drones

MDPI

Article

Potential-Field-RRT: A Path-Planning Algorithm for UAVs Based on Potential-Field-Oriented Greedy Strategy to Extend Random Tree

Tai Huang [1,2], Kuangang Fan [2,3,4,*], Wen Sun [1,2], Weichao Li [2,3] and Haoqi Guo [2,3,4]

1 School of Mechanical and Electrical Engineering, Jiangxi University of Science and Technology, Hongqi Street No. 86, Ganzhou 341000, China
2 Magnetic Suspension Technology Key Laboratory of Jiangxi Province, Jiangxi University of Science and Technology, Hongqi Street No. 86, Ganzhou 341000, China
3 School of Electrical Engineering and Automation, Jiangxi University of Science and Technology, Hongqi Street No. 86, Ganzhou 341000, China
4 National Rare Earth Functional Material Innovation Center, Huilong Street No. 6, Ganzhou 341000, China
* Correspondence: fankuangang@jxust.edu.cn

Abstract: This paper proposes a random tree algorithm based on a potential field oriented greedy strategy for the path planning of unmanned aerial vehicles (UAVs). Potential-field-RRT (PF-RRT) discards the defect of traditional artificial potential field (APF) algorithms that are prone to fall into local errors, and introduces potential fields as an aid to the expansion process of random trees. It reasonably triggers a greedy strategy based on the principle of field strength descending gradient optimization, accelerating the process of random tree expansion to a better region and reducing path search time. Compared with other optimization algorithms that improve the sampling method to reduce the search time of the random tree, PF-RRT takes full advantage of the potential field without limiting the arbitrariness of random tree expansion. Secondly, the path construction process is based on the principle of triangle inequality for the root node of the new node to improve the quality of the path in one iteration. Simulation experiments of the algorithm comparison show that the algorithm has the advantages of fast acquisition of high-quality initial path solutions and fast optimal convergence in the path search process. Compared with the original algorithm, obtaining the initial solution using PF-RRT can reduce the time loss by 20% to 70% and improve the path quality by about 25%. In addition, the feasibility of PF-RRT for UAV path planning is demonstrated by actual flight test experiments at the end of the experiment.

Keywords: path planning; rapidly-exploring random tree; potential field; greedy strategy; root node iteration; unmanned aerial vehicles

Citation: Huang, T.; Fan, K.; Sun, W.; Li, W.; Guo, H. Potential-Field-RRT: A Path-Planning Algorithm for UAVs Based on Potential-Field-Oriented Greedy Strategy to Extend Random Tree. *Drones* **2023**, *7*, 331. https://doi.org/10.3390/drones7050331

Academic Editor: Pablo Rodríguez-Gonzálvez

Received: 19 April 2023
Revised: 18 May 2023
Accepted: 19 May 2023
Published: 21 May 2023

1. Introduction

With the development of technology for UAVs, the application of UAVs is becoming more and more widespread. Military UAVs can perform tasks such as intelligence collection, air early warning and electronic jamming [1–3]. Civil applications include aerial photography [4], agricultural plant protection [5], express transportation [6], geological mapping [7] and fault inspection [8]. UAVs complete their mission with a variety of capabilities, one of which is their path-planning capability.

Common path-planning algorithms can be classified as follows. Graph-based planning methods include A* [9], D* [10], and their optimization algorithms [11,12]. Since this type of algorithm is not suitable for path searching in high-dimensional space, it is often used for unmanned vehicles for path planning in the two-dimensional plane. Bionic-based planning methods, such as GA [13], ACO [14,15], and PSO [16], is a class of algorithms more often used to deal with patrols or secondary path optimization. Potential-field-based planning methods mainly include APF and its optimization algorithms [17–19], which have the

problem of falling into local optimization. Sampling-based planning methods, such as RRT [20] and PRM [21], can be used to solve high-dimensional path-planning problems. In addition, their advantages are more obvious as the dimensionality increases. For this reason, the study of RRT and its optimization algorithm has received considerable attention from scholars in recent years.

Steven M. LaValle proposed the RRT algorithm [20] in 1998 to solve the robot path-planning problem, attempting to expand the path through node iteration, making the path branch like a tree until the branch touches the target point. Kuffner et al. proposed RRT-connect [22] in 2000, which introduces a dual-tree expansion strategy and adds a greedy strategy to the expansion of auxiliary trees to reduce the map complexity by expanding both trees simultaneously. Karaman et al. proposed the RRT* [23] algorithm in 2011, which improves path quality through root node reconstruction, sacrificing small time costs and saving a large amount of path costs, making RRT* the most successful variant of RRT. In 2019, Jeong proposed a Q-RRT* [24] algorithm based on RRT*, which optimizes the path cost of each node under the current expansion state through root node iteration. In addition, the development of RRT has also derived a series of optimization algorithms such as BG-RRT [25], Informed RRT* [26], PQ-RRT* [27], F-RRT* [28], and DPRRT* [29].

This paper proposes an improved path-planning algorithm: PF-RRT, which incorporates a potential-field-oriented greedy strategy in RRT. It uses the field strength matrix of the artificial potential field to calculate and determine whether to perform greedy extension of the random tree, and speeds up the search of the random tree to the target point. After a new node is generated, one iteration of its root node is performed, making PF-RRT improve the path quality without increasing the time cost. In addition, the performance and feasibility of PF-RRT is demonstrated by conducting simulation tests of the algorithm, ROS-based PX4 simulation flight tests, and actual UAV flight tests in turn.

The structure of this paper is as follows: In Section 2, the path-planning problem is described and defined, and the ideas of the RRT algorithm and APF algorithm are introduced to assist in understanding PF-RRT. In Section 3, DG-RRT is proposed to solve the path-planning problem. In Section 4, the superiority of the algorithm is analyzed by comparing the simulation experiments in two-dimensional and three-dimensional states, and the flight states of the UAV in the simulated and real environments are shown.

2. Related Works

In this section, the path-planning problem and its objectives are described. The principles of RRT and APF algorithms for path searching are also introduced in depth to aid in understanding the conception of PF-RRT.

2.1. Problem Definition

$X = (0,1)^n$ is a n-dimensional constructed space, X_{hinder} is an obstacle region, and X_{free} is a blank region satisfying

$$\begin{cases} X_{free} \cap X_{hinder} = \varnothing \\ X_{free} \cup X_{hinder} = X \end{cases}. \tag{1}$$

Meanwhile, a path-planning problem is defined through $(X_{free}, X_{init}, X_{goal})$, where $X_{init} \in X_{free}$ is the initial starting point and $X_{goal} \in X_{free}$ is the target region. Let a continuous function $\Phi : [0,1] \mapsto X$ with bounded variation be a path, and the path Φ is an unobstructed path when $\forall \tau \in [0,1]$ and $\Phi(\tau) \in X_{free}$.

Definition 1. *Effective path planning is finding a feasible path Φ for a given motion planning problem $(X_{free}, X_{init}, X_{goal})$ and making Φ collision-free, where*

$$\begin{cases} \Phi(0) = X_{init} \\ \Phi(1) \in X_{goal} \end{cases}. \tag{2}$$

Definition 2. *Optimal path planning is to find a feasible path* Φ^* *for a given motion planning problem* $(X_{free}, X_{init}, X_{goal})$ *such that the path cost* Φ *is minimized and such that it satisfies*

$$\begin{cases} c(\Phi^*) = \min\{c(\Phi)\} \\ \Phi \in X_{free} \end{cases}, \tag{3}$$

where $c(\Phi)$ *is the path cost of path* Φ *from* X_{init} *to* X_{goal}.

2.2. Rapidly-Exploring Random Tree

"Speed" is the biggest advantage of RRT, and randomness is also its biggest advantage, which makes it free from the trouble of falling into local optimization. The solution idea for the feasible path is to quickly expand the path like a tree, spreading from the starting point x_{init} to the surrounding areas until the target point x_{goal} is searched. It is very common to address the dynamic programming problem by constructing a weighted graph $G = (V, E)$, where $V = \{x_1, x_2, x_3, \cdots x_n\}$ is the vertex set composed of random tree nodes $x_i (i \in \{1, 2, \cdots n\})$ and E is the edge set. Algorithm 1 provides a basic RRT construction idea, which is an iterative process that tries to generate a feasible new node x_{new} for each iteration to expand the random tree after setting the end condition.

