


Article

# A Disaster Relief UAV Path Planning Based on APF-IRRT\* Fusion Algorithm

Qifeng Diao <sup>1,\*</sup>, Jinfeng Zhang <sup>1</sup>, Min Liu <sup>2</sup> and Jiaxuan Yang <sup>1</sup>

<sup>1</sup> Research Center of Fluid Machinery Engineering and Technology of Jiangsu University, Zhenjiang 212013, China

<sup>2</sup> Institute of Fluid Engineering Equipment, JITRI, Zhenjiang 212009, China

\* Correspondence: 2222111034@stmail.ujs.edu.cn

**Abstract:** Unmanned Aerial Vehicle (UAV) path planning has increasingly become the key research point for civilian drones to expand their use and enhance their work efficiency. Focusing on offline derivative algorithms, represented by Rapidly-exploring Random Trees (RRT), are widely utilized due to their high computational efficiency. However, deploying these offline algorithms in complex and changing disaster environments presents its own drawbacks, such as slow convergence speed, poor real-time performance, and uneven generation paths. In this paper, the Artificial Potential Field -Improved Rapidly-exploring Random Trees (APF-IRRT\*) path-planning algorithm is proposed, which is applicable to disaster relief UAV cruises. The RRT\* algorithm is adapted with adaptive step size and adaptive search range coupled with the APF algorithm for final path-cutting optimization. This algorithm guarantees computational efficiency while giving the target directivity of the extended nodes. Furthermore, this algorithm achieves remarkable progress in solving problems of slow convergence speed and unsmooth path in the UAV path planning and achieves good performance in both offline static and online dynamic environment path planning.

**Keywords:** Rapidly-exploring Random Trees algorithm; Artificial Potential Field algorithm; UAV path planning



**Citation:** Diao, Q.; Zhang, J.; Liu, M.; Yang, J. A Disaster Relief UAV Path Planning Based on APF-IRRT\* Fusion Algorithm. *Drones* **2023**, *7*, 323. <https://doi.org/10.3390/drones7050323>

Academic Editor: Diego González-Aguilera

Received: 2 April 2023  
Revised: 16 May 2023  
Accepted: 17 May 2023  
Published: 18 May 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Unmanned Aerial Vehicles (UAVs), notably employed as indispensable intelligent electronic devices in modern life, are showing continuous growth in the proportion of the global trade market. Civilian drones can be divided into patrol drones, agricultural drones, meteorological drones, exploration drones, and disaster relief drones according to their uses. Based on this, UAVs with higher integration, better computing efficiency, and faster reaction time have been invested immensely in research and development by countries around the world [1,2]. However, due to the unpredictability of the disaster area (such as severe weather, disaster areas, etc.), depending solely on radio remote control, is very challenging for UAVs to accomplish tasks such as precise identification, obstacle avoidance flight, and coordinated rescue [3]. Therefore, it is of great essence to develop algorithms for autonomous path planning of disaster relief UAVs.

Disaster relief UAVs generally fly at altitudes between 25 and 150 m above sea level. Signal interference caused by low-altitude clouds, smoke, and dust is the main natural threat to drones when flying [4], which means that UAVs should avoid being disrupted by clouds, smoke, and dust. Therefore, the path-planning algorithm deployed on the UAV has to be efficient, safe, and equipped with local dynamic planning capabilities.

Currently, the existing path-planning algorithms can be divided into two categories: the global path-planning algorithm and the local path-planning algorithm. The global path-planning methods include the A\* algorithm [5], the Dijkstra algorithm [6], the Genetic Algorithm (GA) [7], and the sampling-based Rapidly-exploring Random Trees (RRT) algorithm [8]. The drawback of the heuristic algorithm represented by the A\* algorithm is that it

causes significant performance consumption when the target point is unreachable, which is unsuitable for complex disaster environments. The original version of Dijkstra's algorithm was only suitable for finding the shortest path between two vertices, later more common variants fixed a vertex as the source node and then found the shortest path from that vertex to all other nodes in the graph, yielding a shortest path tree. This algorithm picks out the smallest distance among the unvisited nodes each time and uses this node to update the distances of other nodes. The GA algorithm can obtain the optimal path in route planning, but it requires multiple iterations to obtain the optimal solution. Its complex algorithm leads to low computational efficiency, poor real-time performance, and the inability to use feedback information from the network in a timely manner.

