to the scope of the sensor, there may be cases where the agent is trapped at a local minimum and cannot reach the target point. This situation is called a local extremum problem. As shown in Figure 3a, in this example, the robot is stuck at a local minimum and cannot reach the target point. We avoid this problem by designing a suitable cost function. By layering the scope of the robot's detection, we add the corresponding penalty to the cost function according to the location of the obstacle, which helps to escape the local minimum. See Figure 3 for specific effects.



(**a**) Avoiding failure instances (**b**) Avoiding successful instances

Figure 3. Avoid instances of local extremum.

The green squares in Figure 3 represent sampling points, and the blue squares represent path points. Figure 3a is an example of local extremum avoidance when the detection range of the agent is not layered, and the agent is more likely to fall into local extremum. In Figure 3b, we layer the detection range of the agent. Comparing the two sub-figures, it is found that layering the detection range is beneficial to improve the local extremum avoidance ability of the proposed method.

The specific cost function is shown in Formula (12). We divide the detection range of the agent into two layers. When the agent detects obstacles $\mathbb{O}$ in the internal detection range, it adds a penalty factor α to the cost function. When the outer layer of the detection range has an obstacle, the reward coefficient β is added to the cost function. When there is no obstacle within the detection range, the resultant force of the potential field is directly used as the cost function value.

$$f(\mathbf{x}) = \begin{cases} \alpha(\mathbf{x} - \mathbf{x}_{\text{tar}})^2 + F(\mathbf{x}), & \text{if} \quad \mathbb{O} \in D_{\text{in}}, \\ \beta(\mathbf{x} - \mathbf{x}_{\text{tar}})^2 + F(\mathbf{x}), & \text{if} \quad \mathbb{O} \in D, \\ F(\mathbf{x}), & \text{if} \quad \mathbb{O} \notin D, \end{cases} \tag{12}$$

where α, β is the reward coefficient and penalty factor, $F(\mathbf{x})$ is the potential field force of the current point, D_{in} is the inner detection range, and D is the detection range.

3.1.4. Spline Curve

The initial path we get may be too tortuous. Therefore, we have adopted a more common curve fitting method: B-spline curve [47]. The B-spline curve can achieve the effect of modifying the local path without changing the shape of the entire path [48–50]. $N_{i,p}$ is a normalized B-spline basis function defined by the following Cox-deBoor recursive Formula (13),(14).

$$N_{i,0}(u) = \begin{cases} 1, & \text{if} \quad u_{i+1} \leq u < u_{i+1}, \\ 0, & \text{if} \quad ; otherwise, \end{cases} \tag{13}$$

where $N_{i,0}(\cdot)$ is piecewise constant 1 or zero.

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_i} N_{i+1,p-1}(u), \tag{14}$$

where $N_{i,p}(\cdot)$ is the i-th B-spline basis function of degree p. The u_i represents the i-th item in a set of non-decreasing numbers $[u_0, u_1, ..., u_{\max}]$.

In summary, the entire path planning process is now complete, and the pseudo code that generates the initial path can be viewed in Algorithm 1. It is worth noting that we input the map environment information to simulate the overall simulation environment, but the global environment information is not used in the planning process. That is to say, in every step of path planning, the whole environment information is not used; only the environment information around the robot is used. The content of the next subsection is the tracking optimization process for the initial path.

Algorithm 1: BAS-APF design process

Input: $\mathbf{x}_{\text{sta}}$, $\mathbf{x}_{\text{tar}}$, Map;
Output: Path P;
Set some related parameters: step size s, detection range τ, iteration number n, gravitation factor ϵ, repulsion factor μ, convergence threshold d_{thr}, current best objective function value g_{bes};
$\mathbf{x}_n = \mathbf{x}_{\text{sta}}$;
for $i = 1{:}n$ **do**
    Randomly generated **d** according to Formula (2);
    Calculate candidate point coordinates $\mathbf{x}_{\text{l}}$ and $\mathbf{x}_{\text{r}}$ according to Formula (3);
    Calculate cost function value f_{l} and f_{r} of $\mathbf{x}_{\text{l}}$ and $\mathbf{x}_{\text{r}}$ according to Formula (12);
    Compare f_{l} and f_{r} to select a better point to join Candidate Path P_{can};
    Calculate the evaluation value of the P_{can};
    if $g(P_{can}) < g_{bes}$ **then**
        Add $\mathbf{x}_n$ to P;
        Continue;
    else
        Continue;
    if *Distance between* $\mathbf{x}_n$ *and* $\mathbf{x}_{tar} < d_{thr}$ **then**
        $\mathbf{x}_n$ join P;
        break;
return P;

3.2. Real-Time Path Tracking

The whole process of path tracking is roughly as shown in Algorithm 2. First, in order to speed up planning and reduce the amount of computation, the proposed method does not re-plan the path at each step of the tracking but, instead, constructs the detector during the tracking process. The detector has a radius that is twice the step size. Algorithm 1 is called to re-plan the subsequent path only if there is an obstacle in the probe's detection range. At the same time, we also set limits on the number of re-planned algorithm iterations, so although the obtained path may not be globally optimal, it guarantees the real-time performance of the algorithm.

Algorithm 2: Real-time path tracking

Input: Initial path, Map;
Output: Final path;
Set the detection range, get the number of path points in the initial path pn;
for $pn > 2$ **do**
    if *There are obstacles in the* x_{now} *detection range* **then**
        P = Call Algorithm 1;
        $\mathbf{x}_{\text{now}} = P_{\text{nex}}(x, y)$;
    else
        $\mathbf{x}_{\text{now}} = P_{\text{nex}}(x, y)$;
    $pn = pn$-1;
return Final path;

