applied sciences

MDPI

Article

APF-IRRT*: An Improved Informed Rapidly-Exploring Random Trees-Star Algorithm by Introducing Artificial Potential Field Method for Mobile Robot Path Planning

Daohua Wu [1,2], Lisheng Wei [2], Guanling Wang [2], Li Tian [2] and Guangzhen Dai [2,*]

[1] Anhui Key Laboratory of Detection Technology and Energy Saving Devices, School of Electrical Engineering, Anhui Polytechnic University, Wuhu 241000, China
[2] Key Laboratory of Advanced Perception and Intelligent Control of High-End Equipment, Ministry of Education, Wuhu 241000, China
* Correspondence: daigzh@ahpu.edu.cn

Abstract: An Informed RRT* (IRRT*) algorithm is one of the optimized versions of a Rapidly-exploring Random Trees (RRT) algorithm which finds near-optimal solutions faster than RRT and RRT* algorithms by restricting the search area to an ellipsoidal subset of the state space. However, IRRT* algorithm has the disadvantage of randomness of sampling and a non-real time process, which has a negative impact on the convergence rate and search efficiency in path planning applications. In this paper, we report a hybrid algorithm by combining the Artificial Potential Field Method (APF) with an IRRT* algorithm for mobile robot path planning. By introducing the virtual force field of APF into the search tree expansion stage of the IRRT* algorithm, the guidance of the algorithm increases, which greatly improves the convergence rate and search efficiency of the IRRT* algorithm. The proposed algorithm was validated in simulations and proven to be superior to some other RRT-based algorithms in search time and path length. It also was performed in a real robotic platform, which shows that the proposed algorithm can be well executed in real scenarios.

Keywords: path planning; hybrid algorithm; APF-IRRT* algorithm; mobile robots

Citation: Wu, D.; Wei, L.; Wang, G.; Tian, L.; Dai, G. APF-IRRT*: An Improved Informed Rapidly-Exploring Random Trees-Star Algorithm by Introducing Artificial Potential Field Method for Mobile Robot Path Planning. *Appl. Sci.* **2022**, *12*, 10905. https://doi.org/10.3390/app122110905

Academic Editors: Byung-Cheol Min and Jonghoek Kim

Received: 5 September 2022
Accepted: 25 October 2022
Published: 27 October 2022

1. Introduction

The purpose of mobile robot path planning is to find a feasible and collision-free optimal path from a start point to a target point in an obstacle working environment according to certain evaluation criteria, such as the shortest path length, the minimum energy consumption or the shortest walking time [1–3]. As one of the core technologies in mobile robot navigation, path planning ensures that mobile robots can accomplish tasks efficiently, safely and independently, and it has been widely used in motion planning problems, such as in manipulation robots [4], mobile robots [5,6] and unmanned aerial vehicles [7,8]. According to the robot's knowledge about the environment, path planning is divided into two types [9]: local path planning and global path planning. In local path planning, the robots sense the current local environment (i.e., the location and geometry of the obstacles) and construct an estimated map of the environment in real time during the whole search process to find a feasible obstacle-free path from a start point to a target point. However, this category of path planning can easily conclude with a local optimal solution or even a non-reachable target due to a lack of prior information about the environment. Common local path planning algorithms are the Artificial Immune algorithm [10], A-star algorithm [11], Fuzzy algorithm [12] and APF algorithm [13], etc. In global path planning, owing to knowledge of prior information about the search space, the robots can acquire a global optimal path using optimization algorithms directly. Common global path planning algorithms are the Bee Colony algorithm [14], Genetic algorithm [15] and RRT algorithm [16], etc. Due to the disadvantage of their non-real time process, these algorithms

are ineffective if the environment changes, such as an unknown obstacle appearing. Up to now, how to overcome their respective shortcomings and how to realize advantage complementation of different single algorithms remains a technical challenge.

The RRT algorithm, which is considered as one of the most popular global path planning algorithms, is an efficient incremental sampling-based method to find the global solution proposed by S. M. LaValle of Iowa State University in 1998 [16]. The main idea of the RRT algorithm is to construct a search tree rooted at the starting point and to grow the tree by random samples from the search space. The RRT algorithm is simple and can find a solution in any complex environment, and has been widely used in path planning for mobile robots. However, due to the random sampled process, the RRT algorithm usually provides non-optimal solutions. To combat the disadvantages of RRT algorithm, an optimized RRT inversion called RRT star (RRT*) was proposed [17] in 2010. Different from RRT, the RRT* algorithm does not stop sampling after the first solution is found. It continues iteration to optimize the current solution until a near-optimal solution is found. The increased sampling and iteration process also leads to inefficiency of the algorithm.

In 2014, Gammell from the University of Toronto further optimized the RRT* algorithm, which is called IRRT* algorithm, by restricting the search area to an ellipsoidal subset of the state space [18,19]. This method expedited the optimization process and improved the rate of convergence while retaining the same probabilistic completeness and asymptotic optimality as the RRT* algorithm. However, it still has the disadvantage of the RRT-based algorithm, in which it is difficult to avoid unnecessary extension and find the optimal solution due to the randomness of sampling and non-real time process.

Due to the disadvantages mentioned above, many algorithms were proposed to improve the quality of RRT-based algorithms. Kuffner proposed the RRT-Connect algorithm in which two search trees were constructed in the start state and target state, respectively, and grew toward each other [20]. This method greatly reduced the search time and improved the efficiency rate. However, due to the random sampling process, the RRT-Connect method still did not optimize their search tree. Wu et al. proposed Fast-RRT which can obtain a new near-optimal path 20 times faster than the RRT* algorithm [21]. To obtain an optimal path, the authors proposed a scheme in which an Improved RRT algorithm is used to quickly search a viable path and a Fast-Optimal algorithm is used to merge it with current best paths.

Recently, another kind of important method, which combines two different single algorithms, was proposed to improve the search efficiency. Brunner introduced A* algorithm into RRT* algorithm to guide the sampling procedure, which greatly accelerates the convergence rate [22]. However, it is inefficient for A* to find an initial solution in a large-scale scenario. Wang et al. applies a reinforcement learning algorithm to an RRT algorithm to direct the growth of search trees [23]. The algorithm developed the exploration ability of each search tree and improved the search efficiency, but the learning-based methods may not perform well in a new environment. In addition, some other hybrid algorithms were proposed to improve the quality of RRT-based path planning algorithms. Jeong et al. proposed the Q-RRT* algorithm which optimized the process of selecting the parent node and connection by using the triangle inequality [24]. The Q-RRT* algorithm obtained a better initial solution and convergence rate than RRT* algorithm. Wang et al. combined a bio-inspired algorithm and RRT algorithm to improve the search speed of finding the initial solution by modifying the sampling process [25]. Their algorithm greatly promoted the convergence rate and saved on memory usage. However, this method led to nonuniform sample distributions. Mashayekhi introduced a hybrid RRT method by combining the RRT-Connect algorithm and IRRT* algorithm [26]. Their algorithm combined the advantages of the two algorithms, which can find initial solutions as quickly as the RRT-Connect algorithm and return the near-optimal solutions as fast as IRRT* method. A. Qurishi et al. proposed P-RRT* algorithm by introducing the APF method into an RRT* algorithm to guide the exploration direction, which accelerated the convergence speed faster than the RRT* algorithm [27]. However, these algorithms only considered the

presence of static obstacles and can be only used in global path planning. Up to now, there are only a few reports on path planning algorithms that can be used in complex static and dynamic environments. In 2015, Orozco-Rosas et al. proposed a pseudo-bacterial potential field (PBPF) method [28] by introducing the pseudo-bacterial genetic (PBG) algorithm into the traditional APF method to optimize the gain parameters based on evolutionary computation. Compared with the traditional APF method, the PBPF algorithm provides a simple and powerful way to automatically obtain an optimal value of attractive and repulsive potential gains. Another advantage of the PBPF method is that it is suitable to work in both static and dynamic environments, and it does not need to consider the global map information. Recently, Orozco-Rosas et al. improved their method by combining the APF method with membrane computing with the genetic algorithm [29]. These new algorithms greatly improved the research efficiency by shortening the path length and execution time.

In this paper, we propose a new hybrid APF-IRRT* algorithm by combining the APF algorithm and the IRRT* algorithm. The idea of the virtual force field of APF is introduced into the IRRT* algorithm to guide the random tree to grow towards the goal point, which does not need to model the global environment and avoids random sampling all over the state space. The contributions of the proposed algorithm are: (1) It greatly improves the search efficiency, and reduces the search time by more than 35% and 24% compared with the IRRT* and P-RRT* algorithms, respectively. (2) The APF method increases the guidance for the algorithm, which greatly reduces extended nodes and further shortens the path length and the search time. In addition, the modification of the growth direction of the random tree ensures the optimality of the solution of the algorithm. (3) Compared with most other RRT-based algorithms that can only be applied in static scenarios, the proposed hybrid algorithm includes the advantage of the APF method and can be applied both in static and dynamic environments. The combination of these complementary algorithms also provides another idea in solving the path planning for mobile robots.