In disaster areas, the path-planning algorithm needs to prioritize safety and generate the speed of the UAV moving trajectory not just for the optimal path. Therefore, the RRT algorithm with faster execution efficiency is introduced to generate UAV motion paths quickly. Compared with the RRT algorithm, after the sampling point is added to the path tree, the RRT\* algorithm draws a circle with the new point as the center, and considers whether there is a better parent node, in other words, which nodes can be connected to make the distance from the starting point to the target point shorter (although those nodes are not the closest points to the sampling point). If a more suitable parent node is selected, it then connects them and removes the original connection, which is called rerouting. Karaman et al. [9] proposed a progressive optimal path based on the RRT\* algorithm to guide the target. Qin et al. [10] put forward a posture constraint strategy which enables UAVs to fly more smoothly.

The RRT\* algorithm takes the Artificial Potential Field (APF) algorithm [11] and Dynamic Window Algorithm (DWA) [12] as the typical cases of local path-planning algorithms. The former is widely used in path-smoothing control, while the latter possess relatively high dynamic obstacle avoidance capability. However, the adaptive direction of the existing algorithms is mainly centered around the step size. Most of the optimization work is premised on the characteristics of the algorithms themselves, with less research on algorithm fusion.

In the field of global path planning of UAVs, most of them are static, and the planned path is obtained after the environment is rasterized, which is not a real optimized path, and thus further optimization is still necessitated. Nonetheless, for rescue and disaster relief missions, whether they are reconnaissance or transportation missions, not only must the most basic guarantees be made in terms of path safety, but relatively high requirements are also required for running time. The improved fusion algorithm proposed in this paper possesses an enhanced performance in both dynamic and static obstacle avoidance environments. It is not only satisfied with static obstacle avoidance but also has good safety performance when encountering moving obstacles. Moreover, although the required computing time has been shortened a little, the heading angle has improved a lot, and more importantly, it is suitable for complex and changeable real environments.

From the aforementioned problems, this paper addresses them by improving the RRT\* algorithm in the following four ways:

- (a) Employing adaptive step size, which switches to a small and dynamic step size when approaching obstacles, achieves a step size that changes with path conditions, and improves path smoothness to some extent.
- (b) Employing adaptive search scope, which gives a dynamic constraint to the search scope, effectively reduces the randomness and blindness of node growth in the search process, speeds up the convergence speed, and improves efficiency.
- (c) Combining with the APF algorithm. Attraction and repulsion forces are introduced based on expanded random numbers to expand the target point more effectively under the combined force. This algorithm improves operational efficiency, reduces the turning angle and path cost, and has a safe and efficient real-time obstacle avoidance capability.

- (d) Cutting and optimizing the path. A novel collision detection method is introduced. If the collision detection between  $Q_1$  and  $Q_2$  is passed, and there is no collision with the obstacle, the node is banned, the number of turns is reduced, which keeps the path as smooth as possible.

After the improvement of the RRT\* algorithm and the integration of the APF algorithm, the convergence speed and path smoothness of the new algorithm APF-IRRT\* in both offline static and online dynamic environments are improved, the real-time performance is greatly improved, and the practical implementation is enhanced.

In light of the above discussions, this paper seeks to propose a novel fusion algorithm capable of performing in both dynamic and static obstacle avoidance environments. This algorithm is not only satisfied with static obstacle avoidance, but also has good safety performance when encountering moving obstacles. In this way, the required internal calculation time can be shortened, and its maximum deflection angle can be controlled, whilst improving the rescue speed, and its safety has been greatly improved, which is more suitable for complex and changeable rescue environments.

The remainder of the article is organized as follows: Section 2 generalizes the path planning problem into a mathematical model; Section 3 shows the basic principles of RRT, RRT\*, and APF algorithms; Section 4 describes the details of the method proposed in this paper; Section 5 shows the overall technical route and the acquired data in this paper; Section 6 presents the detailed results of different experiments, then discusses the results of the test phases; finally, the conclusions are presented in Section 7.

## 2. Problem Description

Firstly, the horizontal area of an altitude is supposed to be the entire task space  $Z$ , where  $Z$  represents the state space of the entire system simultaneously [13].  $Z_{free} \subset Z$  indicates a safe area for flying, and  $Z_{obs} \subset Z$  indicates a hazardous area with obstacles, expressed as:

$$Z_{free} \cup Z_{obs} = Z \text{ and } Z_{free} \cap Z_{obs} = \emptyset \quad (1)$$

where  $x_{star} \in Z_{free}$  indicates UAV initial position range;  $x_{goal} \in Z_{free}$  indicates target position range. Then, the dynamic equation of UAV can be expressed by the following equation:

$$\dot{x}(t) = \Gamma(x(t), h(t)), \quad x(0) = x_0 \quad (2)$$

where  $x(t) \in Z$  is the state system;  $h(t) \in H$  is the control input of the system, where the control input set of the system is represented by  $H$ ; and  $\Gamma$  is described as the function of the UAV system model built. When  $t = 0$ , the initial state of the UAV is  $x_0 \in X_{free}$ .