Algorithm 1 RRT

Input: Map, x_{init}, x_{goal};
Output: $G = (V, E)$;
$V = \{x_{init}\}$, $E = \emptyset$, $i = 0$;
while *True* do
    $x_{rand} \leftarrow Sampling(i)$;
    $i \leftarrow i + 1$;
    $x_{nearest} \leftarrow Nearest(G, x_{rand})$;
    $x_{new} \leftarrow GetNode(x_{rand}, x_{nearest})$;
    if $ObstacleFree(x_{new}, x_{nearest})$ then
        $V \leftarrow V \cup \{x_{new}\}$;
        $E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$;
    if $x_{new} \approx x_{goal}$ then
        return (V, E);

The original procedure used in the RRT algorithm is described below:

- $Sampling()$: Given a weighted map $G = (V, E)$, it returns a random coordinate point in the map as a guide point x_{rand}.
- $Nearest()$: Given $G = (V, E)$ and guide point x_{rand}, it calculates the point closest to x_{rand} in V as the extended proximal point $x_{nearest}$, and returns it.
- $GetNode()$: Given $x_{nearest}$ and x_{rand}, it takes $x_{nearest}$ as the starting point x_{rand} as the guide, intercepts the point at which the set step length of the upper guide distance $x_{nearest}$ is x_{new}, and returns to x_{new}.
- $ObstacleFree()$: Given $x_{nearest}$ and x_{new}, it returns *True* or *False* after collision checking. In addition, the calculated distances are all Euclidean distances.

2.3. Artificial Potential Field

The basic principle of the APF is to construct a potential-field function to quantify the influence of the current position by the potential field. Any position in the map is subject to the combined force of gravitational and repulsive forces, where the target point gives the gravitational force and the obstacle provides the repulsive force. It searches for the direction in which the potential energy decreases as the optimal path at the current moment, and ultimately moves to the target point. The gravitational and repulsive forces of an artificial potential field are shown in Figure 1.

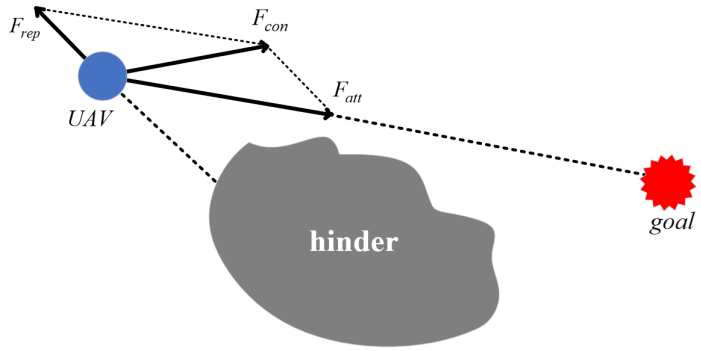


Figure 1. Schematic diagram of comprehensive force.

Any position in space is affected by the gravitational force of the target point, and the farther away from the target point, the greater the gravitational force received. The gravitational potential-field function $U_{att}(q)$ can be defined as Equation (4):

$$U_{att}(q) = \begin{cases} \frac{1}{2}K_{\xi}\rho^2(q, q_{goal}) & q \in \Omega \\ 0 & q \notin \Omega \end{cases}, \tag{4}$$

where q ($q \in \Omega$, Ω is the area for path planning) is the current position, K_{ξ} is the gravitational gain parameter, and $\rho(q, q_{goal})$ is the Euclidean distance from the target point. The corresponding gravitational force calculation function through the gravitational potential-field model is as follows:

$$F_{aat}(q) = -\nabla U_{att}(q) = \begin{cases} K_{\xi}\rho(q_{goal} - q) & q \in \Omega \\ 0 & q \notin \Omega \end{cases}. \tag{5}$$

The repulsive-field for the local field is subject to repulsive force when entering a specific range of the repulsive field, and the closer to the barrier it is, the greater the repulsive force. The repulsion field function $U_{rep}(q)$ is defined as Equation (6):

$$U_{rep}(q) = \begin{cases} \frac{1}{2}K_{\eta}\left(\frac{1}{\rho(q,q_{obs})} - \frac{1}{\rho_0}\right)^2 & \rho(q, q_{obs}) \le \rho_0 \\ 0 & \rho(q, q_{obs}) > \rho_0 \end{cases}, \tag{6}$$

where K_{η} is the repulsive field gain coefficient, q_{obs} is the obstacle position, $\rho(q, q_{obs})$ is the distance from the current position to the obstacle, and ρ_0 is the repulsive range. The corresponding repulsive force calculation function through the repulsive potential-field model is

$$F_{rep}(q) = -\nabla U_{rep}(q) = \begin{cases} K_{\eta}\left(\frac{1}{\rho(q,q_{obs})} - \frac{1}{\rho_0}\right)\frac{1}{\rho^2(q,q_{obs})}\nabla\rho(q, q_{obs}) & \rho(q, q_{obs}) \le \rho_0 \\ 0 & \rho(q, q_{obs}) > \rho_0 \end{cases}. \tag{7}$$

Combining the gravitational occasion repulsive field, the combined force at the current position is

$$F(q) = F_{aat}(q) + \sum_{1}^{n} F_{rep(n)}(q). \tag{8}$$

2.4. Analysis of the Advantages and Disadvantages of Two Algorithms

APF is essentially a feedback control method that uses a gradient descent mechanism to construct a feasible path, which has the advantage of being somewhat robust to control. However, it has the very obvious disadvantage that it tends to fall into local minima in complex environments. In addition, it is accompanied by an exponential increase in the number of operations when solving high-dimensional path-planning problems. The RRT algorithm is the exact opposite of APF. RRT has a great advantage in high-dimensional

path planning, but the stochastic nature of scaling causes its time advantage to be gradually canceled out. The ideal answer to solving path-planning problems is to obtain high-quality feasible paths quickly. By combining the advantages of the rapid scaling of random trees with the optimal orientation of potential fields, it becomes possible to quickly obtain better paths in high-dimensional environments.

3. Potential-Field-RRT

The APF algorithm has a good orientation to the path expansion trend, but it is easy to fall into local misconceptions in a slightly complex map. Combining APF to improve the RRT algorithm, a greedy algorithm based on potential-field orientation is added to the expansion process of the random tree, which accelerates its expansion to more optimal regions and avoids the problem of local errors. In addition, performing one iteration of the root node after generating new nodes makes it possible to improve the path quality without increasing the time cost.

3.1. Greedy Strategy Based on Potential-Field Orientation

First, the artificial potential field is constructed to calculate the force field for the path-planning map area. In practical applications, it is common to slice the large area, calculate the middle value of the small area as the field average force, and output the force field in the form of a matrix. This setup has two advantages:

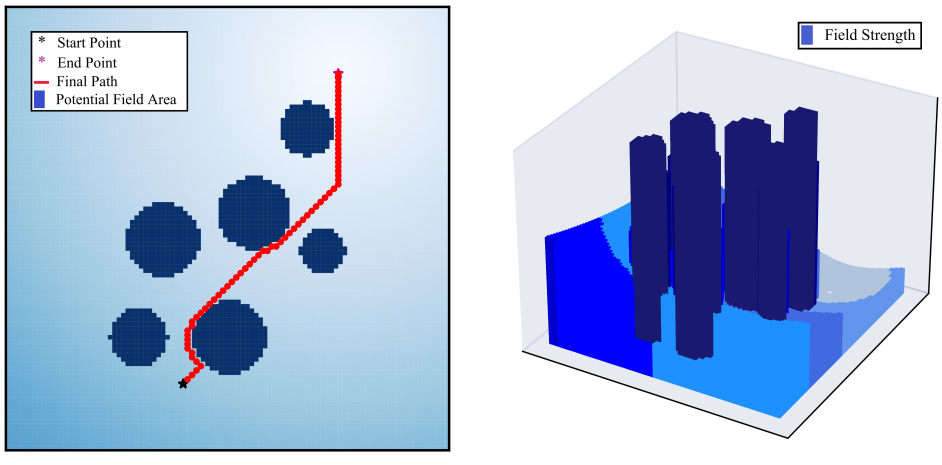
I. It is more convenient to obtain the path field mean force in the form of a look-up table, and it can improve the efficiency of operation;

II. It is easier to distinguish between obstacle areas and blank areas based on the data of the field force matrix, thus making collision detection easy.

Setting the map length and width as W and L, each small area resolution is R_{reso}, then the length and width can be divided into the number of regions $M = W/R_{reso}$ and $N = L/R_{reso}$. Each small area subject to the potential-field force:

$$\begin{cases} F_{aat}(q_{i,j}) = K_\xi \rho(q_{goal} - q_{i,j}) \\ F_{rep}(q_{i,j}) = K_\eta \left(\frac{1}{\rho(q_{i,j}, q_{obs})} - \frac{1}{\rho_0}\right) \frac{1}{\rho^2(q_{i,j}, q_{obs})} \nabla \rho(q_{i,j}, q_{obs}) \\ F(q_{i,j}) = F_{aat}(q_{i,j}) + \sum_{1}^{n} F_{rep}(q_{i,j}) \end{cases}, \tag{9}$$

where $F(q_{i,j})$ denotes the position coordinates of the calculated small area ($i \in [0, M-1]$, $j \in [0, N-1]$), n denotes the number of obstacles, and the final output F_{con} is a two-dimensional matrix of size $M \times N$. The output field strength data are visualized as shown in Figure 2a,b.



(**a**) (**b**)

Figure 2. Potential-field visualization gradient diagram: (**a**) potential-field gradient plan; and (**b**) potential-field stacking diagram.