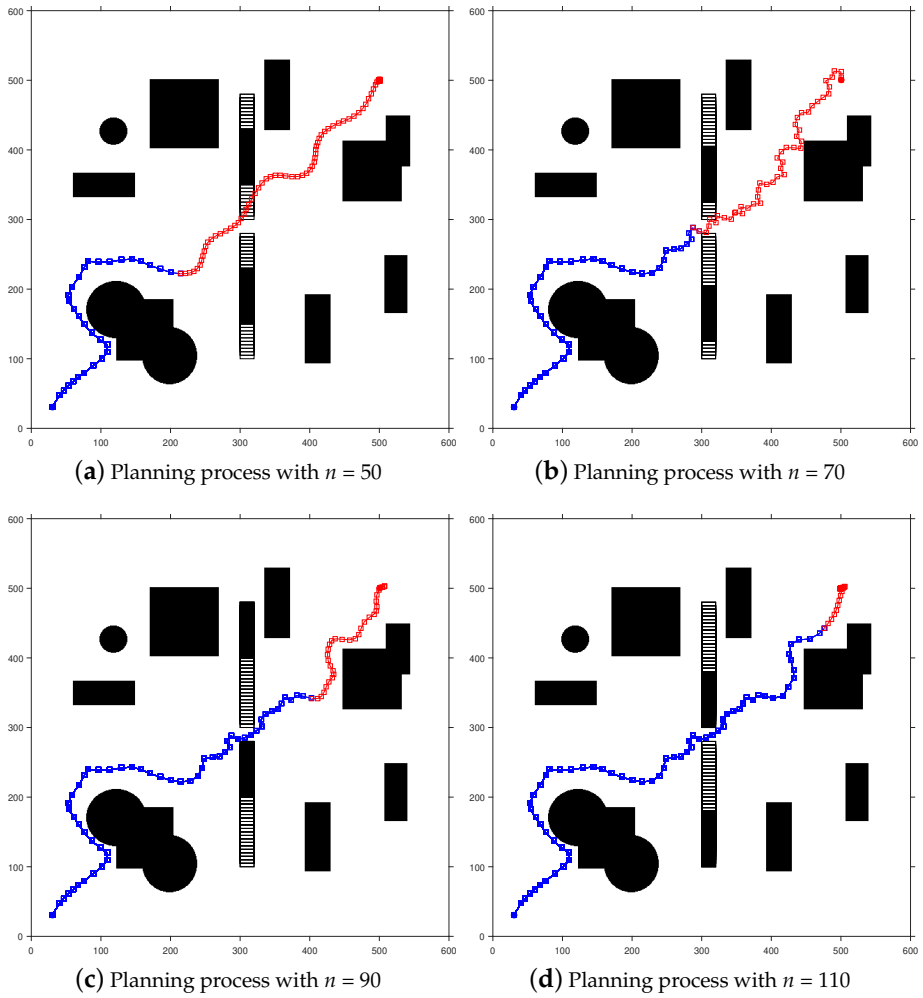
The above is the complete process of the proposed method.

4. Simulations

In order to verify the effectiveness of the proposed method, we performed a series of simulations in MATLAB. In this section, we selected a few representative maps to visualize the results: one is our pre-set simulation map. The other two were obtained by modeling the real room environment. The length of the following maps is in centimeters. At the same time, comparison simulations with several other algorithms were also carried out. By comparing the simulative results, the superiority of the algorithm is verified. In addition, we also made a visual process diagram of the pathfinding process of the BAS-APF method, which can visually see the entire path optimization process.

4.1. Simulation Map Results Using Virtual Map

This section mainly shows the simulation results of the proposed method on the simulated map and the visualization of the planning process. In addition, a comparison of the selection criteria of the comparison algorithm and the results of the single path planning is also introduced. The preset simulation map (Map 1) resolution is 600 × 600, and the map contains multiple static obstacles and two regularly moving dynamic obstacles. Taking Figure 4 as an example, in an environment of 600 cm × 600 cm, dynamic obstacles move 5 cm in each step of the robot, with 130 steps in total, i.e., 650 cm in total, with a total planning time of 0.287 s. The obstacles move with the velocity being 22.65 m/s, and the white hollow square is the trajectory of the obstacles.



(**a**) Planning process with $n = 50$ (**b**) Planning process with $n = 70$

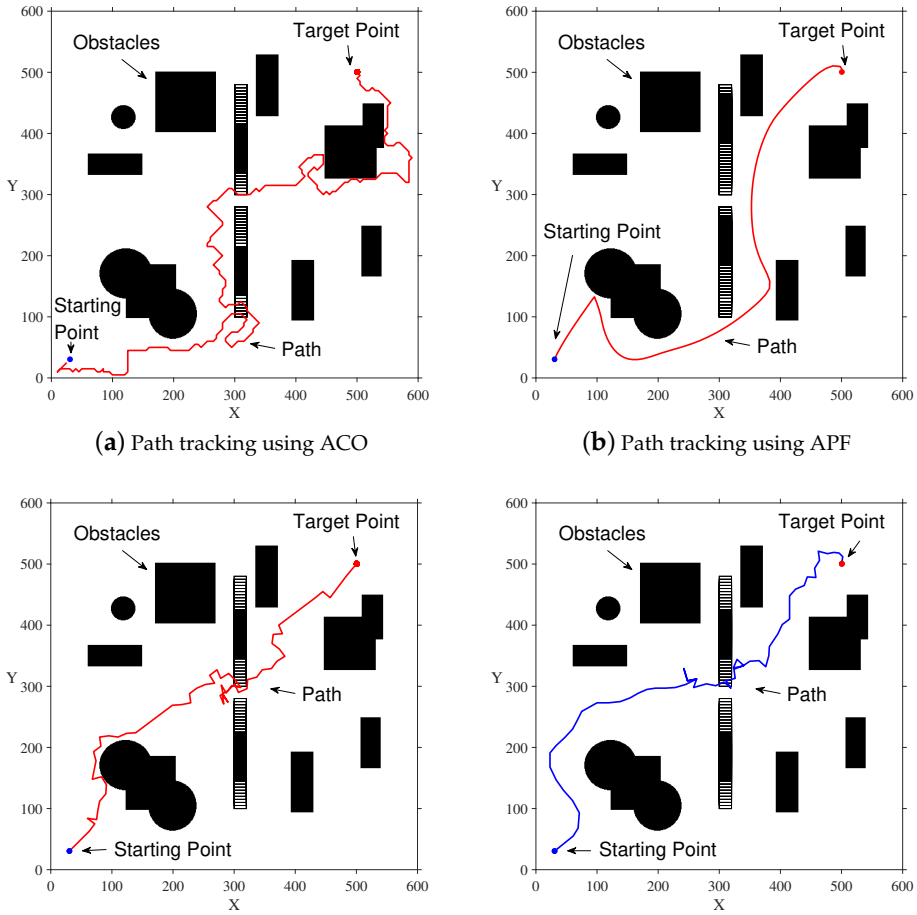
(**c**) Planning process with $n = 90$ (**d**) Planning process with $n = 110$

Figure 4. The process of exploring a path on a simulative map with dynamic obstacles.