## 3. Basic Principles of Algorithm

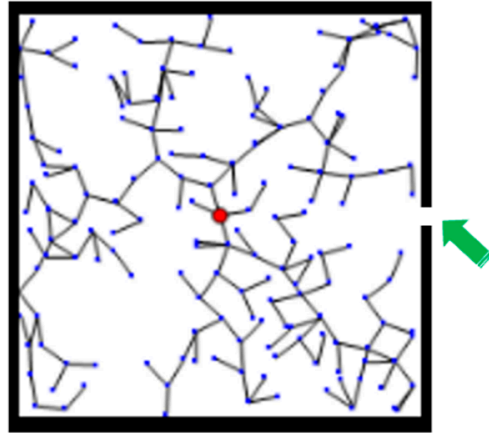
### 3.1. Traditional RRT Algorithm

The path planning problem involves finding a collision-free path from the starting point to the target point within the specified area. It can be classified into global path planning and local path planning. The former is the original path planning problem and involves searching for an optimal path throughout the entire space.

The RRT algorithm, proposed in the late 20th century, is particularly noteworthy for its superior performance of speed and flexibility in many scenarios [14]. However, it also presents the disadvantage of fulling of randomness in its search process, leading to its unstable efficiency.

Furthermore, the lack of target directivity in the iterative process also makes the search paths different and leads to randomness. As shown in Figure 1, it can be seen that the RRT algorithm struggles to find the exit in the closed terrain with a narrow channel, resulting in excessive computation and time consumption. Although the above problems can be improved by changing the step size or setting the adaptive step size, reducing the step size will lead to a tremendous amount of calculation and a longer calculation time. Additionally,

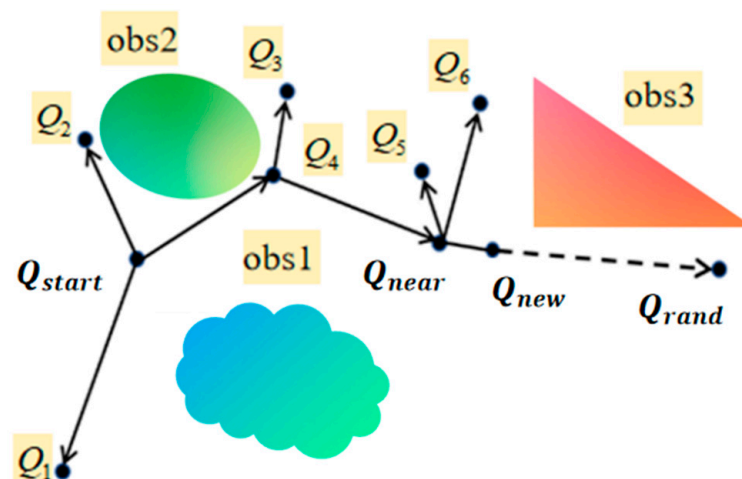
some improved RRT algorithms try to make up for some defects by interpolation correction yet fail to obtain a satisfactory effect.



**Figure 1.** Lacks practicality in scenarios where borrow aisles exist.

The RRT algorithm initializes the map with only the root starting point  $Q_{start}$  and target point  $Q_{goal}$ . A random point  $Q_{rand}$  is then generated according to a search strategy probability,  $P$ , which may either take  $Q_{goal}$  as  $Q_{rand}$  or scatter  $Q_{rand}$  within a specific range of  $Q_{goal}$ . The node closest to the  $Q_{rand}$  on the map and the starting point of the  $Q_{start}$  above process is named  $Q_{near}$ . A new node  $Q_{new}$  is then generated in the direction of the vector pointing from  $Q_{near}$  to  $Q_{rand}$ , with the step length represented by the distance parameter,  $step$ . The value of  $step$  is crucial to the algorithm's accuracy, and careful selection is a top priority.

If there exists a potential collision between  $Q_{new}$  and the parent node  $Q_{near}$ , the path will be abandoned, and a new node will be generated. The loop continues until the distance between  $Q_{new}$  and the  $Q_{goal}$  is less than  $\epsilon$ , where  $\epsilon$  is used to judge whether it is close enough to the target point or reaches the target point. Figure 2 shows the schematic diagram of the RRT algorithm.