The paper is structured as follows: Section 2 presents a short background about the IRRT* algorithm. The proposed algorithm, including the related APF algorithm, is described in detail in Section 3. In Section 4, simulation results are presented to show the effectiveness of the proposed algorithm. The conclusion of the work is presented in Section 5.

2. Background

In this section, we will review the background of the proposed algorithm for mobile robot path planning, including the definitions of the work and the main idea of the IRRT* algorithm.

2.1. Problem Definition

We define the path planning algorithm similarly to [18]. Let $C \subseteq \mathbb{R}^n$ be the state space of the system. Path planning is to search for an optimal continuous path from the starting state x_{start} to the target state x_{tar} in the given space C. To ensure accomplishment of task safely, the mobile robot should avoid collision with any obstacles or threats in the region $C_{obs} \subseteq C$, and should be in the set of the permissible state $C_{free} \subseteq C$ throughout navigation. Here, C_{free} and C_{obs} are complementary, and constitute the state space of the mobile robot.

We define the path function $\pi^*[x_{start}, x_{tar}] = (x[\tau])$ to be the optimal path between x_{start} and x_{tar} as:

$$\pi^*[x_{start}, x_{tar}] = \text{argmin}\{\pi \mid x[0] = x_0, x[\tau] = x\}, \quad (1)$$

where x is the robot position at time τ. It has nothing to do with the boundary and obstacles.

We define the cost function $c^*[x_{start}, x_{tar}]$ to be the energy loss of the search time and path length between x_{start} and x_{tar} as:

$$c^*[x_{start}, x_{tar}] = \min\{\pi \mid x[0] = x_0, x[\tau] = x_1\} c[\pi], \quad (2)$$

where $c[\pi]$ is the cost of the path.

For $0 < t < \tau$, $\pi^*[x_0,\ x_1]$ can be expressed as:

$$\pi^*[x_0, x_1] = \pi^*[x_0, x[t]] + \pi^*[x[t], x_1] \tag{3}$$

which indicates that the optimal collision-free trajectory between the starting point and the target point of the robot π^*_{free} can be connected with the best trajectory among a set of discrete subset states $(x_{start}, x_1, x_2, x_3, \ldots, x_{tar})$.

To search for the shortest path length in state space C, the IRRT* algorithm improves the subset state $X_{\hat{f}} \supseteq X_f$ with the current solution c_{best}. The $X_{\hat{f}}$ can be expressed as

$$X_{\hat{f}} = \{x \in X \mid \| x_{start} - x \|_2 + \| x - x_{tar} \|_2 \leq c_{best}\} \tag{4}$$

The subset space can also be described as a special hyperellipsoid whose cross section is shown in Figure 1, in which the x_{start} and x_{tar} are the focal points. The c_{best} is the transverse diameter of the ellipse and the $\sqrt{c^2_{best} - c^2_{min}}$ is the conjugate diameter.

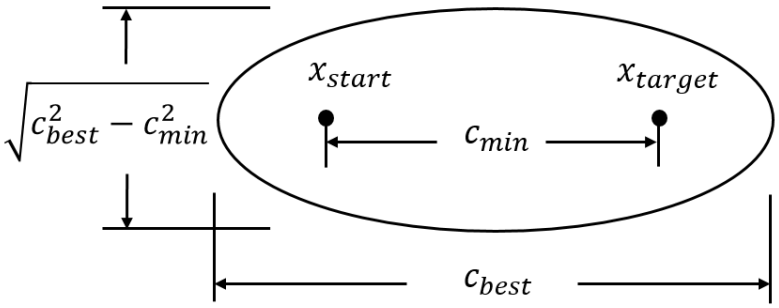


Figure 1. The heuristic sampling ellipse domain. The x_{start} and x_{tar} are the focal points. c_{min} and c_{best} are the theoretical minimum cost and the current solution cost between the start point and target point, respectively.

2.2. Description of IRRT Algorithm*

Algorithm 1 provides the steps of the IRRT* algorithm. It firstly behaves as an RRT* algorithm until a first solution c_{best} is found in line 13, and adds the solution to a list of possible solutions (line 15). Then the IRRT* algorithm uses the minimum solution of this list (line 6) to calculate and continue sampling $X_{\hat{f}}$ (line 7), and repeat the steps as above. Thus, the minimum value of c_{best} of this list can be calculated and updated during iterations. IRRT* algorithm first acts similarly to RRT* algorithm to find an initial solution and calculate the path length. Then it structures an ellipse with the path length c_{best} as its transverse diameter and the start point x_{start} and goal point x_{tar} as the focuses. Finally, it improves the sampling probability of the extended node in the ellipsoidal domain, and decreases the ellipse area by constantly iterating until the optimal solution is found. The iterative process can be shown in Figure 2.

The IRRT* algorithm restricts the search domain to a subset of the initial domain based on the current optimal solution, which improves the rate of convergence. It helps the robot find the near-optimal solution faster than the original RRT* algorithm does. However, due to the disadvantages of the randomness of sampling and non-real time process, the IRRT* algorithm still generates unnecessary extended nodes in invalid regions, which limits the search efficiency and convergence rate.

Algorithm 1: IRRT* (x_{start}, x_{target})

$V \leftarrow \{x_{start}\}$;
$E \leftarrow \varnothing$;
$X_{soln} \leftarrow \varnothing$;
$T = (V, E)$;
for *iteration* $= 1$ *to* n ***do***
  $c_{best} \leftarrow min_{x_{soln} \in X_{soln}} \{Cost(X_{soln})\}$;
  $x_{rand} \leftarrow Sample(x_{start}, x_{target}, c_{best})$;
  $x_{near} \leftarrow Nearnest(T, x_{rand})$;
  $x_{new} \leftarrow Steer(x_{rand}, x_{near}, StepSize)$;
  if $CollisionFree(x_{nearest}, x_{new})$ ***then***
    $X_{near} \leftarrow Nearest(T, x_{new}, r_{RRT^*})$;
      $x_{parent} \leftarrow Parent(x_{near})$;
    if $InEllipseRegion(x_{new})$ ***then***
      $X_{soln} \leftarrow X_{soln} \cup \{x_{new}\}$
      $X_{soln} \cup \{x_{new}\}$;
  $T.addNode(x_{new})$;
  Return T ;

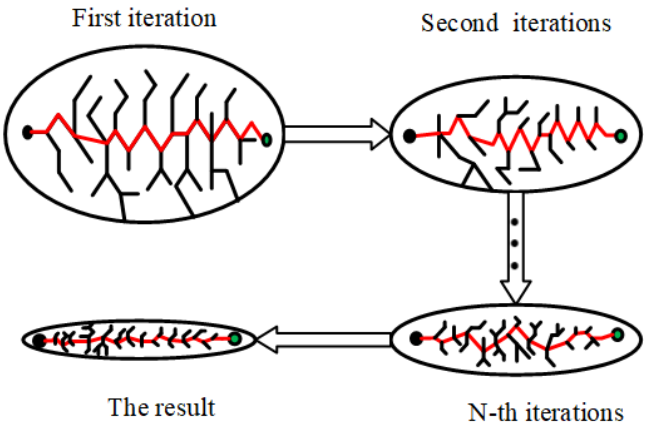


Figure 2. The iterative effect of I-RRT* algorithm.

3. Proposed APF-IRRT* Algorithm

Aiming at improving low search efficiency and the convergence rate caused by unnecessary extended nodes in invalid regions of IRRT* algorithm, we propose a new hybrid method named the APF-IRRT* algorithm. It combines a local APF algorithm and an IRRT* algorithm by introducing the virtual force idea of APF into the IRRT* algorithm at the stage of the search tree expansion. It increases the guidance of the IRRT* algorithm, and improves the search efficiency, while retaining the same probabilistic completeness and asymptotic optimality as IRRT*.

3.1. APF Algorithm

The APF algorithm is a kind of virtual force field method. Its basic idea is to build virtual potential fields in state space C so that the point, which represents the mobile robot, is attracted by the target point x_{tar} , and is repelled by the obstacle x_{obs} in C.

Let $x_t = (x_t^x, x_t^y)$ and $x_{goal} = (x_{tar}^x, x_{tar}^y)$ be the current position and target point, respectively. The attractive field, which guides the robot to the target point, is defined as

$$U_{att}(x) = \frac{1}{2} k_{att} d^2(x_t,\ x_{tar}), \tag{5}$$

where k_{att} and $d(x_t, x_{tar})$ are the attractive coefficient and Euclidean distance between x_t and x_{tar}, respectively. It is a special attractive field which will be stronger when it is further away from the target point. Especially, $U_{att}(x) \to 0$, if $d(x_t, x_{tar}) = 0$.

The attractive force $F_{att}(x)$ can be expressed as the negative gradient of $U_{att}(x)$,

$$F_{att}(x) = -\nabla U_{att}(x) = k_{att} d(x_t, x_{tar}) \tag{6}$$

The direction of $F_{att}(x)$ passes through x_t and x_{tar} to the target point.