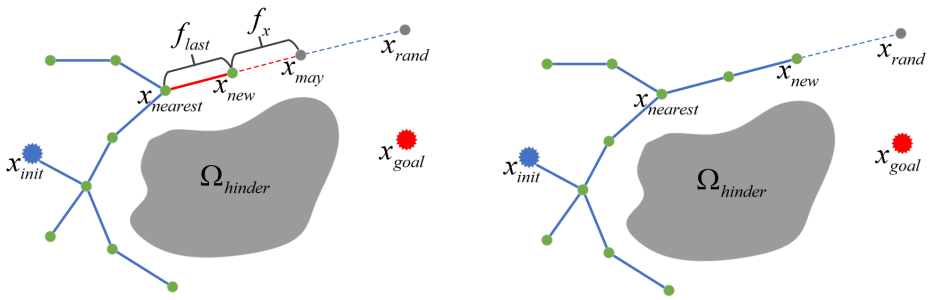
Figure 2a is a flat gradient map; the deeper the color, the greater the force. The output data are stacked according to the size of the value to build Figure 2b. Ideally, the collision-free path extends from the high potential energy region to the low potential energy region. The potential field constructed in this section is only used to assist in the expansion of the random tree. f is a small-area potential field, which is mainly used to obtain the potential energy of a node through coordinate transformation and to calculate the average potential energy of the path. Let the coordinates of the node be $node(x, y)$, then the force on the point is $f_{node(x,y)}$:

$$f_{node(x,y)} = F(q_{\text{int}(x/R_{reso}),\text{int}(y/R_{reso})}), \tag{10}$$

where $int(x)$ means that x is forced to be converted to an integer. The calculation of the mean field force on a section of the path is performed by sampling the section of the path and then calculating the potential field for each sample point to find its mean value f_x:

$$f_x = \frac{1}{m}\sum_{a=1}^{m} f_{node(x_a,y_a)}\ , \tag{11}$$

where m is the number of sampling points on the path. The selection of whether to use the greedy algorithm for path expansion is made by calculating and analyzing the f_x values on the path, as shown in Figure 3.



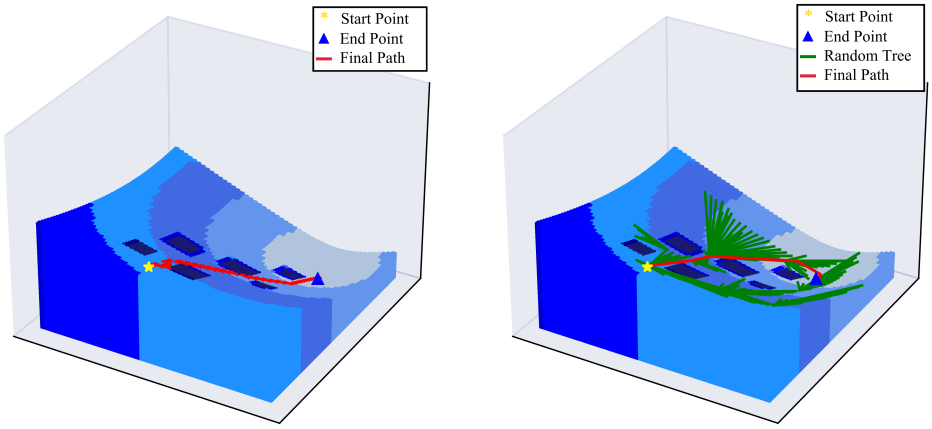
(**a**) (**b**)

Figure 3. Schematic diagram of greedy strategy based on potential-field orientation: (**a**) comparing average field force; and (**b**) path extension.

Figure 3a,b show the schematic diagram of greedy strategy expansion based on the potential-field orientation, where Figure 3a generates random guide points x_{rand}, after calculating the proximity point $x_{nearest}$, after touch detection that satisfies $f_x < F_{threshold}$ ($F_{threshold}$ is a set threshold defining the obstacle area and blank area) after intercepting the end point of the set step as a new node x_{new} according to the single-step expansion step after intercepting $x_{nearest}$ as the center of the circle $\overrightarrow{x_{nearest}x_{rand}}$ as the direction. The difference between other random tree algorithms is that instead of regenerating new bootstrap points immediately after generating new nodes to start the expansion of new arguments, the bootstrap points generated in the previous round are then used to further expand in the current direction by a factor of two steps to generate feasible nodes x_{may}, and calculate the field-average force f_{last} on path $l_{x_{nearest}x_{new}}$ and the field-average force f_x on path $l_{x_{new}x_{may}}$. When satisfied $f_x < f_{last}$, x_{new} is used as the proximal point $x_{nearest}$, and x_{may} is stored as the new node x_{new} to repeat the round. The bootstrap point x_{rand} is regenerated when $f_x > F_{threshold}$ or $f_x \geq f_{last}$ is obtained based on the potential-field matrix F. The expansion of the random tree is re-performed, as shown in Figure 3b.

As shown in Figure 4a,b, the potential-field stacking plots are shown for APF and combined with RRT improvement, respectively. As the blue area deepens, the potential-field force gradually increases. The yellow 'star' is the starting point, the blue 'triangle' is the end point, and the red route is the output path. In addition, the green route in Figure 4b is the extended random tree. In complex maps, APF is prone to fall into local minima, and the stochastic nature of RRT allows it to eliminate that problem. In addition, it can be seen

from Figure 4b that the gradient descent mechanism makes the random tree expand more diffusely in the better region, so it can accelerate the search to the target point.

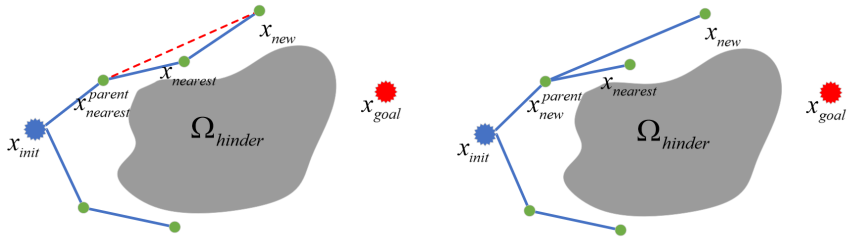


(**a**) (**b**)

Figure 4. Comparison of the effect before and after algorithm improvement: (**a**) APF; and (**b**) RRT-based APF.

3.2. Root Node Iterates Once

To obtain a higher-quality path solution, when a new node x_{new} is created, instead of storing $x_{nearest}$ as its root node, an iteration is performed on its root node. If the new path constructed passes the touch detection, the root node is updated forward once. Compared with the conventional algorithm, which directly takes nearest point $x_{nearest}$ as the root node, this approach sacrifices only a little time cost for more path cost. The schematic diagram is shown in Figure 5.



(**a**) (**b**)

Figure 5. Schematic diagram of root node iteration once: (**a**) probing the root node; and (**b**) path reconfiguration.

Where $x_{nearest}^{parent}$ is the root node of the proximal point $x_{nearest}$, when the new node x_{new} is obtained, $x_{nearest}$ is no longer directly stored as its root node x_{new}^{parent}. As shown in Figure 5a, assuming that a triangle is constructed with $x_{nearest}^{parent}$, $x_{nearest}$, and x_{new} as the vertex, it is known from the triangle theorem:

$$l_{x_{new}x_{nearest}^{parent}} < l_{x_{nearest}x_{nearest}^{parent}} + l_{x_{new}x_{nearest}}. \tag{12}$$

Therefore, its root node is selected by verifying the connectivity of $l_{x_{new}x_{nearest}^{parent}}$ and calculating the field-averaged force f_x on $l_{x_{new}x_{nearest}^{parent}}$, as shown in Equation (13):

$$\begin{cases} x_{new}^{parent} = x_{nearest}^{parent} & f_x < F_{threshold} \\ x_{new}^{parent} = x_{nearest} & f_x \geq F_{threshold} \end{cases}. \tag{13}$$

When the calculated field-averaged force f_b is less than the threshold $F_{threshold}$, as shown in Figure 5b, the reconfiguration path selects $x_{nearest}^{parent}$ as the root node of the new node x_{new}. The core idea of PF-RRT is to reduce the time cost of the search by accelerating the paths to better regions through a potential-field-oriented greedy strategy. In addition, a root node iteration is performed when a new node is generated, and the feasibility of path reconstruction is verified. The complete idea of the PF-RRT algorithm is shown in Algorithm 2.