**Figure 2.** Schematic diagram of the RRT algorithm.

Obstacles are an array of artificial inputs. The detection of obstacles adopts a hypothetical idea, first selects a certain area, and stipulates that all sampling points falling in this area are invalid and will not be included in the growth tree. In other words, the selected area can be equivalent to a “hole”, and the sampling points falling in the “hole” will not be regarded as  $Q_{rand}$  and will not participate in the above judgment process. In this way, the generated path will bypass the area, and there will be no intersection at the edge or

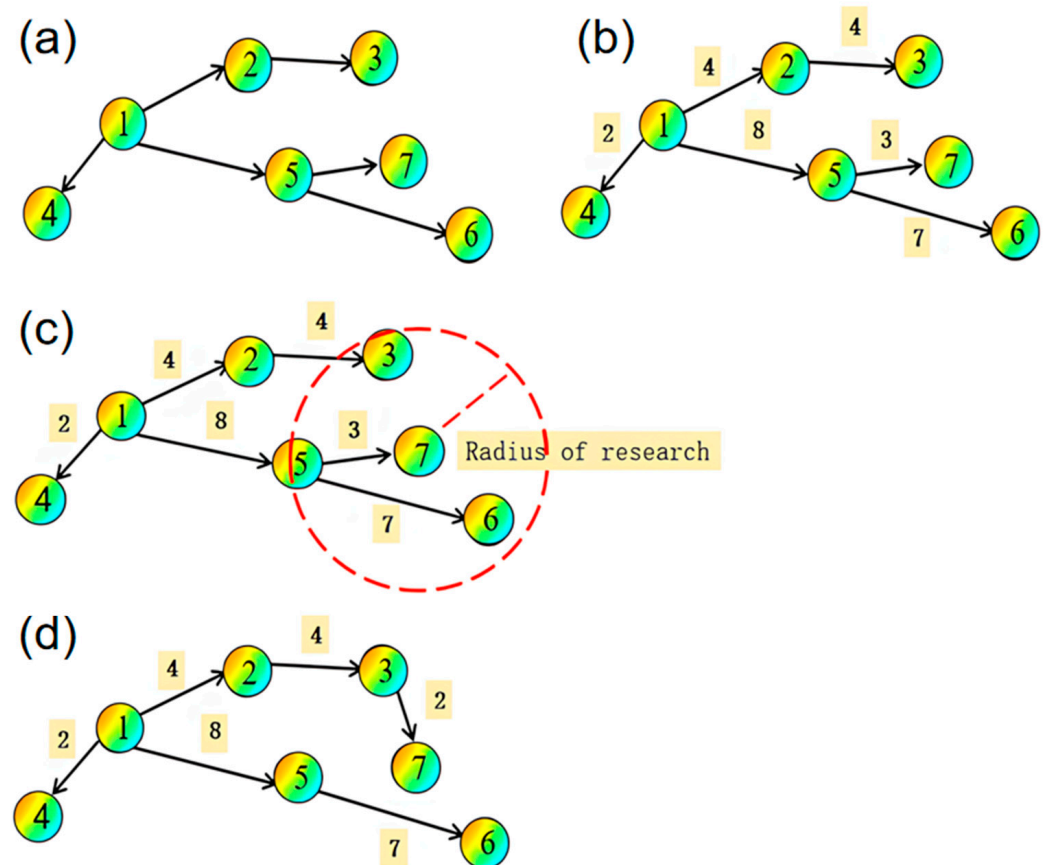
inside of the area, so the area is equivalently regarded as an obstacle and the obstacles can be detected in real-time.

To ensure the feasibility and controllability of the algorithm, a maximum search count limit `COUNT_MAX` needs to be set in advance to complete the search within the given number of cycles. Otherwise, it is set to jump out of the cycle to end the task.

Therefore, the RRT algorithm encompasses six basic steps: import map information, generate random points, extend towards the generated point, add collision detection, set the cycle limit, and use the greedy algorithm to find the shortest path.

### 3.2. The RRT\* Algorithm

The RRT\* algorithm is an improvement on the traditional RRT algorithm. Instead of immediately starting the next cycle after generating a new node  $Q_{new}$ , it reselects the parent node and reroutes the new node through collision detection. As is shown in Figure 3, the round icon represents the node, and the label on the node represents the order of node generation.