Similarly, the repulsive field and repulsive force, which keep the robot away from obstacles, can be expressed as

$$U_{rep}(x) = \begin{cases} \frac{1}{2} k_{rep} \left(\frac{1}{d(x_t, x_{obs})} - \frac{1}{d_0} \right)^2 & if\ d(x_t, x_{obs}) \le d_0 \\ 0, & if\ d(x_t, x_{obs}) > d_0 \end{cases} \tag{7}$$

and

$$F_{rep}(x) = \begin{cases} k_{rep} \frac{1}{d(x_t,\ x_{obs})^2} \left(\frac{1}{d(x_t,\ x_{obs})} - \frac{1}{d_0} \right) & if\ d(x_t, x_{obs}) \le d_0 \\ 0, & if\ d(x_t, x_{obs}) > d_0 \end{cases}, \tag{8}$$

where k_{rep} is the repulsive coefficient, $d(x_t, x_{obs})$ is the Euclidean distance between x_t and x_{obs}, d_0 is the range of influence of obstacle and $U_{rep}(x) \to \infty$ when $d(x_t, x_{obs}) = 0$. The direction of $F_{rep}(x)$ passes through x_t and x_{obs} from the target point.

The total potential $U(x)$ is the sum of an attractive $U_{att}(x)$ and a repulsive potential $U_{rep}(x)$ as

$$U_{total}(x) = U_{att}(x) + U_{rep}(x), \tag{9}$$

whose negative gradient $-\nabla U_{total}(x)$ indicates the most promising local direction of motion.

Having the APF method has the advantages of less calculation, low complexity and good real-time performance; the APF method has been widely used in path planning for mobile robots. However, it easily falls into the target non-reachable and local minima problems due to the lack of prior information about the environment. To this end, the APF method usually combines with other algorithms to overcome these disadvantages. In this article, we propose a hybrid algorithm by combining IRRT* and the APF method by introducing the APF method into the search tree expansion stage of IRRT* algorithm. The APF method greatly improves the guidance for the algorithm, which further shortens the path length and the search time. Furthermore, due to the real-time process of the APF method, the proposed algorithm can be used not only in static environments, but also in dynamic environments.

3.2. Procedure of Hybrid Algorithm

The basic process of the hybrid APF-IRRT* algorithm is described as follows:

Step 1: Establish a grid map based on the workspace of the mobile robot, including the position of obstacles, starting point x_{start} and target point x_{tar}.

Step 2: Set the number and neighborhood radius of iteration. Set step size of I-RRT* algorithm.

Step 3: Using the RRT algorithm to plan the initial path, calculate the path length $\boldsymbol{c_{best}}$ and the Euclidean distance c_{min} between the starting point and the target point.

Step 4: Randomly generate node x_{rand}, find the neighbor node x_{near}, and calculate the vector α from x_{near} to x_{rand}.

Step 5: Initialize the APF parameters, including the attractive coefficient, maximum attractive force, repulsive coefficient and repulsion distance. Calculate the sum vector β of APF.

Step 6: Calculate sum vector θ of α and β, generate a new node x_{new}, and rewrite the parent node for x_{new}.

Step 7: Perform collision detection and elliptical domain detection for the path.

Step 8: Search for a path from the start point to the target point. If found, calculate the path length. If not found, expand the search tree.

Step 9: Determine whether the convergence goal is achieved or not. If it is not achieved, expand the node of the IRRT* until achieved. If it is achieved, end the algorithm.

The chart of the hybrid APF-IRRT* algorithm is shown as Figure 3.

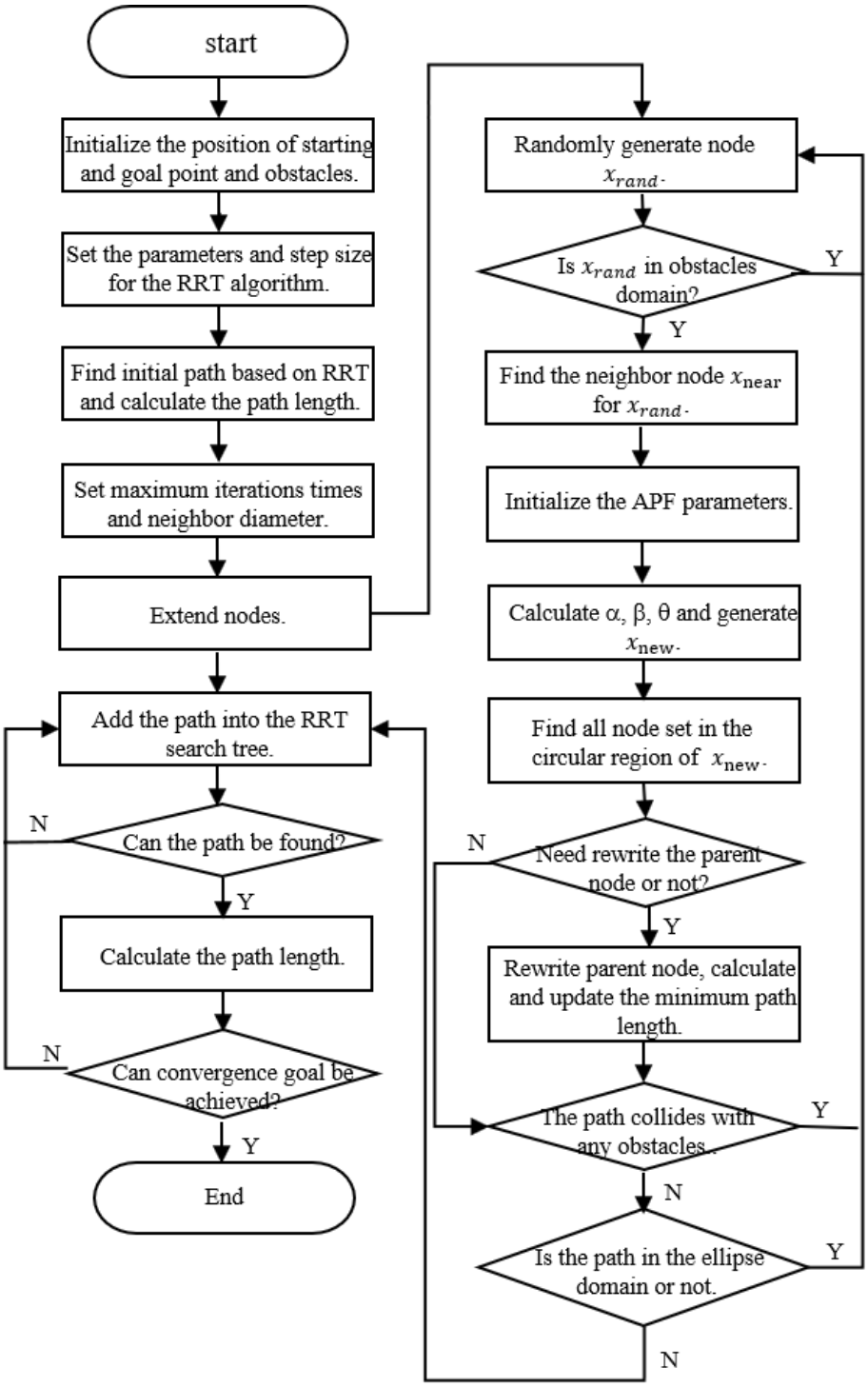


Figure 3. The flow chart of hybrid APF-IRRT* algorithm.

3.3. Description of Hybrid Algorithm

An algorithm example using the proposed hybrid algorithm is presented in Algorithm 2. It is similar to an IRRT* algorithm, which adds the virtual force of APF into an IRRT* algorithm to generate new nodes (line 9). This results in increasing the guidance of the IRRT* algorithm, which will avoid random sampling all over the state space and reduce unnecessary extended nodes in invalid regions. It greatly reduces computation while improving the IRRT* algorithm in real time.

Algorithm 2: $APF-IRRT^{*}(x_{start}, x_{tar})$

1: $V \leftarrow \{x_{start}\}$;
2: $E \leftarrow \varnothing$;
3: $X_{soln} \leftarrow \varnothing$;
4: $T = (V, E)$;
5: ***for*** $iteration = 1$ *to* n ***do***
6: $c_{best} \leftarrow min_{x_{soln} \in X_{soln}} \{Cost(X_{soln})\}$;
7: $x_{rand} \leftarrow Sample(x_{start}, x_{tar}, c_{best})$;
8: $x_{near} \leftarrow Nearnest(T, x_{rand})$;
9: $\boldsymbol{x_{new} \leftarrow Steer(x_{rand}, x_{near}, x_{tar}, map, stepsize)}$;
10: ***if*** $CollisionFree(x_{nearest}, x_{new})$ ***then***
11: $X_{near} \leftarrow Nearest(T, x_{new}, r_{RRT^{*}})$;
12: $x_{parent} \leftarrow Parent(x_{near})$;
13: ***if*** $InEllipseRegion(x_{new})$ ***then***
14: $X_{soln} \leftarrow X_{soln} \cup \{x_{new}\}$ $X_{soln} \cup \{x_{new}\}$;
15: $T.addNode(x_{new})$;
16: ***Return*** T ;

The key steer function algorithm (line 9) can be realized with three steps, as shown in Figure 4. Step one is to calculate the sum vector α of x_{new} and x_{rand} (Figure 4a). Step two uses APF to calculate F_{att} and F_{rep}, and then calculates the sum vector β (Figure 4b). The last step is to calculate the sum vector θ of α and β (Figure 4c).