A visual representation of the proposed method on Map 1 is shown in Figure 4, where the blue open squares indicate the determined path points, and the red open squares indicate the path points to be determined. The black area represents the obstacle, and the white hollow square represents the movement trajectory of the dynamic obstacle. Each square represents one step of the robot's movement, and n represents the number of iteration steps.

Regarding the choice of comparison algorithm, we make a decision based on the following considerations. Firstly, because our proposed method refers to the idea of APF algorithm, in order to highlight the superiority of hybrid algorithm, we choose APF as one of the comparison algorithms. In addition, since the BAS algorithm is an intelligent optimization algorithm, we chose a classic intelligent optimization algorithm: ACO algorithm as a comparison algorithm. Based on the comprehensive consideration of the comparison algorithm, we also selected a sampling-based algorithm: RRT. The superiority of the BAS-APF method is verified by comparison with multiple different types of comparison algorithms.

In Figure 5, we show the results of four different contrast algorithms on a simulated map. It is worth mentioning that the number of iterations of ACO is 1000, and the number of ants is 50. The blue dot represents the starting point and the red dot represents the ending point. By comparing the four subgraphs, we find that the final path obtained by our proposed algorithm is smoother and shorter, and the final path is not too close to the obstacle, and the safety factor is higher.



(**a**) Path tracking using ACO

(**b**) Path tracking using APF

(**c**) Path tracking using rapidly-exploring random trees (RRT)

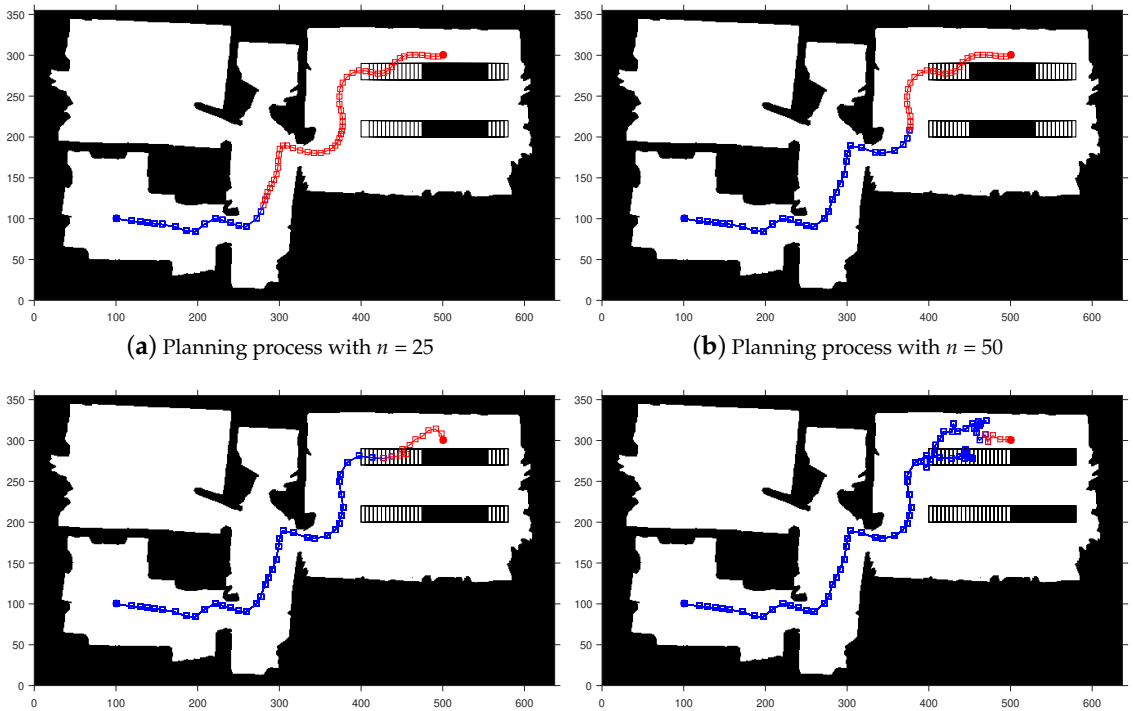
(**d**) Path tracking using beetle antennae search (BAS)-APF

Figure 5. Simulation results of simulation map.

4.2. Simulated Results Using Real Map

Considering the practical applicability of the algorithm, in this section, we have selected two processed actual maps for simulation. Since this paper mainly verifies the obstacle avoidance ability of the proposed method in the dynamic environment, we added two dynamic obstacles that move regularly along the horizontal direction in the first real map (Map 2). And the size of Map 2 is 637 × 355. Similarly, the second real map (Map 3) adds two dynamic obstacles that move regularly in the vertical direction. The specification of Map 3 is 597 × 375.

Figure 6 shows the planning process of our algorithm on Map 2. At the same time, in order to better reflect the entire tracking process, we select the process screenshot of the re-planning. It can be seen that, since in Figure 6a,b, there are no obstacles in the surrounding area. In order to improve the tracking speed, no re-planning is performed. In Figure 6a,d, there are dynamic obstacles in the peripheral area where collisions may occur. The algorithm re-plans and successfully avoids obstacles in reaching the end point.