**Figure 3.** Resetting parent node and rerouting process of the RRT\* algorithm. (a) Extended random tree; (b) assign weights to the tree; (c) reselect the parent node; (d) reconnect the nodes.

The improved performance of the RRT\* algorithm can be seen by adding weights to the directed weight graph. The innovative step reset and reroute the parent node is demonstrated in Figure 3b. By finding a path with the least cost from  $Q_{start}$  to the latest node  $Q_{new}$ , the algorithm resets the parent node. The threshold range for reselecting the parent node is a circle with the new node  $Q_{new}$  as the center and the parameter  $r$  as the radius.

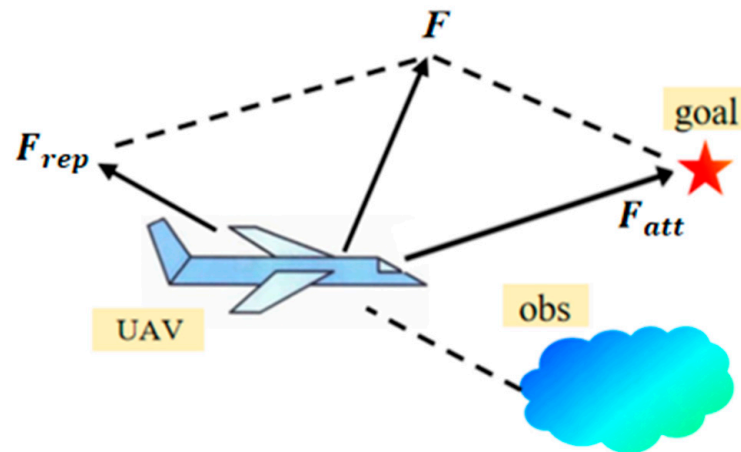
Any node in the circle is regarded as a potential parent node and compared with the original parent node to see if the path cost is lower. As shown in Figure 3c, node 3 is chosen as the new parent node for node 7, and the connection between the original node 6 and node 7 is erased. The algorithm then generates a new search tree.

To sum up, the RRT\* algorithm improves the way of selecting parent nodes based on the original RRT algorithm. The cost function determines the node with the lowest cost in expanding nodes as the parent node. At the same time, nodes on the existing tree will be reconnected after each iteration to ensure computational complexity and an asymptotic optimal solution.

### 3.3. The APF Algorithm

Because the working environment of UAVs is highly complex, even with access to global map information, path planning remains challenging. A crucial aspect is moving towards the target position away from the obstacles, which necessitates the construction of a potential field function that incorporates these effects.

Because of the directivity of the UAV to the target point, the target point provides attraction, whereas obstacles elicit repulsion. The potential energy of any point in the map can be obtained by superimposing the attractive and repulsive potential fields. Path planning is accomplished by searching for and moving toward potential energy reduction. The working principle is shown in Figure 4, where  $F_{rep}$  indicates the repulsion of obstacles to the UAV, and  $F_{att}$  indicates the attraction of target points to the UAV.



**Figure 4.** Schematic diagram of the APF algorithm.

The algorithm stipulates that the UAV will be subjected to the attraction of the target point at any position in the map, and the attraction is inversely proportional to the distance of the target point [15].  $U_{att}(q)$  is defined as an attractive potential field function:

$$U_{att}(q) = \frac{1}{2}\zeta dist^2(q, q_{goal}) \quad (3)$$

where  $q$  is the current position of the UAV,  $q_{goal}$  is the position of the target point,  $\zeta$  is the scale coefficient of the attractive potential field, and  $dist(q, q_{goal})$  represents the Euclidean distance between the UAV and the target point. The gradient is expressed in  $\nabla U_{att}(q)$ .

Correspondingly,  $U_{rep}(q)$  is defined as the repulsive potential field function:

$$U_{rep}(q) = \frac{1}{2}\eta \left( \frac{1}{dist(q, q_{obs})} - \frac{1}{dist^*} \right) \quad (4)$$

where  $\eta$  is the scale coefficient of the potential repulsion field,  $dist(q, q_{obs})$  represents the distance between UAV and the obstacle, and  $dist^*$  ( $d^*$ ) is the repulsion field range of the obstacle. If  $dist(q, q_{obs}) > dist^*$ , then the potential repulsion field  $U_{rep}(q) = 0$ . The gradient is expressed in  $\nabla U_{rep}(q)$ .