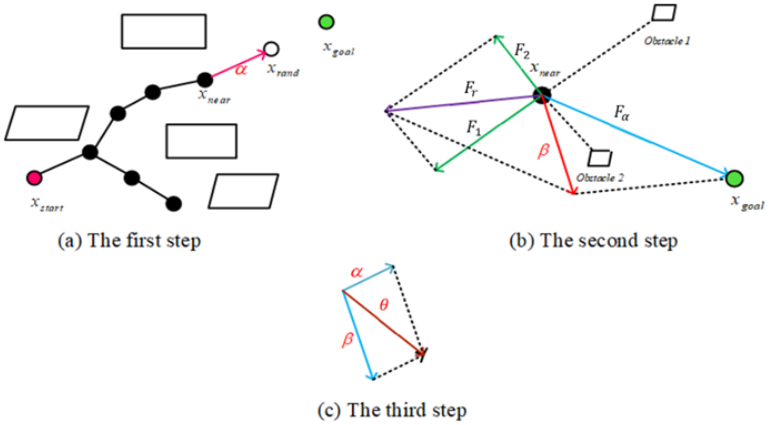


Figure 4. Realization of the steer algorithm in three steps.

The steer function Algorithm 3 is described as:

Algorithm 3: $\boldsymbol{Steer(x_{rand}, x_{near}, x_{tar}, map, stepsize)}$

1 ***if*** $x_{rand} \in C_{free}$ ***then;***
2 $\alpha \leftarrow atan2(x_{near}, x_{rand})$;
3 $angle \leftarrow angle(x_{near}, x_{tar}, map, length(map))$;
4 $Calculation\ \beta()$;
5 $F_a \leftarrow APF\_attact(x_{near}, x_{tar}, angle, AFP\_a, AFP\_amax)$;
6 $F_r \leftarrow APF\_repulsion(x_{near}, map, angle, AFP\_r, AFP\_po)$;
7 $\beta \leftarrow atan2(F_a, F_r)$;
8 $x_{new} \leftarrow Steer(\alpha, \beta, stepsize)$;
9 ***else***
10 $x_{rand} \sim U(X)$;
11 ***Return*** T;

4. Simulation Results and Discussion

In this section, we first perform simulation experiments in six different environments using Matlab to verify the effectiveness of the APF-IRRT* algorithm. Then, a comparison of the proposed algorithm against RRT*, P-RRT* and IRRT* algorithms on a variety of planning problems is presented. Lastly, we perform the proposed APF-IRRT* algorithm in a real laboratory environment to test the path planning on a real mobile robot.

4.1. Comparison with APF Algorithm

To verify the superiority of the algorithm, we first compare the APF-IRRT* with APF algorithm alone. We first construct a grid map as 50 m × 50 m, and set (1, 1) and (49, 49) as start point and target point, respectively. Three obstacles, which are represented by black squares, are set at (15, 20), (32, 16) and (36, 40), respectively. The parameters of APF are set as: attractive coefficient $APF\_a = 20$, the maximum attractive force $APF\_amax = 1$, repulsive coefficient $APF\_r = 15$, and repulsion distance $APF\_po = 20m$. The maximum number of iterations is set to 1000, and the step size is set to 0.4.

Figure 5 shows the simulation results of APF algorithm and proposed algorithm. From 20 independent experiments, we obtained the average path lengths of the APF algorithm as 82.34 m and search time as 10.27 s, and the APF-IRRT* algorithm obtained an average of 70.76 m and 2.36 s. Obviously, compared to the APF algorithm, the APF-IRRT* algorithm has significant advantages in terms of path length and search time.

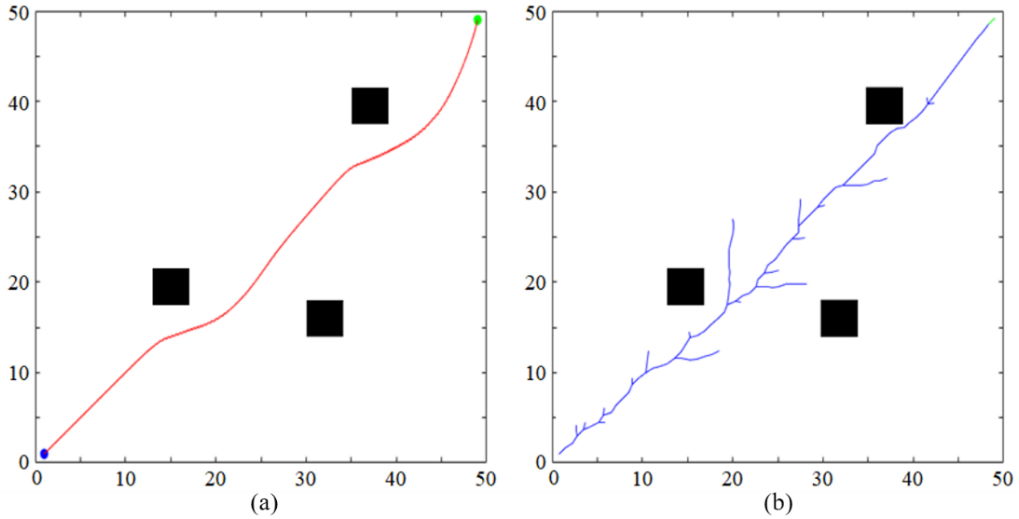


Figure 5. A comparison of the APF algorithm and APF-IRRT* algorithm in a simulated example. (**a**) shows the simulation result of APF algorithm. (**b**) shows the simulation result of APF-IRRT* algorithm.

4.2. Path Planning in Static Scenarios

In this section, the grid map used to model the environmental information is set to 150 m × 250 m. The parameters of APF are set as: attractive coefficient $APF\_a = 200$, the maximum attractive force $APF\_amax = 1$, repulsive coefficient $APF\_r = 150$, and repulsion distance $APF\_po = 50m$. The maximum number of iterations is set to 3000, and the step size of x_{rand} and x_{near} are both set to 5.

Table 1 shows the test environments which contain start position, target position and the obstacles information. Our aim is to find a feasible and collision-free optimal path from a start point to a target point in each obstacles test environment.

Table 1. Test environment of the simulations.

Test Environment	Start Position	Target Position	Obstacles Position
M1	(10, 75)	(240, 75)	(125, 75)
M2	(10, 75)	(240, 75)	(125, 52), (125, 75), (125, 98)
M3	(10, 75)	(240, 75)	(100, 52), (160, 75), (100, 98)
M4	(10, 10)	(240, 140)	(100, 98), (100, 75), (120, 45)
M5	(10, 10)	(240, 140)	(60, 60), (60, 35), (100, 98), (100, 75), (120, 45)
M6	(80, 85)	(80, 200)	(55, 104), (55, 92), (55, 80), (55, 68), (72, 68), (89, 68) (89, 104), (72, 104), (123, 68), (123, 80), (123, 92), (123, 104)

Figure 6 presents the best path which has the shortest collision-free path length based on the proposed algorithm in each test environment described in Table 1. To understand the advantage of the proposed hybrid algorithm directly, the path lengths of the RRT*, IRRT*, A*-RRT*, P-RRT* and APF-IRRT* algorithms for the above six test environments are shown in Figure 7. The simulation results in Figure 7 shows that the proposed APF-IRRT* algorithm provides better solutions than those obtained using RRT* and IRRT*, A*-RRT*, and P-RRT* algorithms, especially in more complex environments. For example, in the test environment M5, the RRT* algorithm obtained an average path length of 305.47 m, the IRRT* algorithm obtained 295.47 m, the A*-RRT* algorithm obtained 289.74 m and the P-RRT* algorithm obtained 283.24 m from 20 independent experiments, the APF-IRRT* algorithm obtained an average of 268.01 m. This gives a difference of 37.47 m, 27.46 m, 21.50 m and 15.23 m between the average path length that was obtained by the RRT*, IRRT*, A*-RRT* and P-RRT* algorithms, respectively.