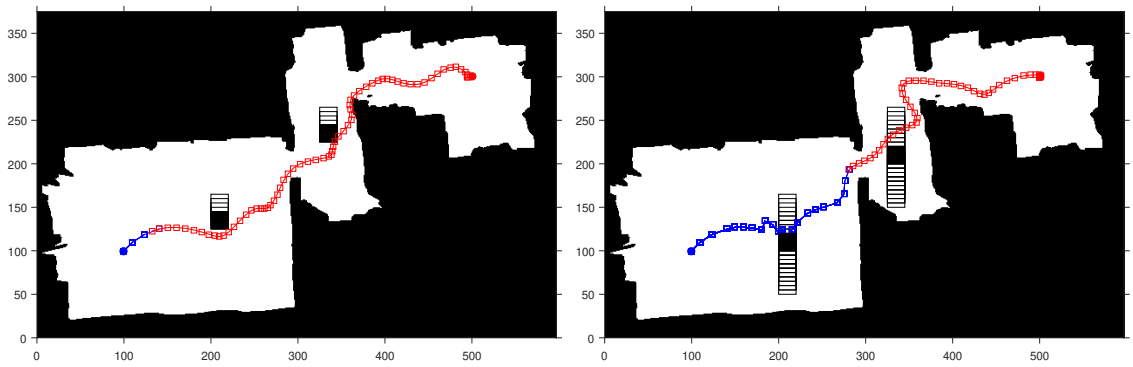
(**a**) Planning process with n = 25 (**b**) Planning process with n = 50

(**c**) Planning process with n = 75 (**d**) Planning process with n = 100

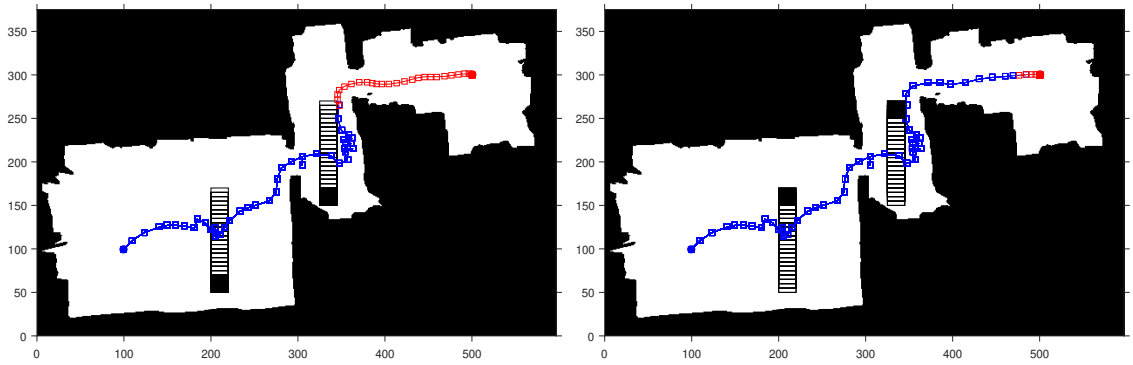
Figure 6. The process of exploring a path on the map with horizontally dynamic obstacles.

Similarly, Figure 7 shows the tracking process on Map 3 in its entirety. It can be seen from the two figures that the algorithm has good adaptability in different environments and can plan a safe path, while dynamically avoiding obstacles.

In order to more intuitively reflect the superiority of BAS-APF algorithm, we also conducted a comparison simulation on Map 2 and Map 3. Figure 8 shows the tracking results of the BAS-APF and comparison algorithm in Map 2. As can be seen from the figure, the path obtained by ACO algorithm is too close to the obstacle, and the path obtained by APF is smooth, but there are too many useless turns. The path obtained by RRT algorithm is too tortuous, and there is also a case where a useless return path is taken. The path obtained by BAS-APF takes into account the balance between path length and path smoothness.

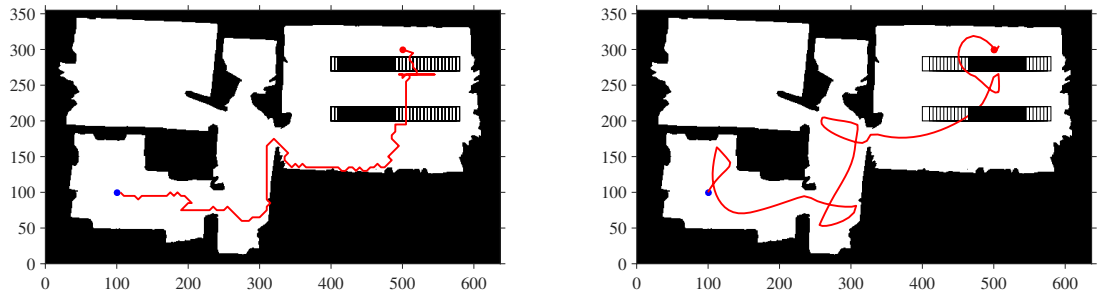


(**a**) Planning process with $n = 5$ (**b**) Planning process with $n = 30$

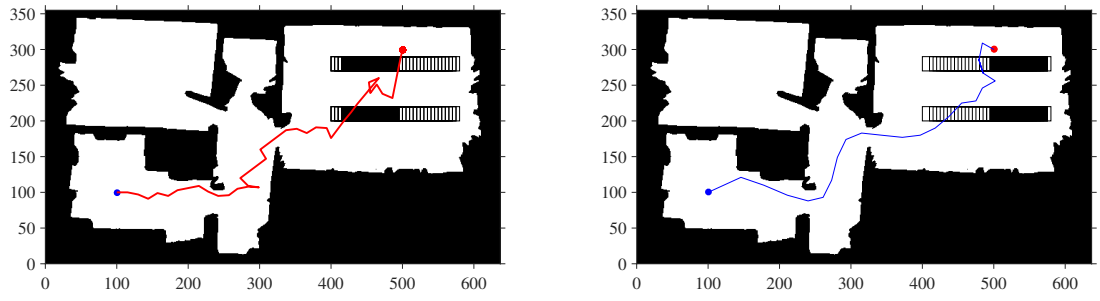


(**c**) Planning process with $n = 55$ (**d**) Planning process with $n = 80$

Figure 7. The process of exploring a path on a map with vertical dynamic obstacles.