Algorithm 2 PF-RRT

Input: Map, x_{init}, x_{goal};
Output: $G = (V, E)$;
$V = \{x_{init}\}$, $E = \emptyset$, $i = 0$, $f_{last} = F_{threshold} = 50$;
$F_{con} = FieldCalculation(Map)$;
while *True* do
  $x_{rand} \leftarrow Sampling(i)$;
  $i \leftarrow i + 1$;
  $x_{nearest} \leftarrow Nearest(G, x_{rand})$;
  while *True* do
    $x_{new} \leftarrow GetNode(x_{rand}, x_{nearest}, F_{con})$;
    $f_x \leftarrow ObstacleFree(x_{new}, x_{nearest}, F_{con})$;
    if $f_x < F_{threshold}$ and $f_x < F_{last}$ then
      $V \leftarrow V \cup \{x_{new}\}$;
      if $ObstacleFree(x_{new}, nearest_{nearest}, F_{con}) < F_{threshold}$ then
        $x_{nearest} \leftarrow nearest_{nearest}$;
      $E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$;
      $f_{last} \leftarrow f_x$;
      $x_{nearest} \leftarrow x_{new}$;
    else *break*;
    if $x_{new} \approx x_{goal}$ then
      return (V, E);

The following explains the original procedure in the PF-RRT that differs from the RRT:

- $FieldCalcuation()$: Given the map information, the map is sliced, and the potential-field force matrix F_{con} is calculated and output according to Equation (9).
- $ObstacleFree()$: Unlike other RRT algorithms, here the field-averaged force f_x of the detected segment path is calculated and returned based on the input potential-field force matrix F_{con}.

4. Simulation Experiment and Analysis

As shown above, PF-RRT consists of two parts, path expansion and root node iterative optimization, in order to obtain high-quality initial solutions faster. Based on the idea of the original RRT (shown in Algorithm 1), RRT, RRT*, and Q-RRT* are programmed and used as comparison algorithms to verify the performance of PF-RRT.

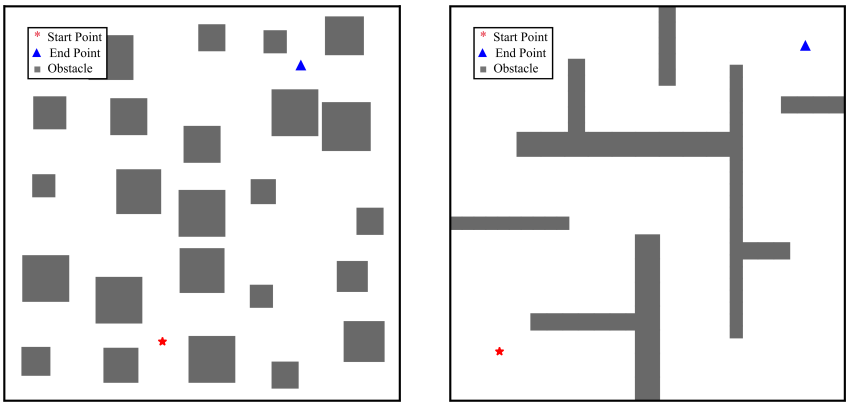
This paper conducts a series of experiments to evaluate the performance of the PF-RRT algorithm. The simulation experiment selects RRT as the benchmark algorithm for longitudinal comparison, and the classical RRT* and the newer Q-RRT* as the cross-sectional comparison algorithm, and then quantifies the output of RRT as the benchmark value normalized cross-sectional algorithm output to prove the performance advantages and disadvantages of the four algorithms in obtaining the initial solution. Meanwhile, iterative convergence experiments are conducted to compare the four algorithms and verify the performance of PF-RRT.

4.1. Simulation Environment and Its Initial Conditions

Pycharm was chosen as the simulation platform, the programming language is Python 3.7, and the computing processing core is Intel Core i7-7700, RAM 16G DDRT. Figure 6a,b show the simple fragmented obstacle map and the more complex maze map in

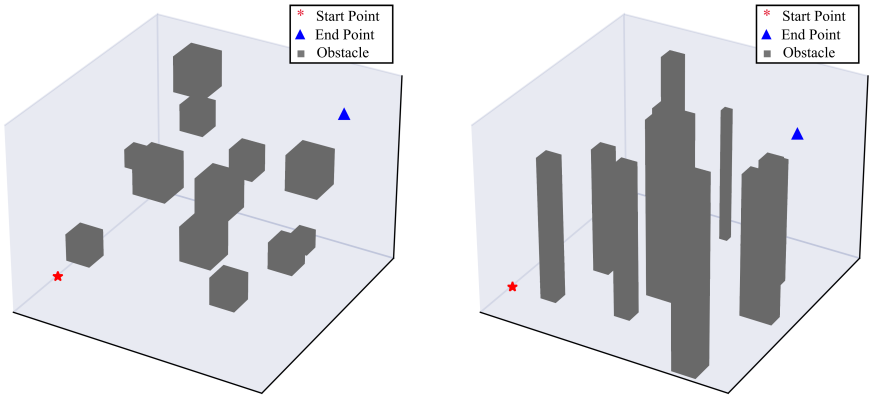
a two-dimensional plane, with the size of 200 m (horizontal direction) × 200 m (vertical direction). Figure 7a,b show the columnar map and the fragmented map in the three-dimensional space, with the size of 30 m (horizontal) × 30 m (vertical)× 20 m (vertical). In each map, the red pentagram represents the starting point, the blue triangle represents the target point x_{init}, and the gray area is the impassable obstacle area.

In order to obtain more realistic and comparable data, the single extension step size is set to 5 for each algorithm in the 2D environment, and the root node reconstruction search radius is set to 15 for both RRT* and Q-RRT*. The single extension step for each algorithm in the 3D environment is set to 1.5, and the root node reconstruction search radius for RRT* and Q-RRT* is 3. In order to keep the program construction as consistent as possible, the same program modules are used for all parts except for the original program specific to the algorithm itself.



(**a**) (**b**)

Figure 6. Two-dimensional test map: (**a**) 2D-map1; and (**b**) 2D-map2.



(**a**) (**b**)

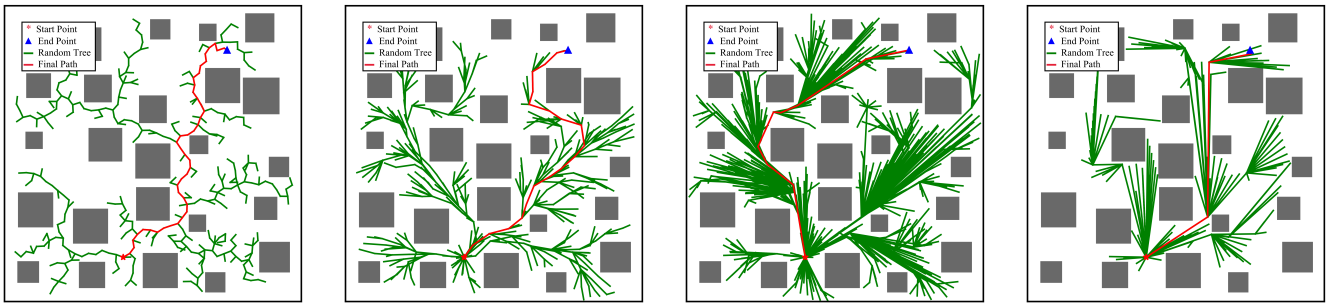
Figure 7. Three-dimensional test map: (**a**) 3D-map1; and (**b**) 3D-map2.

4.2. Analysis of Experimental Results in the Two-Dimensional Plane

The simulation experiments are set up with 1000 sets of repeated trials to obtain the initial solutions from each algorithm searching in 2 different 2D test maps. The corresponding experiments counts 1000 initial solution path costs and search times for each algorithm in each map. In addition, iterative experiments are used to compare the convergence states of the optimal paths obtained by different algorithms in the same time span.

Figure 8a–d show the path expansion results of each of the four algorithms when searching for the target point in 2D-map1. Intuitively, the path of the RRT algorithm is more sinuous; the extended path of RRT* is "bundle-like" and relatively more rounded;

Q-RRT* is an optimization algorithm based on RRT*, which iteratively searches the parent node during path reconstruction to make its path quality rise to a higher level, and the corresponding time loss is higher. Influenced by the potential-field orientation, the greedy strategy is no longer triggered arbitrarily and is only triggered when $\overrightarrow{x_{nearest}x_{new}}$ points to the direction of the falling potential field, making the path expand rapidly toward the target point.



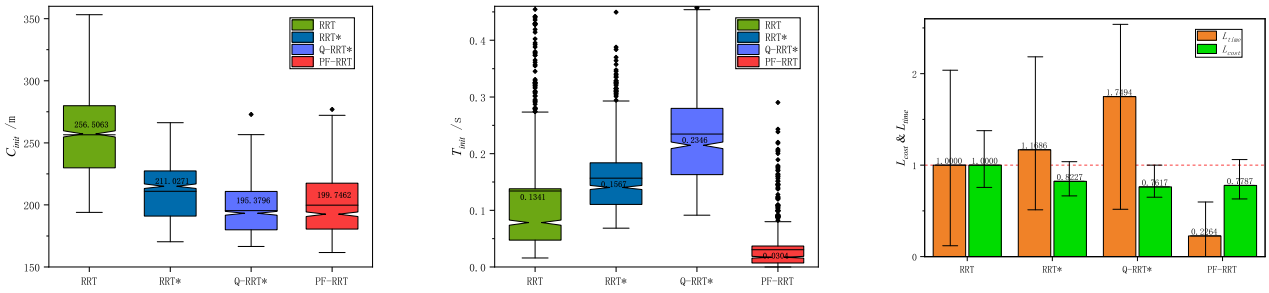
(**a**) (**b**) (**c**) (**d**)

Figure 8. Experimental results of 2D-map1: (**a**) RRT; (**b**) RRT*; (**c**) Q-RRT*; and (**d**) PF-RRT.