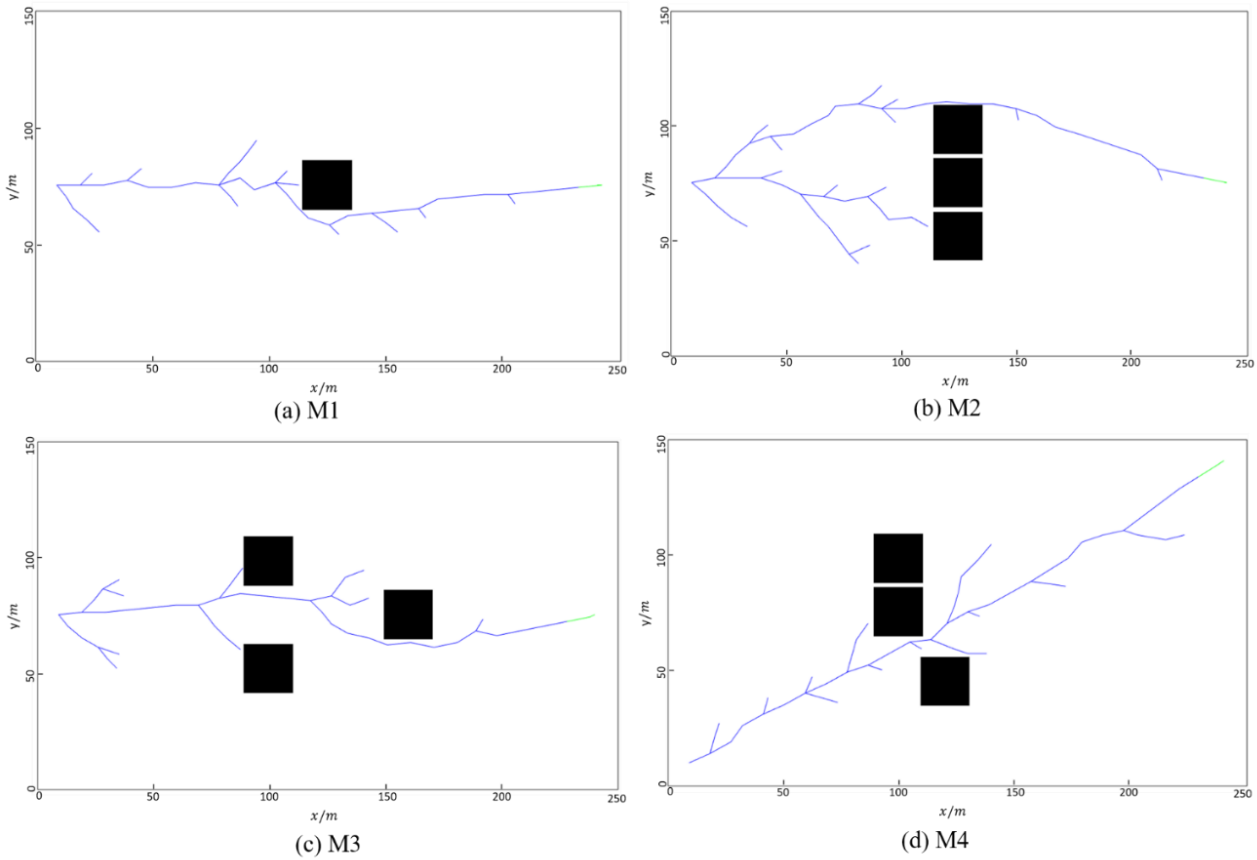


Figure 6. *Cont.*

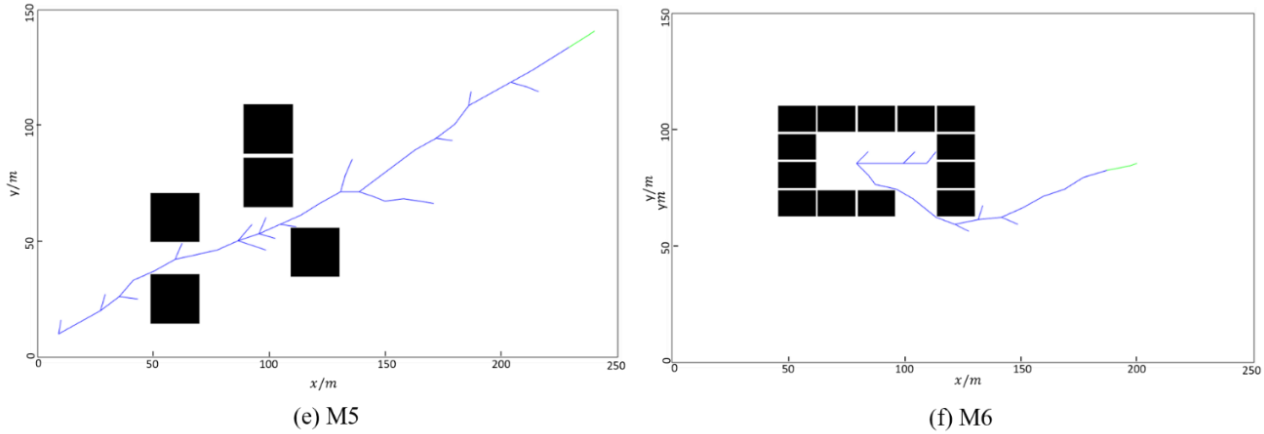


Figure 6. The best path planning of simulation results for different test environments.

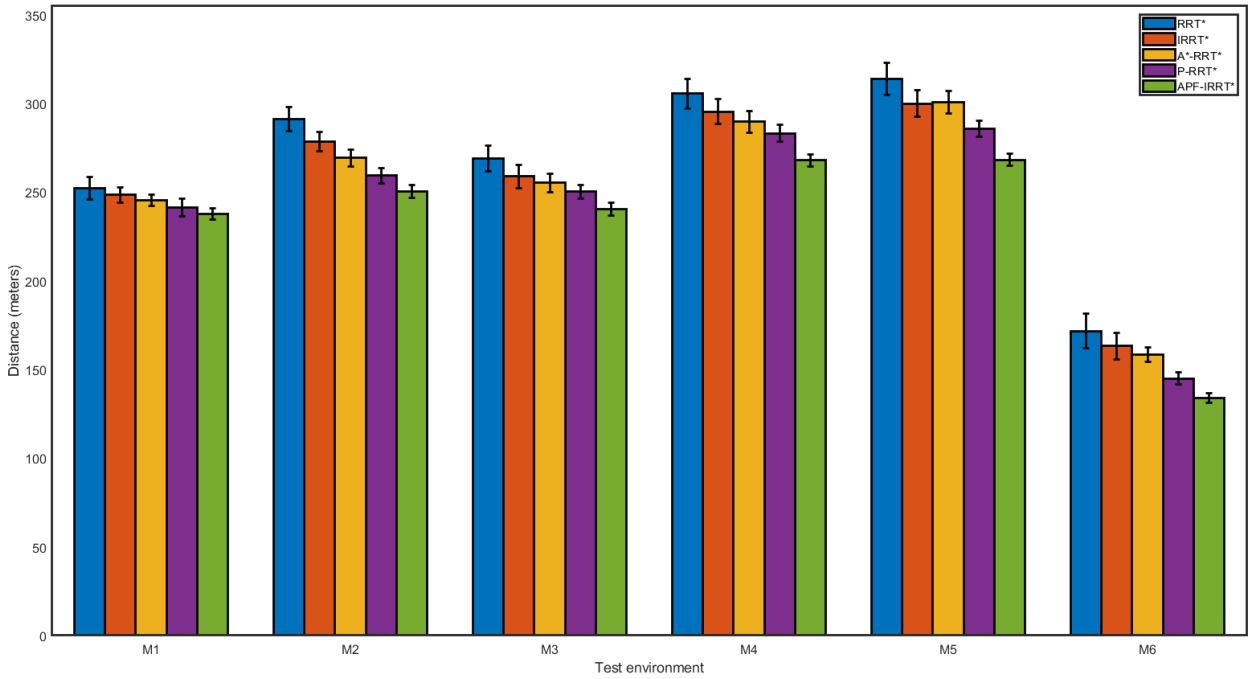


Figure 7. Comparison of path length for five algorithms in six different test environments.

Figure 7 also shows that the standard deviation for the 20 independent experiments of the APF-IRRT* algorithm, compared with other four algorithms, is superior in each test environment. Similarly, taking the test environment M5 as an example, the RRT*, IRRT*, A*-RRT* and P-RRT* algorithms obtained a standard deviation of 9.21, 7.46, 6.45, and 4.60 m, respectively, while the standard deviation of APF-IRRT* algorithm was 3.35 m. This reveals that the path length value of the APF-IRRT* algorithm is closer to the mean value in each experiment, which indicates that the proposed algorithm has lower deviation than the other three algorithms. We can also see from Figure 7 that the more complex the environment is, the more apparent the superiority of the standard deviation of APF-IRRT* algorithm over the other algorithms is. For example, in test environment M1, the RRT*, IRRT*, A*-RRT* and P-RRT* algorithms obtained a standard deviation of 6.54 m, 5.00 m, 4.05 m, and 4.16 m, respectively; the standard deviation of APF-IRRT* algorithm was 3.18 m. This gives a difference of 3.36, 1.82, 0.87, and 0.98 m between the RRT*, IRRT*, A*-RRT*, P-RRT* algorithms and the APF-IRRT* algorithm. While in test environment M5, the differences are 5.86, 4.11, 1.41, and 1.25 m, respectively. It is indicated that compared with the other three algorithms, the APF-IRRT* algorithm has an advantage in complex environments.

Another important evaluation parameter is search time. Table 2 gives the mean value of search time for the four algorithms mentioned above in six different test environments described in Table 1. All simulations were performed 20 times. As can be seen in Table 2, the APF-IRRT* algorithm could find the optimal solution faster than other algorithms. Again, taking the environment M5 as an example, it gives a difference of 1.14 s between the APF-IRRT* and IRRT* algorithms. According to the search time of the algorithm, the search efficiency of APF-IRRT* algorithm increases by about 35%, 30.5%, and 24% more than that of IRRT*, A*-RRT*, and P-RRT* algorithm, respectively. The simulation results indicate that the APF-IRRT* algorithm, by introducing the virtual force of APF into the expansion stage of the search tree, increases the guidance for the algorithm, greatly improves the search efficiency, and further shortens the path length of the algorithm.