(**a**) Path tracking using ACO (**b**) Path tracking using APF

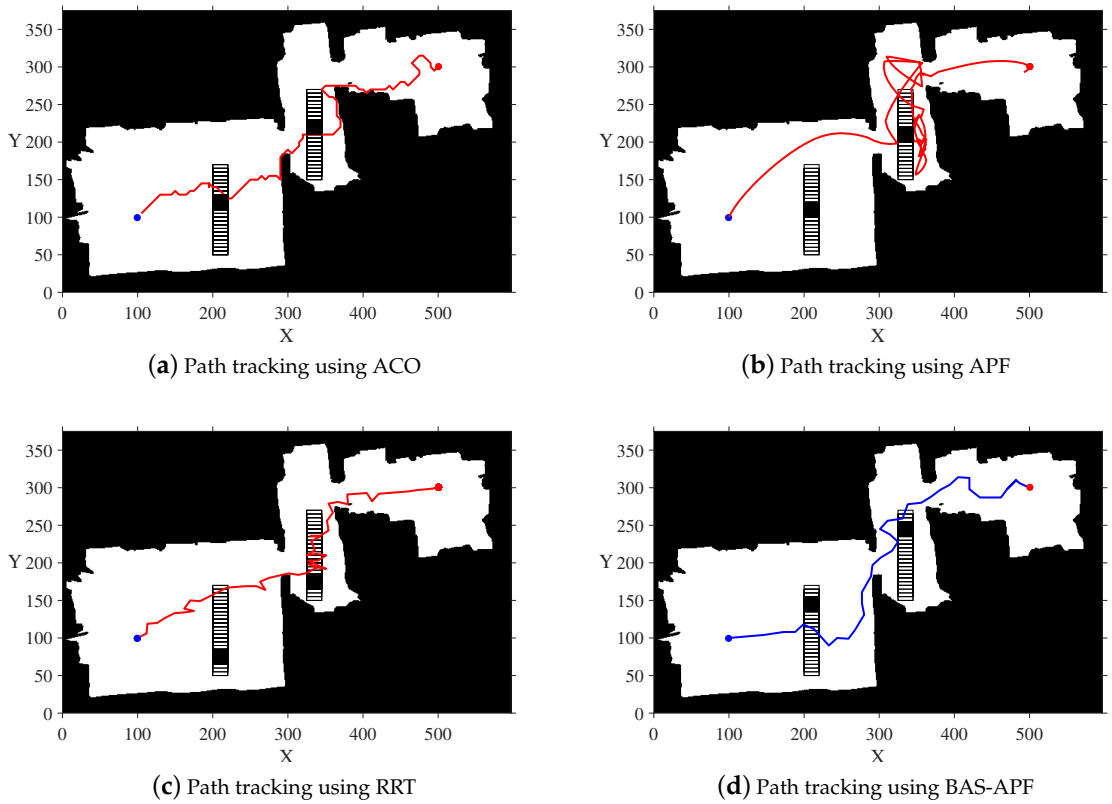


(**c**) Path tracking using RRT (**d**) Path tracking using BAS-APF

Figure 8. Comparative simulation results on a map with horizontal dynamic obstacles.

Similarly, in Figure 9, we show the results of planning for multiple planning algorithms under a map with vertically moving dynamic obstacles. In Figure 9a, the path planned by ACO is almost

attached to the obstacle, which lacks practical application value. In Figure 9c, it can be seen that the path obtained by APF is cluttered. The path obtained by the RRT shown in sub-figure rrtMap 3 is found to have a large angle occasionally, which will affect the tracking speed of the robot during the actual tracking process. The final path obtained by our proposed algorithm, shown in Figure 9d, maintains a certain distance from the obstacle, while maintaining a certain degree of smoothness.



(**a**) Path tracking using ACO

(**b**) Path tracking using APF

(**c**) Path tracking using RRT

(**d**) Path tracking using BAS-APF

Figure 9. Comparative simulation results on a map with vertical dynamic obstacles.

4.3. Comparisons with Other Algorithms

This section mainly shows the comparison of the results of our method and comparison algorithms in batch simulations. The batch simulations in this paper refers to the average of 200 simulative results. We have shown the single-planning results of the comparison algorithm in Sections 4.1 and 4.2. In order to make a more rigorous and scientific comparison of the proposed method and other comparison algorithms, we conducted batch simulations on each of the three maps and summarized the numerical results. Like most researchers, we chose path length and planning time as a measure of the superiority of the algorithm. Table 1 expresses the comparison results of the path lengths of the various algorithms. By comparison, it is found that, although the advantage is not particularly obvious, the proposed method has the shortest path on all maps. Table 2 exhibits the comparison results of planning time for all algorithms. And the BAS-APF method has obvious advantages in planning time.

Table 1. Summary of comparison results of various algorithm path length.

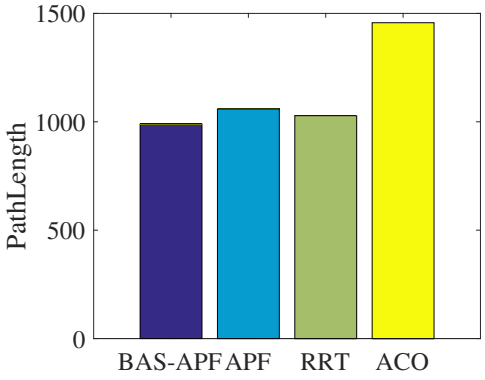
Algorithm / Map	BAS-APF	APF	RRT	ACO
Map 1	**989.8**	1060.4	1028.6	1457
Map 2	**762.2**	1104.2	938.8	842.9
Map 3	**666.5**	1203	670.9	801.3

Table 2. Summary of comparison results of various algorithm planning time.