Figure 9a,b show the statistics of the path cost and search time for each of the 4 algorithms in 2D-map1 for 1000 iterations of testing and their data distribution. Figure 9c shows the path-cost loss rate and time loss rate of different algorithms relative to RRT, which are calculated as follows:

$$\begin{cases} l_{c(i)} = \sum_{a=1}^{1000} \frac{c_{a(i)}}{C_{mean(RRT)}} \\ l_{t(i)} = \sum_{a=1}^{1000} \frac{t_{a(i)}}{T_{mean(RRT)}} \end{cases}, \tag{14}$$

where $l_{c(i)}$ and $l_{t(i)}$ are relative path-cost loss rate and time loss rate (i denotes the algorithm category), $c_{a(i)}$ and $t_{a(i)}$ are path cost and time; in addition, $C_{mean(RRT)}$ and $T_{mean(RRT)}$ denote the mean value of path cost and mean value of search time for the initial solution of RRT. The data in Figure 8a–c show that Q-RRT* and PF-RRT obtain better path quality for a single solution of the simple map, but the average time loss of Q-RRT* is about 7.7 times that of PF-RRT. PF-RRT makes the random tree accelerate to approach the target point quickly in a simple map by a greedy strategy based on potential-field orientation, and the iterative optimization of one root node makes the path smoother.

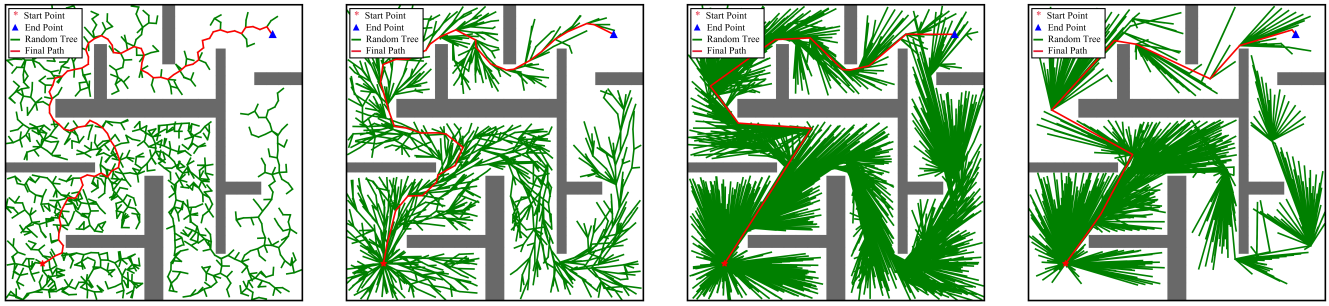


(**a**) (**b**) (**c**)

Figure 9. Statistical results of 2D-map1: (**a**) C_{init}; (**b**) T_{init}; and (**c**) L_{cost} & L_{time}.

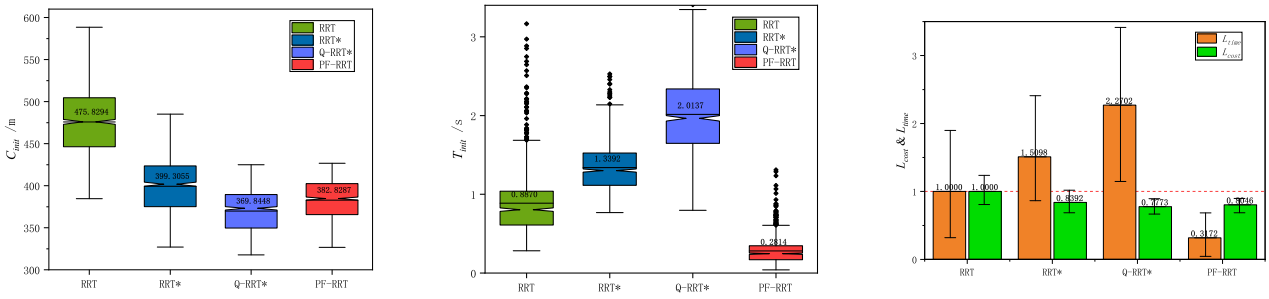
The results show that DG-RRT is able to obtain higher-quality initial solutions in relatively less time in a simple environment. In 2D-map1, the time loss of PF-RRT is only 22.6% of that of RRT, and its path quality is improved by about 22.1%.

As the nemesis of sampling-based algorithms, maze maps make the sampling process more complicated because of their winding paths. At the same time, it is used as a test map because it has fewer path channels and the trend of feasible solutions is more or less the same, so it is more intuitive to observe the strengths and weaknesses of its paths. Figure 10a–d show the status of the four algorithms for obtaining path solutions in single-channel maze maps, Figure 11a,b show the corresponding path cost and search time statistics, and Figure 11c shows the relative path-cost loss rate and time loss rate calculated based on Equation (14).



(**a**) (**b**) (**c**) (**d**)

Figure 10. Experimental results of 2D-map2: (**a**) RRT; (**b**) RRT*; (**c**) Q-RRT*; and (**d**) PF-RRT.

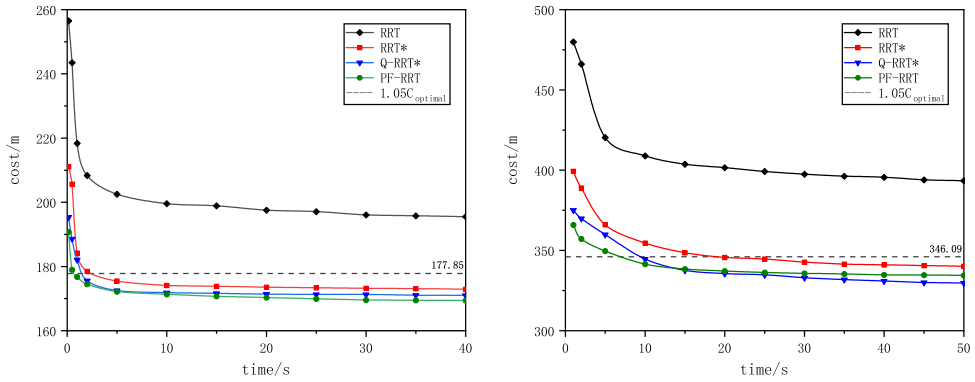


(**a**) (**b**) (**c**)

Figure 11. Statistical results of 2D-map2: (**a**) C_{init}; (**b**) T_{init}; and (**c**) L_{cost} & L_{time}.

Q-RRT* is an upgraded version of RRT*, and the time loss is further increased with the improvement of the path quality. Even though the iterative process of one root node increases the arithmetic power requirement, the preprocessed field strength matrix for obstacle perception and touch detection greatly improves the computational efficiency and reduces the time loss by 68.3% compared with the original RRT algorithm. The results in Figure 11c shows that PF-RRT is able to maintain its characteristics and obtain better path results with less time loss in relatively complex maps. Meanwhile, the error bars based on the statistics in Figure 11c shows that the path search results of PF-RRT are more stable compared to several other algorithms.

Figure 12a,b show the cost of optimal paths for each of the four algorithms in 2D-map1 and 2D-map2 in the same time span, where $C_{optimal}$ is the optimal path generation value. It can be seen that all algorithms except RRT have optimal convergence properties and are able to converge to $1.05C_{optimal}$ from Figure 12, with PF-RRT taking the shortest time to plan the suboptimal path. Statistical results show that the average time for PF-RRT to plan suboptimal paths is 0.1s and 7.5s in 2D-map1 and 2D-map2, respectively, requiring only 50% and 75 % of the time of Q-RRT*. Since PF-RRT generates near-suboptimal paths faster than other algorithms and it generates a solution in a much faster time, it is more suitable in a two-dimensional environment.



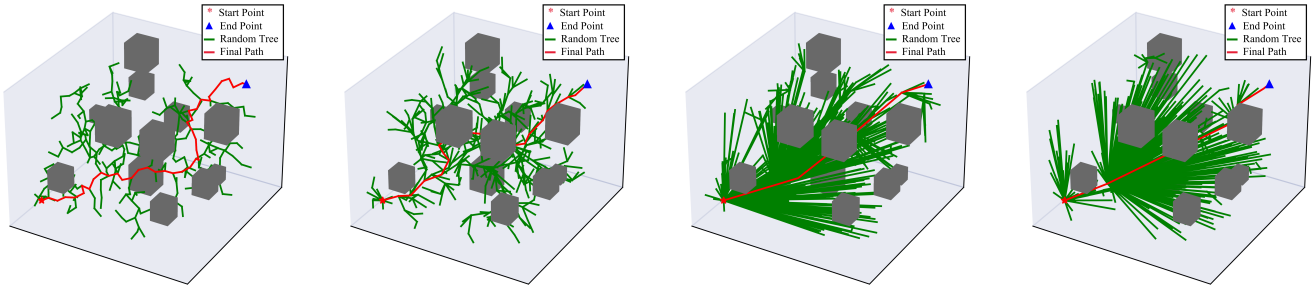
(**a**) (**b**)

Figure 12. Optimal convergence diagram in two-dimensional environment: (**a**) 2D-map1; and (**b**) 2D-map2.