Table 2. The mean value of search time for 20 independent simulations of RRT*, IRRT*, A*-RRT*, P-RRT* and APF-IRRT* algorithms.

Environment	RRT*	IRRT*	A*-RRT*	P-RRT*	APF-IRRT*
M1	3.56	2.31	2.33	2.01	1.53
M2	4.08	2.53	2.52	2.25	1.67
M3	4.01	2.51	2.45	2.15	1.60
M4	4.21	3.01	2.78	2.55	1.93
M5	4.50	3.24	3.02	2.76	2.10
M6	2.02	1.64	2.84	1.58	1.22

To verify the effectiveness of the APF-IRRT* algorithm in more complex environments, we construct a grid map based on a real environment of our laboratory, as shown in Figure 8. We can see from Figure 8 that the proposed algorithm can find a best path which has the shortest collision-free path length from the start point to the target point. It also indicates that the proposed APF-RRT* algorithm is effective in a more complex environment.

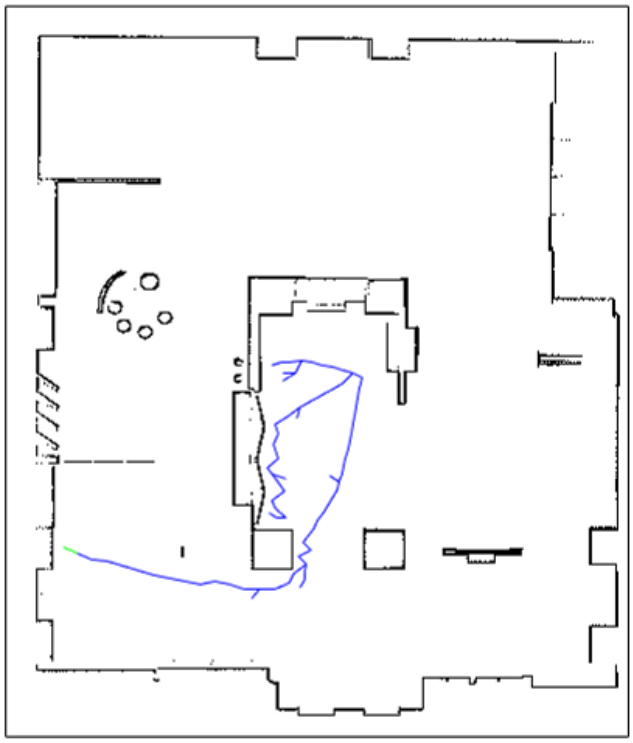
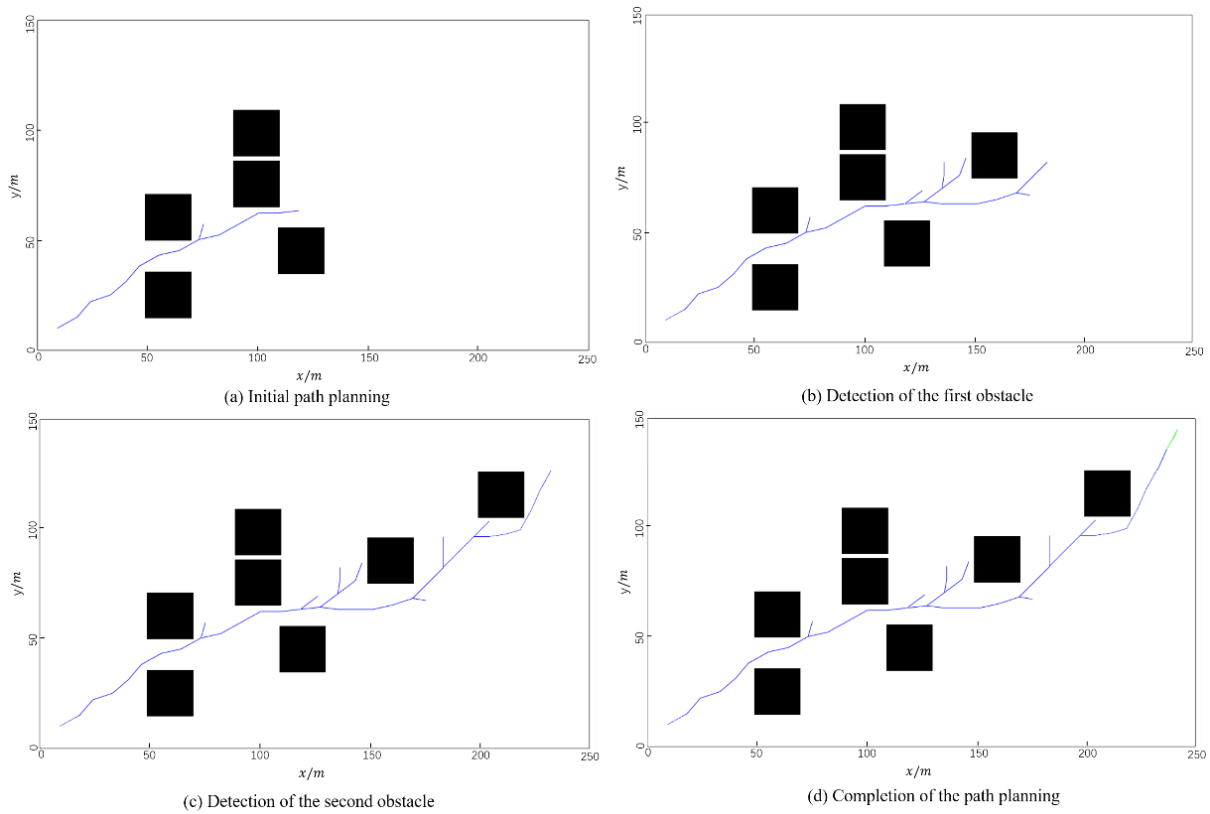


Figure 8. A simulation in a more complex environment.

4.3. Path Planning in a Dynamic

In real working scenarios, the mobile robots are usually faced with environmental changes—one or more unknown obstacles appear. To ensure the mobile robot works well, the state-of-the-art path planning methods present solutions to respond to the unknown information and adjust the original path by considering the unknown information. In this sub-section, we employ the proposed APF-IRRT* algorithm to perform the on-line path planning in the test environment M5 where some unknown random static obstacles appear on the known path.

To complete the experiment, we perform the path planning as the following steps: First, we perform the off-line path planning in the known test environment M5 in which information is described in Table 1. An average path length of 268.01 m from the start point to the target point is obtained. Next, when the mobile robot travels along the original path at position (120, 60), the mobile robot senses a new obstacle. It calculates the obstacle position and re-plans a new feasible path to avoid collision with the obstacle, as shown in Figure 9b. Then, when the mobile robot travels along the new path at position (190, 85), the mobile robot senses another obstacle. Again, the mobile robot calculates the second obstacle position and re-plans another feasible path to avoid collision with the second obstacle, as shown in Figure 9c. Finally, the mobile robot travels along the latest path to the target point, as shown in Figure 9d. The total path length from the start point to the target point is 285.54 m.



(a) Initial path planning (b) Detection of the first obstacle

(c) Detection of the second obstacle (d) Completion of the path planning

Figure 9. Path planning results with two random static obstacles in test environment M5.

4.4. Effect of Complexity and Size of Grid Map in Search Time of the Algorithms

To test the effect of different complexities and sizes of grid maps in the search time of the proposed hybrid algorithm, two different complexity maps and two different area maps are selected for the simulation using the four algorithms mentioned above.

Figure 10a shows a comparison of search times for the four algorithms in the test environments M2 and M5. The results were obtained by repeating the simulations 20 times independently. We can see from the figure that the complexity does not change the trend

of the curves, nor change the superiority of APF-IRRT* algorithm over the other three algorithms. It indicates that the proposed APF-RRT* algorithm has a good ability to adapt to the complexity of the map.