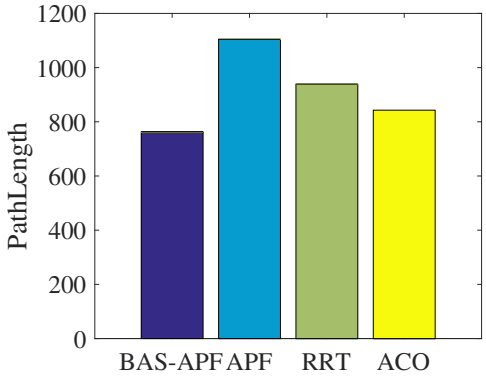
Map \ Algorithm	BAS-APF	APF	RRT	ACO
Map 1	**0.287**	1.75	1.9	8.27
Map 2	**0.276**	0.871	3.31	11.24
Map 3	**0.244**	0.804	1.79	5.14

From the above two tables, we can see that BAS-APF method has certain advantages in terms of path length and planning time, and the advantage in planning time is significant. In order to more intuitively demonstrate the advantages of the proposed method, we visualized the results.

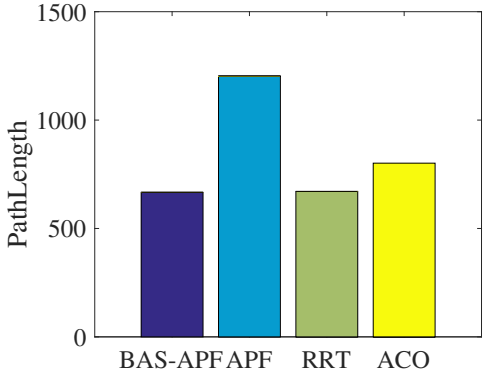
The results of the comparison are shown in detail in Figures 10 and 11. Figure 10 is a comparison result of path lengths on respective maps. It can be seen that the performance of other algorithms is not stable, and the proposed method is more stable and optimal. Figure 11 displays the comparison of the planning time of each algorithm on the simulative map. The comparison of the three sub-figures in Figure 11 illustrates that our proposed method has significant advantages in planning time.



(**a**) Comparison of path length in Map 1

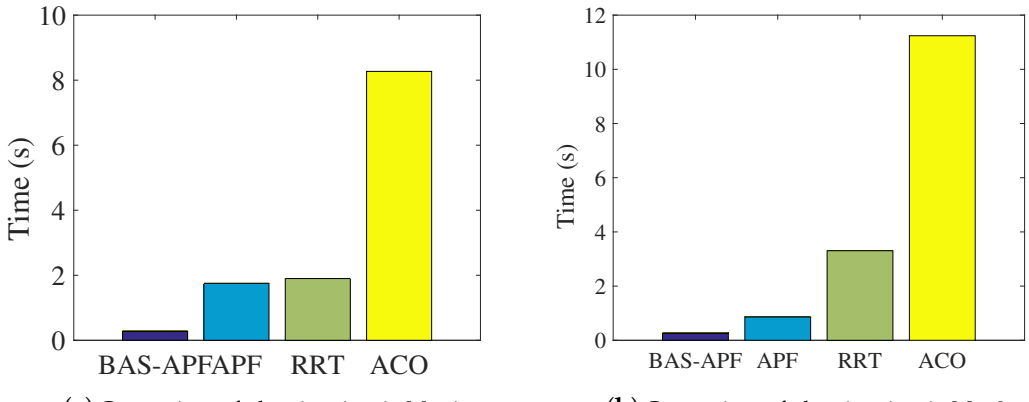


(**b**) Comparison of path length in Map 2



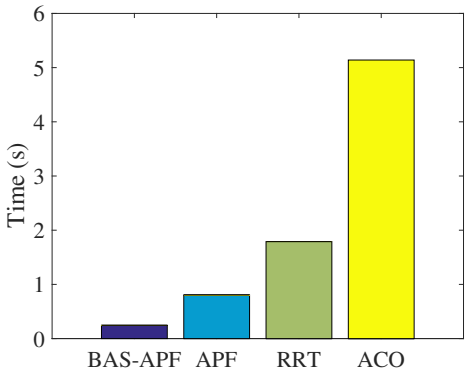
(**c**) Comparison of path length in Map 3

Figure 10. Comparison of path lengths on simulative maps.



(**a**) Comparison of planning time in Map1

(**b**) Comparison of planning time in Map2



(**c**) Comparison of planning time in Map3

Figure 11. Comparison of planning time on simulative maps.

5. Conclusion and Future Work

This paper is concerned with the path planning problem of mobile robots in dynamic environments. We proposed a new two-stage hybrid method called BAS-APF to solve this problem. It could find a safe and feasible path, while ensuring real-time performance in a dynamic environment. Based on the real-time requirements of dynamic programming, the algorithm weighs path optimization and planning time, and it achieved good results in both aspects. In addition, we verified the effectiveness and robustness of the BAS-APF method through simulations. Moreover, the superiority of the proposed method was further verified by comparing it with several different types of classical path planning algorithms. The experiment test with real robots in real environments is not considered at the current state of this work. On the one hand, this is because the conditions of our laboratory are really limited, and the price of mobile robots is expensive, which we cannot afford, for the time being. On the other hand, the focus of this paper was mainly the research of algorithm, not the construction of the physical platform of mobile robot.

In the future, we intend to continue work on the following two aspects: One is that we would like to use real environment and real mobile robots for experiment tests in future work and realize the optimization on the accuracy of obstacle avoidance, so as to improve our algorithm and reflect the advantages of the proposed algorithm in the real environment. The other is to extend it to multi-objective planning so that it can simultaneously plan multiple mobile robots.

Author Contributions: Conceptualization, Q.W., Z.C., L.W. and H.L.; methodology, Q.W., Z.C., L.W. and H.L.; software, Q.W. and D.C.; validation, Q.W. and S.L.; formal analysis, Q.W., Z.C., L.W. and H.L.; investigation, Q.W., Z.C., L.W. and H.L.; resources, Q.W., Z.C., L.W. and H.L.; data curation, Q.W. and S.L.; writing—original draft preparation, Q.W.; writing—review and editing, Q.W., Z.C., L.W., H.L. and Z.J.; visualization, Q.W., Z.C., L.W. and

H.L.; supervision, Q.W., Z.C., L.W. and H.L.; project administration, Q.W., Z.C., L.W. and H.L.; funding acquisition, Z.C., L.W. and H.L. and D.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China (with numbers 61906054 61401385 and 61702146), by Hong Kong Research Grants Council Early Career Scheme (with number 25214015), by Departmental General Research Fund of Hong Kong Polytechnic University (with number G.61.37.UA7L), and also by PolyU Central Research Grant (with number G-YBMU).