Therefore, the total potential field function of UAV is the superposition of attractive potential field and repulsive potential field, known as

$$U(q) = U_{att}(q) + \sum_1^n U_{rep}(q) \quad (5)$$

The total gradient is expressed as

$$\nabla U(q) = \nabla U_{att}(q) + \nabla U_{rep}(q) \quad (6)$$

The resultant force is expressed as

$$F(q) = -\nabla U(q) \quad (7)$$

#### 4. Algorithm Improvement

The RRT\* algorithm distributes numerous extraneous points during global sampling to enhance its search's randomness, occupying significant memory space and eventually leading to slow convergence and overly convoluted paths. On the other hand, in the artificial potential field method, the complexity, danger, and particularity of the environment may result in an abundance of points with the lowest energy in the map or even positions where the attractive and repulsive forces cancel each other out. This can make the UAV become trapped in local optimization or stagnation. Therefore, to address these issues, this paper proposes an improvement to the RRT\* algorithm by combining it with the APF algorithm and integrating the ideas of attraction and repulsion to make the algorithm more sensitive and efficient.

##### 4.1. Adaptive Search Step

Although the RRT\* algorithm has a few improvements relative to the traditional RRT algorithm, the algorithm hardly determines the optimal path due to the fixed expansion step size. Therefore, the dynamic adaptive step size is adopted in this paper. The basic principle is: when  $Q_{new}$  approaches obstacles, the small step size is adopted for expansion; in contrast, when it is far away from the obstacle, it can be expanded with a large step size. The dynamic step size can realize that the step size changes with the change of path conditions, which accurately solves the problem of practicality. It also improves the smoothness of the path to a certain extent, which is practical [16]. Meanwhile, the dynamic step is expressed as  $S_v$ :

$$S_v = \begin{cases} \left(1 + k \ln\left(\frac{r}{r_1}\right)\right) \times step & r \leq r_1 \\ step & r > r_1 \end{cases} \quad (8)$$

where  $k$  is the proportion constant which is used to control the adjustment range of step size,  $r$  and  $r_1$  are the actual and safe critical distance between  $Q_{new}$  and obstacles, respectively. It can be seen from the formula that when the actual distance  $r$  is less than the safe distance  $r_1$ , the step size will be reduced under the control of the proportional coefficient  $k$ , effectively realizing the mutual consideration between obstacle avoidance and smoothness. The latest node  $Q_{new}$  can be represented as follows:

$$Q_{new} = Q_{new} + S_v \cos\theta \quad (9)$$

##### 4.2. Adaptive Search Range

In the RRT\* algorithm, the sampling mechanism of the algorithm itself has not changed, and the problem of slow convergence still exists. Therefore, the angle-increasing sampling method is introduced based on the RRT algorithm. It is to retain the original random sampling of the whole map, give a constraint to the search scope, and limit the initial search scope to a particular angle space. Figure 5 shows the basic principle of this method, taking

the direction of the resultant force  $F$  of  $Q_{near}$  as the starting axis, counterclockwise as the positive direction, the initial angle  $\theta = 15^\circ$ , and the increment  $\Delta = 15^\circ$ .

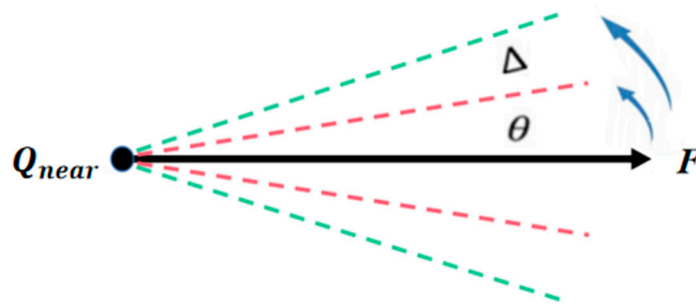


Figure 5. Schematic diagram of the angle-increasing sampling method.

The technique involves sampling within the range of  $\pm\theta$  around the direction of  $F$ , followed by collision detection on the connection between the new node  $Q_{new}$  and  $Q_{near}$ . If the detection fails, the  $\Delta$  is increased to the current angle  $\theta$ , and then the sampling and collision detection are performed again within the range of  $\pm(\theta + \Delta)$ . This process continues until  $(\theta + \Delta)$  falls within the range  $[0, \pi]$ . This method can effectively reduce the randomness and blindness of node growth in the search process, as well as accelerate the convergence speed and improve efficiency.