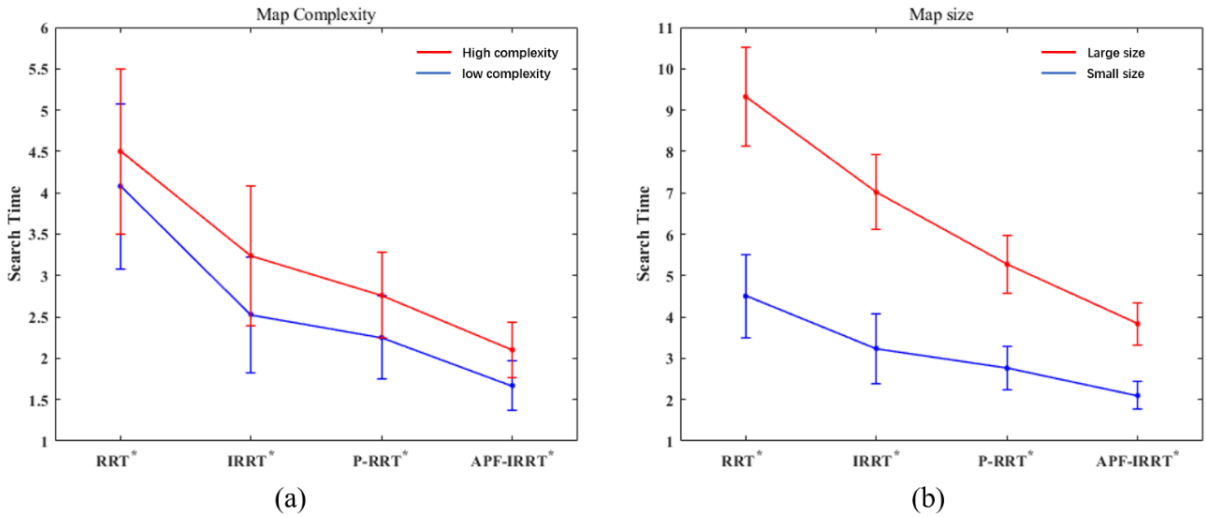


Figure 10. The effect of complexity and size of grid map in search time of four algorithms. (**a**) shows the effect of two different complexity grid maps in search time of four algorithms. The red and blue curves are test environment M5 and M2, respectively. (**b**) shows the effect of two different area grid maps in the search time of four algorithms. The red and blue curves are relatively larger area map and smaller area map, respectively.

To test the effect of different sizes of grid maps for the proposed algorithm, the test environment M5 was enlarged to 300 m $\times$ 500 m. The start point and the target point were reset to (10, 290) and (490, 10), and the coordinates of the obstacles were changed proportionally as (205, 135), (205, 170), (235, 215), (140, 185), and (140, 240), respectively. The comparison of the mean search time for 20 independent simulations of the four algorithms is shown in Figure 10b. We can see from the figure that the two curves, except for numerical differences, keep a similar trend. It indicates that the proposed algorithm has a good ability to adapt to the change of the size of the map.

4.5. Experiment in a Real Environment

To test the proposed APF-IRRT* algorithm for mobile robot path planning, we performed it on an open robotic platform: the Mecanum wheel omni-directional mobile robot U-car, as shown in Figure 11. The main parameters of the U-car are the maximum linear velocity (4.5 m/s) and the maximum angular velocity (3.5 rad/s).

Figure 11. The Mecanum wheel omni-directional mobile robot U-car which was used in the experiment.

The experiments were performed in a real laboratory environment considering the test environment M3 with three static obstacles. The size of the experiment area was set as 5 m × 5 m. The start point and the target point were set to (0, 2.5) and (5, 2.5), and the obstacles were set to (1.5, 3), (1.5, 2), (3, 2.5), respectively. For testing purposes, the linear velocity and the angular velocity of the omni-directional mobile robot were set to 1.0 m/s and 1.5 rad/s, respectively. The proposed APF-IRRT* algorithm planned a path to provide a sequence of motion commands to the mobile robot from the start point to the target point. The test results are shown in Figure 12, with a set of images of the mobile robot which show that the mobile robot drove at different positions. From Figure 12, we can see that the Mecanum wheel omni-directional mobile robot executes the path planning satisfactorily in a real test environment.

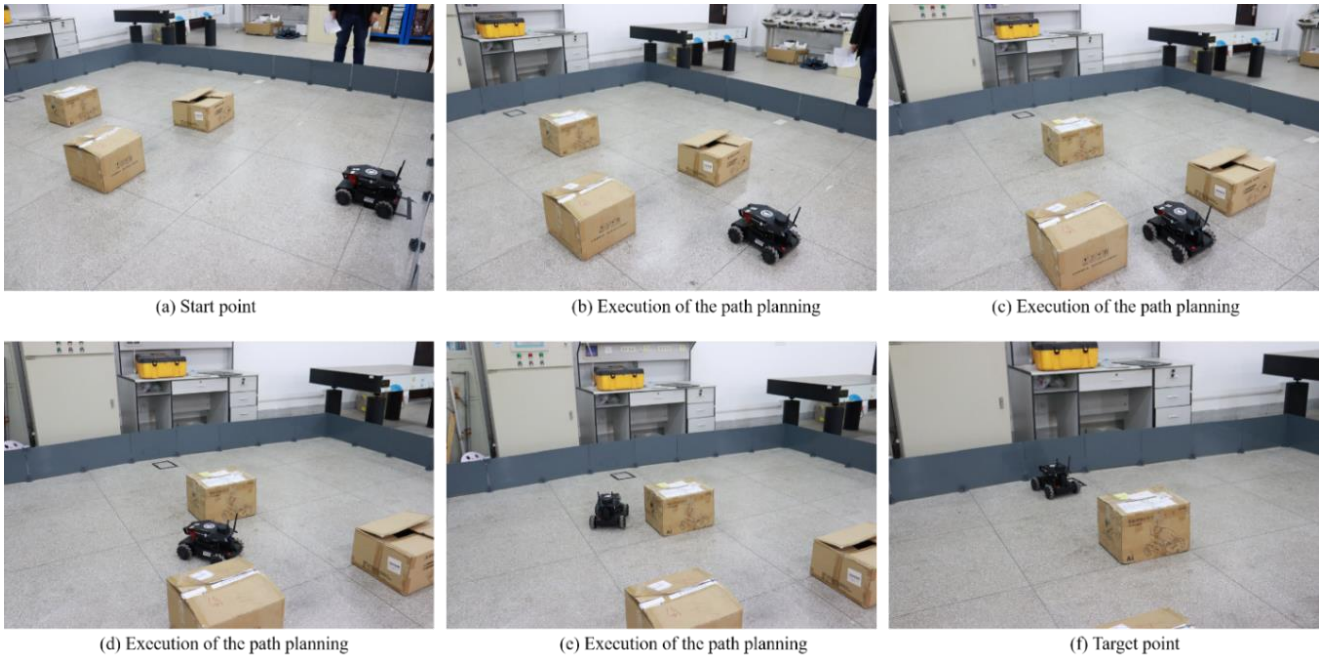


Figure 12. Experiment of the proposed APF-IRRT* algorithm in a real scenario.

To test the ability of avoiding dynamic obstacles for APF-IRRT* algorithm in a real environment, we performed the experiment with two Mecanum wheel omni-directional mobile robot U-cars, robot 1 was used as a main robot, robot 2 was used as a dynamic obstacle. To distinguish the two robots, the robot 1 was set to move backward, and its linear velocity and angular velocity were set to 0.4 m/s and 1 rad/s, respectively. Robot 2 was set to move forward, and its linear velocity and the angular velocity were set to 0.2 m/s. A set of images of the test results, which show how the robot moves at different positions, is shown in Figure 13. As shown in Figure 13, we can see that the main robot can avoid dynamic obstacles well.

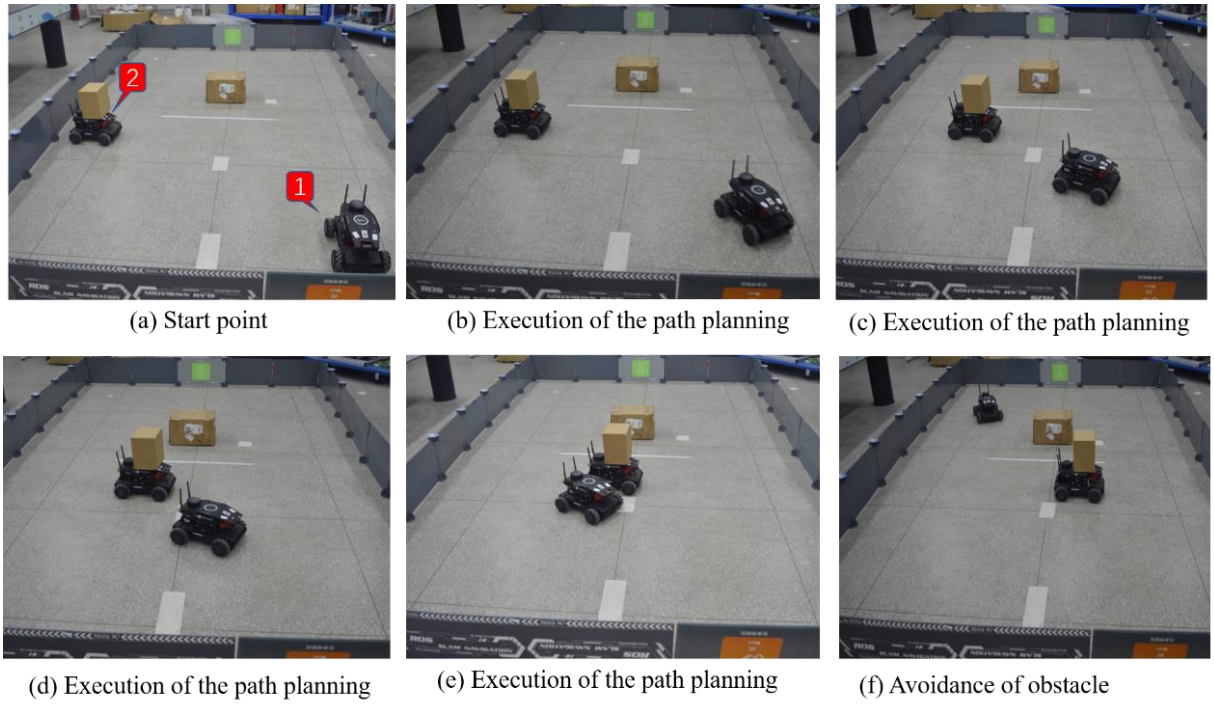


Figure 13. Experiment of avoidance of a dynamic obstacle of APF-IRRT* algorithm in a real scenario.