Acknowledgments: In this section you can acknowledge any support given which is not covered by the author contribution or funding sections. This may include administrative and technical support, or donations in kind (e.g., materials used for experiments).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chen, D.; Li, S.; Lin, F.; Wu, Q. New Super-Twisting Zeroing Neural-Dynamics Model for Tracking Control of Parallel Robots: A Finite-Time and Robust Solution. *IEEE Trans. Cybern.* **2019**. [CrossRef]
2. Jin, L.; Li, S. Distributed task allocation of multiple robots: A control perspective. *IEEE Trans. Syst. Man Cybern. Syst.* **2018**, *48*, 693–701. [CrossRef]
3. Chen, D.; Zhang, Y.; Li, S. Tracking Control of Robot Manipulators with Unknown Models: A Jacobian-Matrix-Adaption Method. *IEEE Trans. Ind. Inform.* **2018**, *14*, 3044–3053. [CrossRef]
4. Sariff, N.; Buniyamin, N. An Overview of Autonomous Mobile Robot Path Planning Algorithms. In Proceedings of the 2006 4th Student Conference on Research and Development, Selangor, Malaysia, 27–28 June 2006; pp. 183–188.
5. Montiel, O.; Orozco-Rosas, U.; Sepúlveda, R. Path planning for mobile robots using Bacterial Potential Field for avoiding static and dynamic obstacles. *Expert Syst. Appl.* **2015**, *42*, 5177–5191. [CrossRef]
6. Chen, D.; Li, S.; Li, W.; Wu, Q. A Multi-Level Simultaneous Minimization Scheme Applied to Jerk Bounded Redundant Robot Manipulators. *IEEE Trans. Autom. Sci. Eng.* **2019**. [CrossRef]
7. Wu, Q.; Shen, X.; Jin, Y. Intelligent beetle antennae search for UAV sensing and avoidance of obstacles. *Sensors* **2019**, *19*, 1758. [CrossRef] [PubMed]
8. Roberge, V.; Tarbouchi, M.; Labonté, G. Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning. *IEEE Trans. Ind. Inform.* **2013**, *9*, 132–141. [CrossRef]
9. Chen, Y.; Luo, G.; Mei, Y. UAV path planning using artificial potential field method updated by optimal control theory. *Int. J. Syst. Sci.* **2016**, *47*, 1407–1420. [CrossRef]
10. Chen, D.; Li, S.; Wu, Q. Rejecting Chaotic Disturbances Using a Super-Exponential-Zeroing Neurodynamic Approach for Synchronization of Chaotic Sensor Systems. *Sensors* **2019**, *19*, 74. [CrossRef]
11. Li, S.; He, J.; Li, Y.; Rafique, M.U. Distributed recurrent neural networks for cooperative control of manipulators: A game-theoretic perspective. *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, *28*, 415–426. [CrossRef]
12. Wu, Q.; Lin, H.; Jin, Y. A new fallback beetle antennae search algorithm for path planning of mobile robots with collision-free capability. *Soft Comput.* **2019**. [CrossRef]
13. Chen, D.; Li, S.; Wu, Q.; Luo, X. Super-twisting ZNN for coordinated motion control of multiple robot manipulators with external disturbances suppression. *Neurocomputing* **2019**. [CrossRef]
14. Vourchteang, S.; Sugawara, T. Area partitioning method with learning of dirty areas and obstacles in environments for cooperative sweeping robots. In Proceedings of the 2015 IIAI 4th International Congress on Advanced Applied Informatics, Okayama, Japan, 12–16 July 2015; pp. 523–529.
15. Zhang, G.Q.; Li, L.M.; Choi, S. Robotic automated battery sorting system. In Proceedings of the 2012 IEEE International Conference on Technologies for Practical Robot Applications (TePRA), Woburn, MA, USA, 23–24 April 2012; pp. 117–120.
16. Aravind, K.R.; Raja, P.; Pérez, R.M. Task-based agricultural mobile robots in arable farming: A review. *Span. J. Agric. Res.* **2017**, *15*, 1–16. [CrossRef]
17. Tsardoulias, E.G.; Iliakopoulou, A. A review of global path planning methods for occupancy grid maps regardless of obstacle density. *J. Intell. Robot. Syst.* **2016**, *84*, 829–858. [CrossRef]
18. Jin, L.; Zhang, Y.; Li, S.; Zhang, Y. Modified ZNN for time-varying quadratic programming with inherent tolerance to noises and its application to kinematic redundancy resolution of robot manipulators. *IEEE Trans. Ind. Electron.* **2016**, *63*, 6978–6988. [CrossRef]