4.3. Analysis of Experimental Results in Three-Dimensional Space

The UAV flight environment is three-dimensional space, for which two sets of experiments were designed to test the performance of the algorithms based on three-dimensional space, and the test maps are shown in Figure 6a,b. Figures 13a–d and 14a–d show the path search states of RRT, RRT*, Q-RRT*, and PF-RRT algorithms in 3D-map1 and 3D-map2. Simultaneously, 1000 sets of repetitive tests were conducted to obtain the initial solution search status of the 4 algorithms in each map, and the statistics are shown in Tables 1 and 2.

Figure 13a–d show the path search diagram in 3D-map1 for the four algorithms RRT, RRT*, Q-RRT*, and PF-RRT. Table 1 shows the eigenvalues of the path-cost statistics and time statistics of the four algorithms in obtaining the initial solution, where c_{mean}, c_{median}, c_{max}, and c_{min} are the mean, median, maximum, and minimum values of the 1000 sets of path cost statistics in 3D-map1, respectively. Meanwhile, t_{mean}, t_{median}, t_{max}, and t_{min} are the mean, median, maximum, and minimum values of 1000 sets of search time statistics, respectively. The statistical results show that PF-RRT has a higher quality and more stable path, and that it takes less time on average to search.



(**a**) (**b**) (**c**) (**d**)

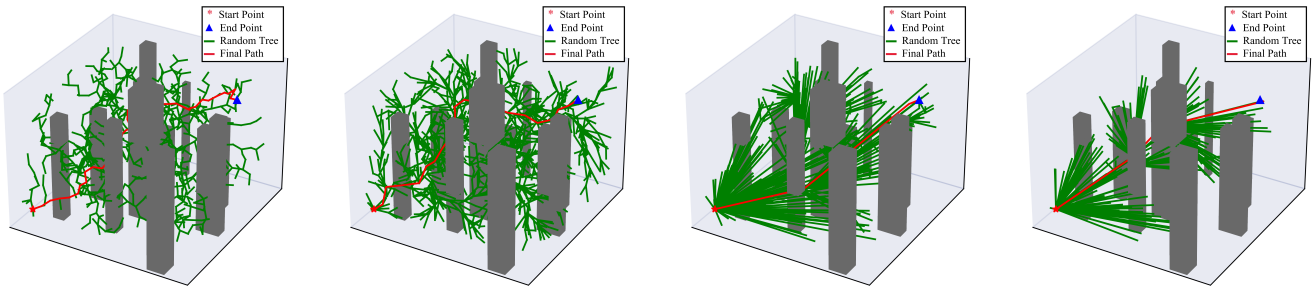
Figure 13. Experimental results of 3D-map1: (**a**) RRT; (**b**) RRT*; (**c**) Q-RRT*; and (**d**) PF-RRT.

Table 1. Eigenvalues of the 1000 sets of initial solution data in 3D-map1.

Method	c_{mean}/m	c_{median}/m	c_{max}/m	c_{min}/m	t_{mean}/s	t_{median}/s	t_{max}/s	t_{min}/s
RRT	56.83	56.51	77.67	44.26	0.90	0.26	21.74	0.02
RRT*	45.93	45.43	60.44	37.90	1.37	0.39	28.21	0.03
RRT-C	39.52	39.07	49.91	35.71	1.49	0.48	20.14	0.04
DG-RRT	38.74	38.37	47.10	34.58	0.79	0.21	17.10	0.01

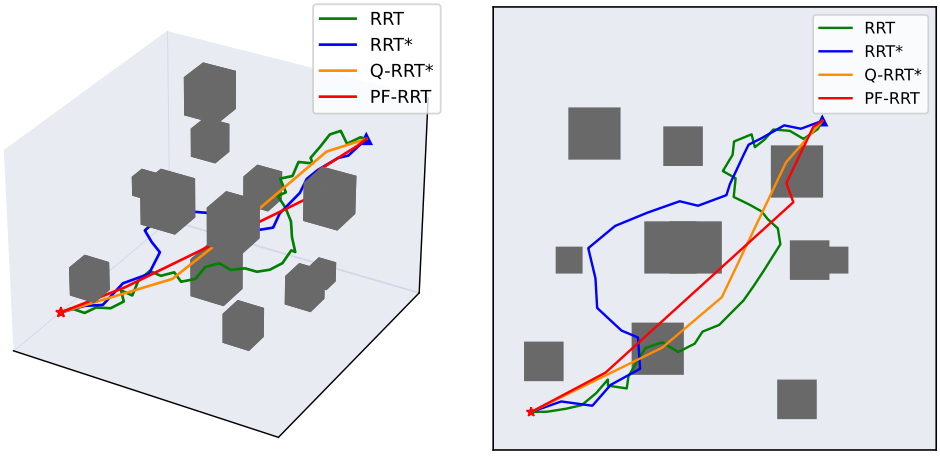
Figure 15a,b compare the paths output by the four algorithms in 3D-map1. It is obvious, both in the three-dimensional view and in the top view, that PF-RRT has a much

smoother path. In the top view of Figure 15b, the effect of the path crossing the obstacle can be seen because of the perspective view of the route.



(**a**) (**b**) (**c**) (**d**)

Figure 14. Experimental results of 3D-map2: (**a**) RRT; (**b**) RRT*; (**c**) Q-RRT*; and (**d**) PF-RRT.



(**a**) (**b**)

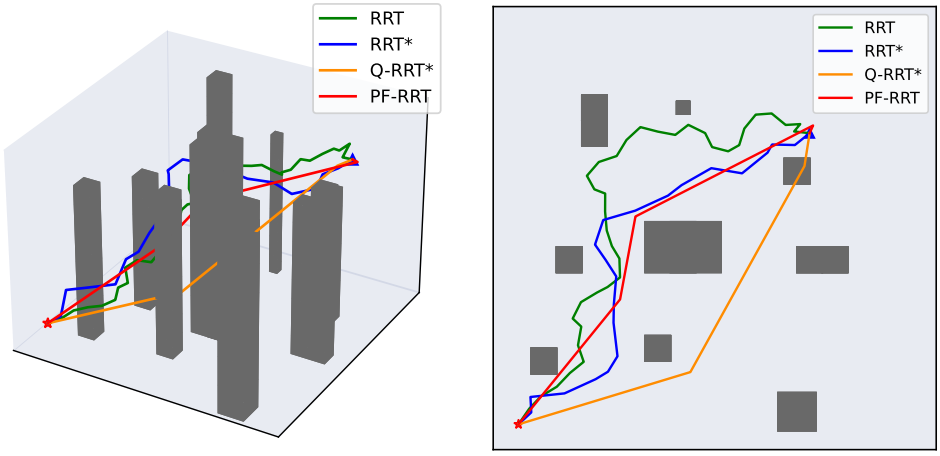
Figure 15. Trajectory comparison chart of 3D-map1: (**a**) stereogram; and (**b**) top view.

Figure 14a–d show the final paths output from the path search in 3D-map2 for four different algorithms. Table 2 shows the 4 algorithms performing 1000 sets of replicate trials in 3D-map2 to obtain the eigenvalues of the time and cost statistics during the initial solution, where its special symbols express the same meaning as in Table 1.

Table 2. Eigenvalues of the 1000 sets of initial solution data in 3D-map2.

Method	c_{mean}/m	c_{median}/m	c_{max}/m	c_{min}/m	t_{mean}/s	t_{median}/s	t_{max}/s	t_{min}/s
RRT	56.24	55.51	75.01	34.24	0.96	0.27	15.54	0.03
RRT*	45.68	45.34	59.29	35.56	1.79	0.53	20.36	0.06
RRT-C	39.25	39.04	45.71	37.83	2.64	1.01	18.05	0.12
DG-RRT	37.96	37.58	49.69	34.52	0.87	0.17	13.06	0.01

The effect plots shown in Figures 15 and 16 clearly show that the trajectories obtained by Q-RRT* and PF-RRT planning are smoother compared to RRT and RRT*. Based on the c_{mean} and t_{mean} data in Tables 1 and 2, it is obvious that PF-RRT has more advantages in both path quality and search time. In addition, comparing the time extremes t_{max} and t_{min} in the statistics of Tables 1 and 2 with the path acquisition time in two dimensions, it is found that the time span for path search in three dimensions is larger and the uncertainty is higher.