5. Conclusions

A hybrid algorithm based on the APF method and IRRT* algorithms was proposed to solve the path planning of mobile robots. By introducing the virtual force field of APF into the search tree expansion stage of the IRRT* algorithm, the guidance of the proposed algorithm was greatly improved, which greatly reduces extended nodes and further shortens the path length and search time. In addition, since the APF method is a local path planning algorithm which can construct an estimated map of the environment in real time, the proposed algorithm can be used both in known static scenarios and partially dynamic scenarios.

The simulations were performed in different static scenarios and a dynamic scenario. For the static scenarios, the results show that the hybrid APF-IRRT* algorithm, compared with RRT*, IRRT* and P-RRT* algorithms, has the smallest values in search time and path length. For the dynamic scenario, the APF-IRRT* also finds a feasible path to avoid collision with unknown obstacles. Therefore, the proposed APF-IRRT* algorithm exhibits great potential in real path planning applications.

As for future work, we will focus on the following aspects: (1) we are going to find a solution to obtain the optimal parameters automatically. In our experiment, the gain parameters of the APF were set heuristically, which leads to non-optimal parameters. (2) We will extend the proposed algorithm for the mobile robot path planning to a 3D environment since this work only considered a 2D environment. (3) We are considering improving the proposed algorithm in a maze-like environment since the success rate is not high comparing with that in other environments

Author Contributions: Conceptualization, D.W. and G.W.; methodology, G.D.; software, G.D.; validation, D.W. and L.T.; formal analysis, D.W.; investigation, D.W.; resources, D.W.; data curation, G.D.; writing—original draft preparation, D.W.; writing—review and editing, L.T. and L.W.; visualization, G.D.; supervision, G.W.; project administration, D.W.; funding acquisition, D.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Key Program of Natural Science Foundation of Anhui Provincial Education Department (KJ2020A0349, KJ2020ZD39), the Priming Scientific Research Foundation for the Introduction Talent in Anhui Polytechnic University (No: 2018YQQ008), the Advance Research Program for National Science Foundation in Anhui Polytechnic University (No: Xjky02201904) and the Open Research Fund of Anhui Key Laboratory of Detection Technology and Energy Saving Devices, Anhui Polytechnic University (No: DTES2020A05, DTESD2020A02).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Goerzen, C.; Kong, Z.; Mettler, B. A Survey of Motion Planning Algorithms from the Perspective of Autonomous UAV Guidance. *J. Intell. Robot. Syst.* **2010**, *57*, 65–100. [CrossRef]
2. Galceran, E.; Carreras, M. A survey on coverage path planning for robotics. *Robot. Auton. Syst.* **2013**, *61*, 1258–1276. [CrossRef]
3. Mac, T.; Cosmin, C.; Tran, D.T.; Keysera, R. Heuristic approaches in robot path planning: A survey. *Robot. Auton. Syst.* **2016**, *86*, 13–28. [CrossRef]
4. Park, J.; Kim, J.; Song, J. Path Planning for A Robot Manipulator Based on Probabilistic Roadmap and Reinforcement Learning. *Int. J. Control. Autom.* **2007**, *5*, 674–680.
5. Elbanhawi, M.; Simic, M. Sampling-based robot motion planning: A review. *IEEE Access* **2014**, *2*, 56–77. [CrossRef]
6. Paden, B.; Cap, M.; Yong, S.Z.; Yershov, D.; Frazzoli, E. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Trans. Intell. Veh.* **2016**, *1*, 33–55. [CrossRef]
7. Gonzalez, D.; Perez, J.; Milanes, V.; Nashashibi, F. A review of motion planning techniques for automated vehicles. *IEEE Trans. Intell. Transp Syst.* **2016**, *17*, 1135–1145. [CrossRef]
8. Zammit, C.; van Kampen, E.J. Comparison between A-star 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.
9. Koubaa, A.; Bennaceur, H.; Chaari, I.; Trigui, S.; Ammar, A.; Sriti, M.F.; Alajlan, M.; Cheikhrouhou, O.; Javed, Y. Introduction to Mobile Robot Path Planning. In *Robot Path Planning and Cooperation*, 1st ed.; Springer International Publishing: Cham, Switzerland, 2018; pp. 3–12.
10. Castro, D.; Leandro, N.; Jonathan, T. *Artificial Immune Systems: A New Computational Intelligence Approach*, 1st ed.; Springer International Publishing: Berlin/Heidelberg, Germany, 2002; pp. 57–58.
11. Zeng, W.; Church, R.L. Finding shortest paths on real road networks: The case for A*. *Int. J. Georg. Inf. Sci.* **2009**, *23*, 531–543. [CrossRef]
12. Zadeh, L.A. Fuzzy algorithms. *Inf. Control.* **1968**, *12*, 94–102. [CrossRef]
13. Javad, A.; Mansour, J. Adaptive motion planning with artificial potential fields using a prior path. In Proceedings of the 2015 3rd RSI International Conference on Robotics and Mechatronics (ICROM), Tehran, Iran, 7–9 October 2015; pp. 731–736.
14. Karaboga, D.; Akay, B. A Comparative Study of Artificial Bee Colony Algorithm. *Appl. Math. Comput.* **2009**, *214*, 108–132. [CrossRef]
15. Darrell, W. A genetic algorithm tutorial. *Stat. Comput.* **1994**, *4*, 65–85.
16. LaValle, S.M.; Kuffner, J.J. Rapidly-Exploring Random Trees: A New Tool for Path Planning. Ames, IA, USA, Tech. Rep. TR98-11. 1998. Available online: https://www.cs.csustan.edu/~{}xliang/Courses/CS4710-21S/Papers/06%20RRT.pdf (accessed on 8 August 2019).
17. Karaman, S.; Frazzoli, E. Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.* **2011**, *30*, 846–894. [CrossRef]
18. Gammell, J.D.; Srinivasa, S.S.; Barfoot, T.D. Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, 14–18 September 2014; pp. 2997–3004.
19. Gammell, J.D.; Barfoot, T.D.; Srinivasa, S.S. Informed sampling for asymptotically optimal path planning. *IEEE Trans. Robot.* **2018**, *34*, 966–984. [CrossRef]
20. Kuffner, J.; LaValle, S.M. RRT-connect: An efficient approach to single-query path planning. *Proc. IEEE Int. Conf. Robot. Automat.* **2000**, *2*, 995–1001.
21. Wu, Z.; Meng, Z.; Zhao, W.; Wu, Z. Fast-RRT: A RRT-based optimal path finding method. *Appl. Sci.* **2021**, *11*, 11777. [CrossRef]
22. Brunner, M.; Brüggemann, B.; Schulz, D. Hierarchical rough terrain motion planning using an optimal sampling-based method. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013; pp. 5539–5544.
23. Wang, W.; Zuo, L.; Xu, X. A learning-based multi-RRT approach for robot path planning in narrow passages. *J. Intell. Robot. Syst.* **2018**, *90*, 81–100. [CrossRef]

24. Jeong, I.; Lee, S.; Kim, J. Quick-RRT*: Triangular inequality-based implementation of RRT* with improved initial solution and convergence rate. *Expert Syst. Appl.* **2019**, *123*, 82–90. [CrossRef]
25. Wang, J.; Chi, W.; Shao, M.; Meng, Q.H. Finding a high-quality initial solution for the RRTs algorithms in 2D environments. *Robotica* **2019**, *37*, 1677–1694. [CrossRef]
26. Mashayekhi, R.; Idris, M.Y.I.; Anisi, M.H.; Ahmedy, I. Hybrid RRT: A semi-dual-tree RRT-based motion planner. *IEEE Access* **2020**, *8*, 18658–18668. [CrossRef]
27. Qureshi, A.; Ayaz, Y.; Ayaz, Y. Potential functions based sampling heuristic for optimal path planning. *Auton. Robot.* **2018**, *40*, 1079–1093. [CrossRef]
28. Orozco-Rosas, U.; Montiel, O.; Sepúlveda, R. Pseudo-bacterial potential field based path planner for autonomous mobile robot navigation. *Int. J. Adv. Robot Syst.* **2015**, *12*, 60715. [CrossRef]
29. Orozco-Rosas, U.; Montiel, O.; Sepúlveda, R. Mobile robot path planning using membrane evolutionary artificial potential field. *Appl. Soft Comput.* **2019**, *77*, 236–251. [CrossRef]