19. Chen, D.; Li, S.; Wu, Q.; Luo, X. New Disturbance Rejection Constraint for Redundant Robot Manipulators: An Optimization Perspective. *IEEE Trans. Ind. Inform.* **2019**. [CrossRef]
20. Chen, D.; Li, S. A recurrent neural network applied to optimal motion control of mobile robots with physical constraints. *Appl. Soft Comput.* **2019**. [CrossRef]
21. Jauwairia, N.; Malik, F.I.U.; Yasar, A.; Osman, H.; Mushtaq, K.; Muhammad, M.S. RRT*-SMART: A Rapid Convergence Implementation of RRT*. *Int. J. Adv. Robot. Syst.* **2013**. [CrossRef]
22. Noreen, I.; Khan, A.; Asghar, K.; Habib, Z. A Path-Planning Performance Comparison of RRT*-AB with MEA* in a 2-Dimensional Environment. *Symmetry* **2019**, *11*, 945. [CrossRef]
23. Iram, N.; Amna, K.; Zulfiqar, H. A Comparison of RRT, RRT* and RRT*-Smart Path Planning Algorithms. *IJCSNS Int. J. Comput. Sci. Netw. Secur.* **2016**, *16*, 20.
24. Raja, P.; Pugazhenthi, S. Optimal path planning of mobile robots: A review. *Int. J. Phys. Sci.* **2012**, *7*, 1314–1320. [CrossRef]
25. Li, S.; Zhang, Y.; Jin, L. Kinematic control of redundant manipulators using neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, *28*, 2243–2254. [CrossRef]
26. Mac, T.T.; Copot, C.; Tran, D.T.; De Keyser, R. Heuristic approaches in robot path planning: A survey. *Robot. Auton. Syst.* **2016**, *86*, 13–28. [CrossRef]
27. Chen, D.; Zhang, Y. A hybrid multi-objective scheme applied to redundant robot manipulators. *IEEE Trans. Autom. Sci. Eng.* **2017**, *14*, 1337–1350. [CrossRef]
28. Guo, D.; Xu, F.; Yan, L. New Pseudoinverse-Based Path-Planning Scheme With PID Characteristic for Redundant Robot Manipulators in the Presence of Noise. *IEEE Trans. Control Syst.* **2017**, *26*, 2008–2019. [CrossRef]
29. LaValle, S.M. Rapidly-Exploring Random Trees: A New Tool for Path Planning, 1998. Available online: http://janowiec.cs.iastate.edu/papers/rrt.ps (accessed on 11 November 2019).
30. Warren, C.W. Global path planning using artificial potential fields. In Proceedings of the 1989 International Conference on Robotics and Automation, Scottsdale, AZ, USA, 14–19 May 1989; pp. 316–321.
31. Wang, Z.; Cai, J. Probabilistic roadmap method for path-planning in radioactive environment of nuclear facilities. *Prog. Nucl. Energy* **2018**, *109*, 113–120. [CrossRef]
32. Zammit, C.; Van Kampen, E.J. Comparison between A* and RRT algorithms for UAV path planning. In Proceedings of the 2018 AIAA Guidance, Navigation, and Control Conference, Kissimmee, FL, USA, 8–12 January 2018; p. 1846.
33. Tahir, Z.; Qureshi, A.H.; Ayaz, Y. Potentially guided bidirectionalized RRT* for fast optimal path planning in cluttered environments. *Robot. Auton. Syst.* **2018**, *108*, 13–27. [CrossRef]
34. Ferguson, D.; Stentz, A. Using interpolation to improve path planning: The Field D* algorithm. *J. Field Robot.* **2006**, *23*, 79–101. [CrossRef]
35. Sudhakara, P.; Ganapathy, V. Trajectory planning of a mobile robot using enhanced A-star algorithm. *Indian J. Sci. Technol.* **2016**, *9*, 1–10. [CrossRef]
36. Loong, W.Y.; Long, L.Z.; Hun, L.C. A star path following mobile robot. In Proceedings of the 2011 4th International Conference on Mechatronics (ICOM), Kuala Lumpur, Malaysia, 17–19 May 2011; pp. 1–7.
37. Xin, Y.; Liang, H.; Du, M. An improved A* algorithm for searching infinite neighbourhoods. *Robot* **2014**, *36*, 627–633.
38. Duchoň, F.; Babinec, A.; Kajan, M. Path planning with modified a star algorithm for a mobile robot. *Procedia Eng.* **2014**, *96*, 59–69. [CrossRef]
39. Châari, I.; Koubaa, A.; Bennaceur, H.; Trigui, S.; Al-Shalfan, K. SmartPATH: A hybrid ACO-GA algorithm for robot path planning. In Proceedings of the 2012 IEEE congress on evolutionary computation, Brisbane, QLD, Australia, 10–15 June 2012; pp. 1–8.
40. Chen, D.; Li, S.; Wu, Q.; Liao, L. Simultaneous identification, tracking control and disturbance rejection of uncertain nonlinear dynamics systems: A unified neural approach. *Neurocomputing* **2019**. [CrossRef]
41. Brand, M.; Masuda, M. Ant colony optimization algorithm for robot path planning. In Proceedings of the 2010 International Conference On Computer Design and Applications, Qinhuangdao, China, 25–27 June 2010; Volume 3, p. V3-436.
42. Chen, D.; Zhang, Y. Robust Zeroing Neural-Dynamics and Its Time-Varying Disturbances Suppression Model Applied to Mobile Robot Manipulators. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 4385–4397. [CrossRef] [PubMed]

43. Contreras-Cruz, M.A.; Ayala-Ramirez, V.; Hernandez-Belmonte, U.H. Mobile robot path planning using artificial bee colony and evolutionary programming. *Appl. Soft. Comput.* **2015**, *30*, 319–328. [CrossRef]
44. Xue, Y. Mobile Robot Path Planning with a Non-Dominated Sorting Genetic Algorithm. *Appl. Sci.* **2018**, *8*, 2253. [CrossRef]
45. Wang, H.; Zhou, Z. A Heuristic Elastic Particle Swarm Optimization Algorithm for Robot Path Planning. *Information* **2019**, *10*, 99. [CrossRef]
46. Jiang, X.; Li, S. BAS: Beetle Antennae Search Algorithm for Optimization Problems. *Int. J. Robot. Control* **2018**, *1*, 1–2. [CrossRef]
47. Catmull, E.; Clark, J. Recursively generated B-spline surfaces on arbitrary topological meshes. *Comput.-Aided Des.* **1978**, *10*, 350–355. [CrossRef]
48. Berglund, T.; Brodnik, A.; Jonsson, H.; Staffanson, M.; Soderkvist, I. Planning smooth and obstacle-avoiding B-spline paths for autonomous mining vehicles. *IEEE Trans. Autom. Sci. Eng.* **2010**, *7*, 167–172. [CrossRef]
49. Foo, J.L.; Knutzon, J.; Kalivarapu, V.; Oliver, J.; Winer, E. Path planning of unmanned aerial vehicles using B-splines and particle swarm optimization. *J. Aerosp. Inf. Syst.* **2009**, *6*, 271–290. [CrossRef]
50. Tsai, C.C.; Huang, H.C.; Chan, C.K. Parallel elite genetic algorithm and its application to global path planning for autonomous robot navigation. *IEEE Trans. Ind. Electron.* **2011**, *58*, 4813–4821. [CrossRef]