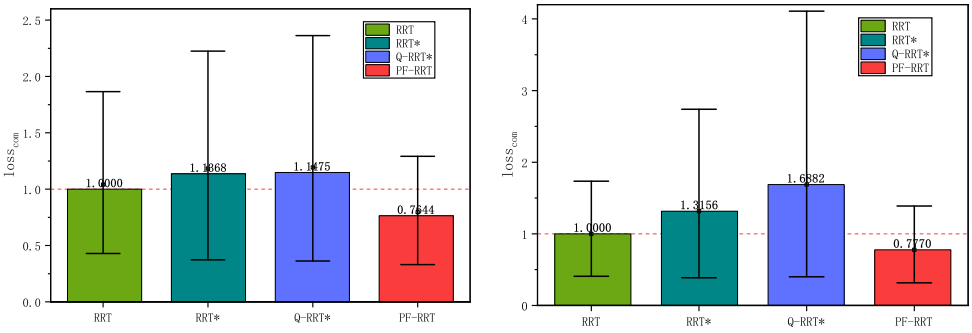
(**a**) (**b**)

Figure 16. Trajectory comparison chart of 3D-map2: (**a**) stereogram; and (**b**) top view.

To facilitate comparison of the combined efficiency of the four algorithms, the relative path-cost loss rate $l_{c(i)}$ and relative time loss rate $l_{t(i)}$ are calculated for each algorithm in the two maps, respectively, as shown in Equation (15), and the combined loss rate is calculated by assigning weights μ and ω.

$$\begin{cases} L_{com}(i) = \mu \cdot l_{c(i)} + \omega \cdot l_{t(i)} \\ \mu + \omega = 1 \end{cases} \tag{15}$$

Here, the weight of $l_{c(i)}$ and $l_{t(i)}$ are set to 1:1, and by calculating the results according to Equation (15), the combined loss rate and its regional distribution of different algorithms in 3D-map1 and 3D-map2 are obtained, as shown in Figure 17a,b, respectively. The results show a very good improvement in the overall performance of PF-RRT in 3D space, and the integrated loss rate is only 77.7% compared with the original RRT. At the same time, the error bars indicate that the results of PF-RRT are more stable.



(**a**) (**b**)

Figure 17. Distribution map of L_{com}: (**a**) 3D-map1; and (**b**) 3D-map2.

Figure 18a,b show the four algorithms in 3D-map1 and 3D-map2, respectively, in the same time span. From Figure 18, it can be seen that only Q-RRT* and PF-RRT in 3D space have the optimal convergence characteristics and can converge to $1.05C_{optimal}$. The average time taken by PF-RRT to plan suboptimal paths in the two maps is 1.50 s and 1.75 s, respectively. Meanwhile, Q-RRT* not only has a worse initial result performance than PF-RRT, but also has a more limited convergence.

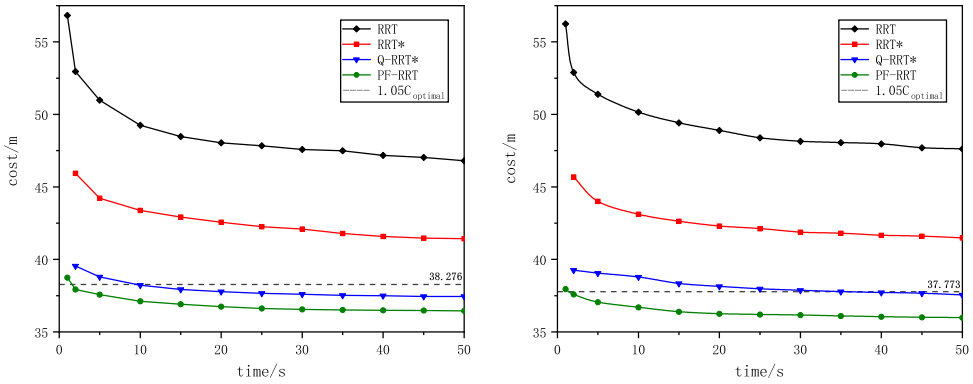


Figure 18. Optimal convergence diagram in three-dimensional environment: (**a**) 3D-map1; and (**b**) 3D-map2.

4.4. 3D Simulation and Physical Flight

Based on the performance testing map, a realistic 3D scene was built here for simulation testing to verify the effectiveness of the algorithm. The simulation model of PX4 UAV was loaded into the ROS system under Ubuntu environment, a simple UAV flight environment scenario was set up by Gazebo9. Finally, the PF-RRT path-planning program is loaded into the simulated UAV and simulated flight tests are performed. The simulation experiment senses the environment using LIDAR and visualizes the environment and flight trajectory using Rviz.

PF-RRT is used as the global path-planning algorithm in the flight test, and the test results are shown in Figure 19a–c, where Figure 19a is the PX4 UAV simulation model, and the map shown in Figure 19b is built for the flight test, and finally, the UAV's travel trajectory is marked and displayed in Rviz as shown in Figure 19c. In the simulation flight, the planned path is updated by refreshing the current position of no earth in real time and using PF-RRT until the UAV approaches the target point and finally obtains the complete flight trajectory. The output trajectory is smoother, as shown in Figure 19c, and the shorter time required for PF-RRT to obtain the initial solution makes the refresh frequency of real-time path planning higher.

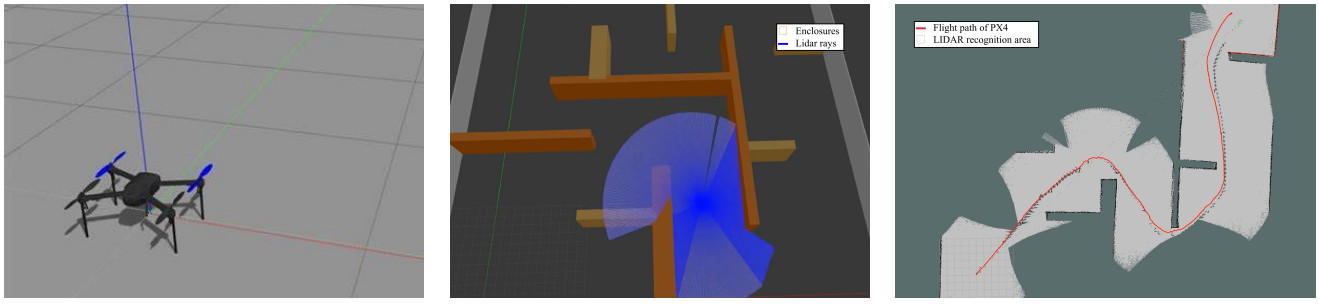


Figure 19. Simulation in ROS: (**a**) the model of PX4; (**b**) simulation map; and (**c**) flight path.

The current UAV pilot study is based on 'red track' rail fault detection. The test site is the red track test line of Jiangxi University of Technology Golden Campus. The UAV selected is PX4 equipped with 2D LIDAR and lntel T265 camera. The purpose of the test to be achieved is as follows:

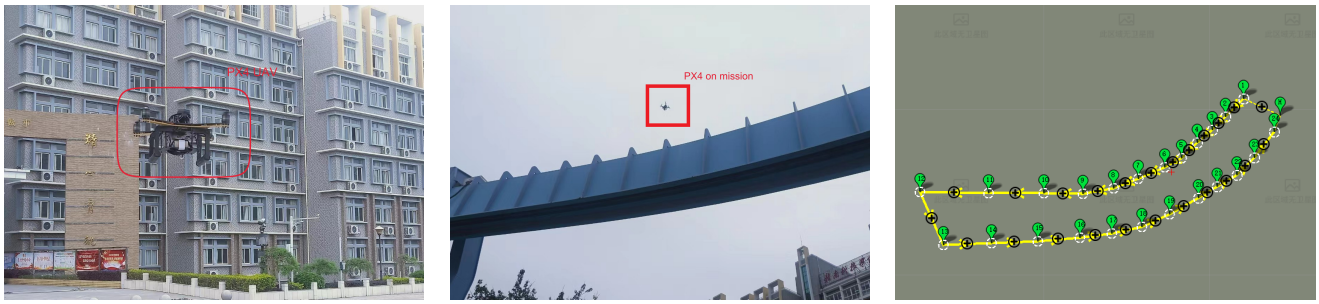
I. Accurate arrival at the designated detection point;

II. Planning a no-touch path.

In the actual flight, PF-RRT is used as a global path-planning algorithm and an initial unobstructed path is obtained. The local path-planning algorithm then uses a dynamic

windowing algorithm to prevent other irresistible force factors, such as birds, maintenance personnel, and trains running on the track.

Figure 20a–c show the actual flight effect of the UAV, where Figure 20a shows the PX4 UAV. Figure 20b shows the UAV inspecting the suspended track of the 'red track', and Figure 20c shows the UAV flight trajectory shown by the ground station. At present, it has been initially achieved by drones to monitor the track at fixed points. The preliminary test results prove that it is also feasible to use PF-RRT for path planning of UAVs and to apply UAVs to spot detection, cruising, and tracking in complex environments in the field.



(**a**) (**b**) (**c**)

Figure 20. Physical flight testing: (**a**) PX4; (**b**) flying along an orbit; and (**c**) flight trajectory.

5. Conclusions

In this paper, an efficient path-planning algorithm, PF-RRT, is proposed based on RRT. It is based on the idea of preprocessing the map by constructing a potential field and constructing a greedy strategy oriented by the potential field to accelerate the extension of the path towards a better region, which makes the random tree search reach the target point quickly. In addition, the path quality is improved in path reading by performing one iteration of the root node for the new node. The experimental results show that PF-RRT performs better in both simple or complex environments, and in two-dimensional planes or three-dimensional spaces. PF-RRT is able to reduce the time loss by 20% to 70% and improve the path quality by about 25% compared to RRT in the process of obtaining the initial solution. PF-RRT not only obtains high-quality initial solutions quickly, but also has better convergence properties. UAV real-time path planning is more responsive, by using the PF-RRT algorithm to quickly obtain feasible paths in high-dimensional environments. The feasibility of UAV path planning using PF-RRT was verified by simulation flight and actual flight testing.

In addition, PF-RRT is a tree extension algorithm, and the greedy strategy based on potential-field orientation can be combined with any other random tree optimization algorithm to further improve the performance. The performance of the current algorithm for UAV path planning based on PF-RRT and the feasibility of its actual flight have been initially verified, but there are several directions for further research, as follows:

I. The proposed PF-RRT algorithm has a great advantage over other algorithms in obtaining the initial solution in a two-dimensional environment, and further explores how to ensure that its advantage is not reduced when performing a path search in three-dimensional space;

II. Applying 3D LIDAR to current UAVs to enhance their environmental awareness and to adapt to interference from dynamic obstacles;

III. The incorporation of cameras and machine vision technology in UAVs enables them to perform fault detection in unmanned conditions in complex environments.

Author Contributions: Funding acquisition, project administration, supervision, K.F.; methodology, investigation, writing—original draft, data curation, T.H.; writing—review and editing, W.S.; investigation, W.L.; visualization, H.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (No. 61763018), the Central Guided Local Science and Technology Funding Project of the Science and Technology Department of Jiangxi Province (Cross-regional Cooperation, 20221ZDH04052), the China Scholarship Council (CSC, No. 201708360150), the Program of Qingjiang Excellent Young Talents in Jiangxi University of Science and Technology (JXUSTQJBJ2019004), and the cultivation project of the State Key Laboratory of Green Development and High-Value Utilization of Ionic Rare-Earth Resources at Jiangxi Province (20194AFD44003), the Key Research and Development Plan of Ganzhou (industrial field), the Science and Technology Innovation Talent Project of Ganzhou, and a grant from the Research Projects of Ganjiang Innovation Academy, Chinese Academy of Sciences (No. E255J001).

Data Availability Statement: Data are only available upon request due to restrictions regarding, e.g., privacy and ethics. The data presented in this study are available from the corresponding author upon request. The data are not publicly available due to their relation to other ongoing research.

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

References

1. Chamola, V.; Kotesh, P.; Agarwal, A.; Gupta, N.; Guizani, M. A comprehensive review of unmanned aerial vehicle attacks and neutralization techniques. *Ad Hoc Netw.* **2021**, *111*, 102324. [CrossRef] [PubMed]
2. Ha, S.W.; Paul, Q.; Moon, Y.H. Improvement of UAV attitude information estimation performance using image processing and kalman filter. *J. Converg. Inf. Technol.* **2018**, *8*, 135–142.
3. Evers, L.; Dollevoet, T.; Barros, A.I.; Monsuur, H. Robust UAV mission planning. *Ann. Oper. Res.* **2014**, *222*, 293–315. [CrossRef]
4. Wang, Z.; Yue, P. Marine Island UAV Aerial Photography: A Path-Planning Algorithm-Based Study. *J. Coast. Res.* **2020**, *106*, 642–645.
5. Wang, S.; Xu, S.; Yu, C.; Wu, H.; Liu, Q.; Liu, D.; Jin, L.; Zheng, Y.; Song, J.; He, X. Obstacle Avoidance and Profile Ground Flight Test and Analysis for Plant Protection UAV. *Drones* **2022**, *6*, 125. [CrossRef]
6. Li, J.; Liu, H.; Lai, K.K.; Ram, B. Vehicle and UAV Collaborative Delivery Path Optimization Model. *Mathematics* **2022**, *10*, 3744. [CrossRef]
7. Papadopoulou, E.E.; Vasilakos, C.; Zouros, N.; Soulakellis, N. DEM-Based UAV Flight Planning for 3D Mapping of Geosites: The Case of Olympus Tectonic Window, Lesvos, Greece. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 535. [CrossRef]
8. Sun, W.; Fan, K.; Huang, T.; Cui, J. Path Optimization of UAV Inspection Suspended Track Based on Dynamic Adaptive Ant Colony Optimization. In Proceedings of the 2022 6th International Conference on Automation, Control and Robots (ICACR), Shanghai, China, 23–25 September 2022; pp. 142–147.
9. Samar, R.; Rehman, A. Autonomous terrain-following for unmanned air vehicles. *Mechatronics* **2011**, *21*, 844–860. [CrossRef]
10. Stentz, A. Optimal and efficient path planning for partially known environments. In *Intelligent Unmanned Ground Vehicles: Autonomous Navigation Research at Carnegie Mellon*; Springer: Berlin/Heidelberg, Germany, 1997; pp. 203–220.
11. Daniel, K.; Nash, A.; Koenig, S.; Felner, A. Theta*: Any-angle path planning on grids. *J. Artif. Intell. Res.* **2010**, *39*, 533–579. [CrossRef]
12. Ferguson, D.; Stentz, A. Field D*: An interpolation-based path planner and replanner. In *Robotics Research: Results of the 12th International Symposium ISRR*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 239–253.
13. Tu, J.; Yang, S.X. Genetic algorithm based path planning for a mobile robot. In Proceedings of the 2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422), Taipei, Taiwan, 14–19 September 2003; Volume 1, pp. 1221–1226.
14. Dorigo, M.; Maniezzo, V.; Colorni, A. Ant system: Optimization by a colony of cooperating agents. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **1996**, *26*, 29–41. [CrossRef] [PubMed]
15. Oleiwi, B.K.; Roth, H.; Kazem, B.I. A hybrid approach based on ACO and GA for multi objective mobile robot path planning. *Appl. Mech. Mater.* **2014**, *527*, 203–212. [CrossRef]
16. Kennedy, J.; Eberhart, R.C. A discrete binary version of the particle swarm algorithm. In Proceedings of the 1997 IEEE International Conference on Systems, Man, and Cybernetics, Computational Cybernetics and Simulation, Orlando, FL, USA, 12–15 October 1997; Volume 5, pp. 4104–4108.
17. Khuswendi, T.; Hindersah, H.; Adiprawita, W. Uav path planning using potential field and modified receding horizon A* 3D algorithm. In Proceedings of the 2011 International Conference on Electrical Engineering and Informatics, Bandung, Indonesia, 17–19 July 2011; pp. 1–6.
18. Saravanakumar, S.; Asokan, T. Multipoint potential field method for path planning of autonomous underwater vehicles in 3D space. *Intell. Serv. Robot.* **2013**, *6*, 211–224. [CrossRef]
19. Raja, R.; Dutta, A.; Venkatesh, K.S. New potential field method for rough terrain path planning using genetic algorithm for a 6-wheel rover. *Robot. Auton. Syst.* **2015**, *72*, 295–306. [CrossRef]
20. LaValle, S.M. Rapidly-exploring random trees: A new tool for path planning. In *The Annual Research Report*; Iowa State University: Ames, IA, USA, 1998.
21. Kavraki, L.E.; Kolountzakis, M.N.; Latombe, J.C. Analysis of probabilistic roadmaps for path planning. *IEEE Trans. Robot. Autom.* **1998**, *14*, 166–171. [CrossRef]

22. Kuffner, J. RRT-Connect: An efficient approach to single-query path planning. In Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, CA, USA, 24–28 April 2000; pp. 473–479.
23. Karaman, S.; Walter, M.R.; Perez, A.; Frazzoli, E.; Teller, S. Anytime motion planning using the RRT. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 1478–1483.
24. Jeong, I.B.; Lee, S.J.; Kim, J.H. Quick-RRT*: Triangular inequality-based implementation of RRT* with improved initial solution and convergence rate. *Expert Syst. Appl.* **2019**, *123*, 82–90. [CrossRef]
25. Urmson, C.; Simmons, R. Approaches for heuristically biasing RRT growth. In Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No. 03CH37453), Las Vegas, NV, USA, 27–31 October 2003; Volume 2, pp. 1178–1183.
26. 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.
27. Li, Y.; Wei, W.; Gao, Y.; Wang, D.; Fan, Z. PQ-RRT*: An improved path planning algorithm for mobile robots. *Expert Syst. Appl.* **2020**, *152*, 113425. [CrossRef]
28. Liao, B.; Wan, F.; Hua, Y.; Ma, R.; Zhu, S.; Qing, X. F-RRT*: An improved path planning algorithm with improved initial solution and convergence rate. *Expert Syst. Appl.* **2021**, *184*, 115457. [CrossRef]
29. Guo, Y.; Liu, X.; Jiang, W.; Zhang, W. Collision-Free 4D Dynamic Path Planning for Multiple UAVs Based on Dynamic Priority RRT* and Artificial Potential Field. *Drones* **2023**, *7*, 180. [CrossRef]