#### 4.3. Fusion with the APF Algorithm

To better solve the path planning problem of UAVs in emergency rescue and disaster relief, the traction force of target points on UAV and the repulsion force of obstacles in the APF algorithm are integrated into the RRT\* algorithm. The essence of the RRT\* algorithm is to expand the search tree, filter available nodes, and add them to the search tree. After the introduction of the APF algorithm, the target attraction function  $F_{att}(Q)$  and obstacle repulsion function  $F_{rep}(Q)$  are constructed for each node as a unit when the node is expanded so that the search tree can move towards the target point under the effect of the resultant force.

This approach significantly increases the search's purpose and target and speeds up the convergence speed [17]. The function expression of  $F_{att}(Q)$  and  $F_{rep}$  are as follows:

$$F_{att} = \zeta(Q_{goal} - Q_{near}) \tag{10}$$

$$F_{rep} = \eta \left( \frac{1}{d} - \frac{1}{d^*} \right) \frac{1}{d^2} \nabla d \tag{11}$$

where  $\zeta$  indicates the coefficient of attraction and  $\eta$  indicates the coefficient of repulsion.  $dist(Q_{near}, Q_{obs})$  is simplified by  $d$  to represent the distance between the drone and the obstacles. Equation (2) illustrates that the UAV is within the repulsive force range of the obstacle, that is,  $0 < d < d^*$  when  $d > d^*$ , the repulsion force is zero.

The resultant force of the new node is expressed as the vector sum of attractive and repulsive forces:

$$F = F_{att} + F_{rep} \tag{12}$$

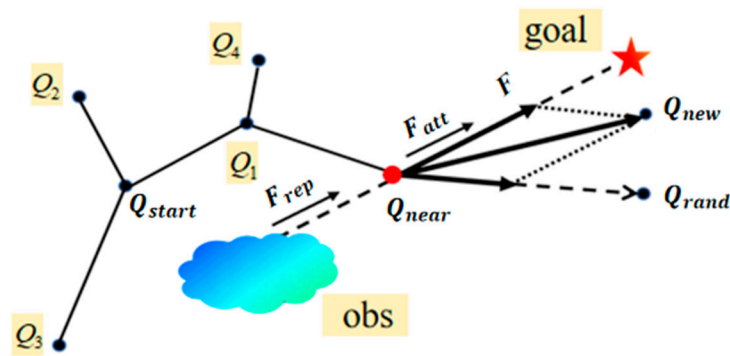
The generation of new nodes is also the superposition of two directed vectors:

$$Q_{new} = step \left( \frac{d}{\|d\|} + \frac{F}{\|F\|} \right) \tag{13}$$

However, in this case, it is often trapped in a local optimum issue for which we can add a disturbance (random walk) or backtrack at the local minimum to jump out of the local minimum.

The node expansion mode after the algorithm fusion is shown in Figure 6.





**Figure 6.** Schematic diagram of the fusion algorithm.

#### 4.4. Sectional Cutting Optimization Method

Although the APF-IRRT\* algorithm after algorithm fusion has relatively improved in sampling adjustment and convergence speed, the core mode of random sampling nevertheless remains unchanged, which leads to the path still having twists and turns. This is very unfavorable for UAVs in operations. Therefore, to circumvent the UAV from turning to the target point and causing problems such as too fast a speed or too large an angle and to reduce the tortuosity of the path, the original method of traversing from the starting point needs to be improved.

Since the obstacles in this paper are superimposed by circles, for the collision detection of a solid circle, it is only necessary to judge whether the distance between the vertex and the center of the circle is less than the radius. A new rerouting method is introduced after collision detection. Suppose the path goes through  $Q_1$ ,  $Q_2$ , and  $Q_3$ . If the collision detection between  $Q_1$  and  $Q_3$  is passed, and there is no collision with the obstacle, the node  $Q_2$  is banned, the number of turns is reduced, and the path is kept as smooth as possible. Secondly, the global path obtained by the fusion algorithm is cut, and the local path after cutting is optimized. In this process, it is necessary to connect the points into a line and calculate the steering angle;  $\gamma_{max}$  represents the max angle.

$$\gamma = \arctan\left(\frac{Y_{near} - Y_{new}}{X_{near} - X_{new}}\right) \quad (14)$$

### 5. Algorithm Flow

The steps of the new algorithm after fusing are as follows, and the flow chart is shown in Figure 7:

- (1) Parameter setting: initialize the coordinates of the starting point  $Q_{start}$  and the target point  $Q_{goal}$ , give the attractive coefficient  $\zeta$  and repulsion coefficient  $\eta$ , search  $step$ , and the value of the maximum number of samples COUNT\_MAX.
- (2) Randomly generate the sampling point  $Q_{rand}$ , calculate the resultant force direction of attraction and repulsion of  $Q_{near}$  after finding it, and then form a vector parallelogram in the direction of  $Q_{rand}$ . Calculate the direction of the new node  $Q_{new}$  and judge whether it is within the range of  $\theta$  and carry out collision detection. If it is within the range of  $\theta$  and does not pass through obstacles, go on to step (3); otherwise, increase the angle  $\Delta$  to continue searching. If no suitable  $Q_{new}$  is found within the range of  $[0, \pi]$ , give up the current random point  $Q_{rand}$  and continue searching for the next random point.
- (3) Search for potential parent nodes within a given range nearby  $Q_{new}$ , reselect the parent node, and reroute the line between  $Q_{new}$  and  $Q_{near}$ .
- (4) Judge whether  $dist(Q_{new}, Q_{goal})$  between  $Q_{new}$  and  $Q_{goal}$  is less than the given value. If it is less than that, the search is completed, and skip to step (5); otherwise, skip back to step (2) and select  $Q_{rand}$  again.

- (5) Connect the discrete points, generate the path and then divide it, smooth each path, and calculate whether the maximum turning angle  $\gamma_{max}$  meets the requirements. If yes, the final path is generated, and the algorithm ends and returns COUNT\_MAX and other parameters; if not, go to step (2) to select a new point.

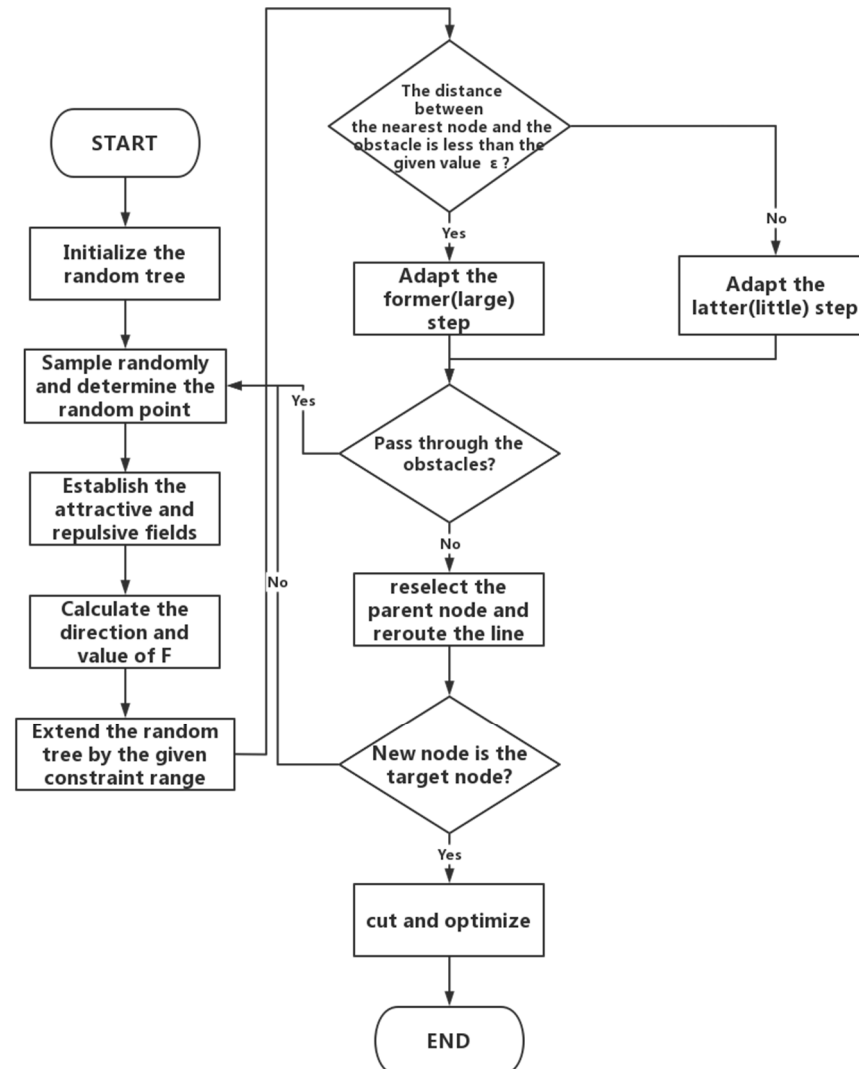


Figure 7. The APF-IRRT\* fusion algorithm flow chart.

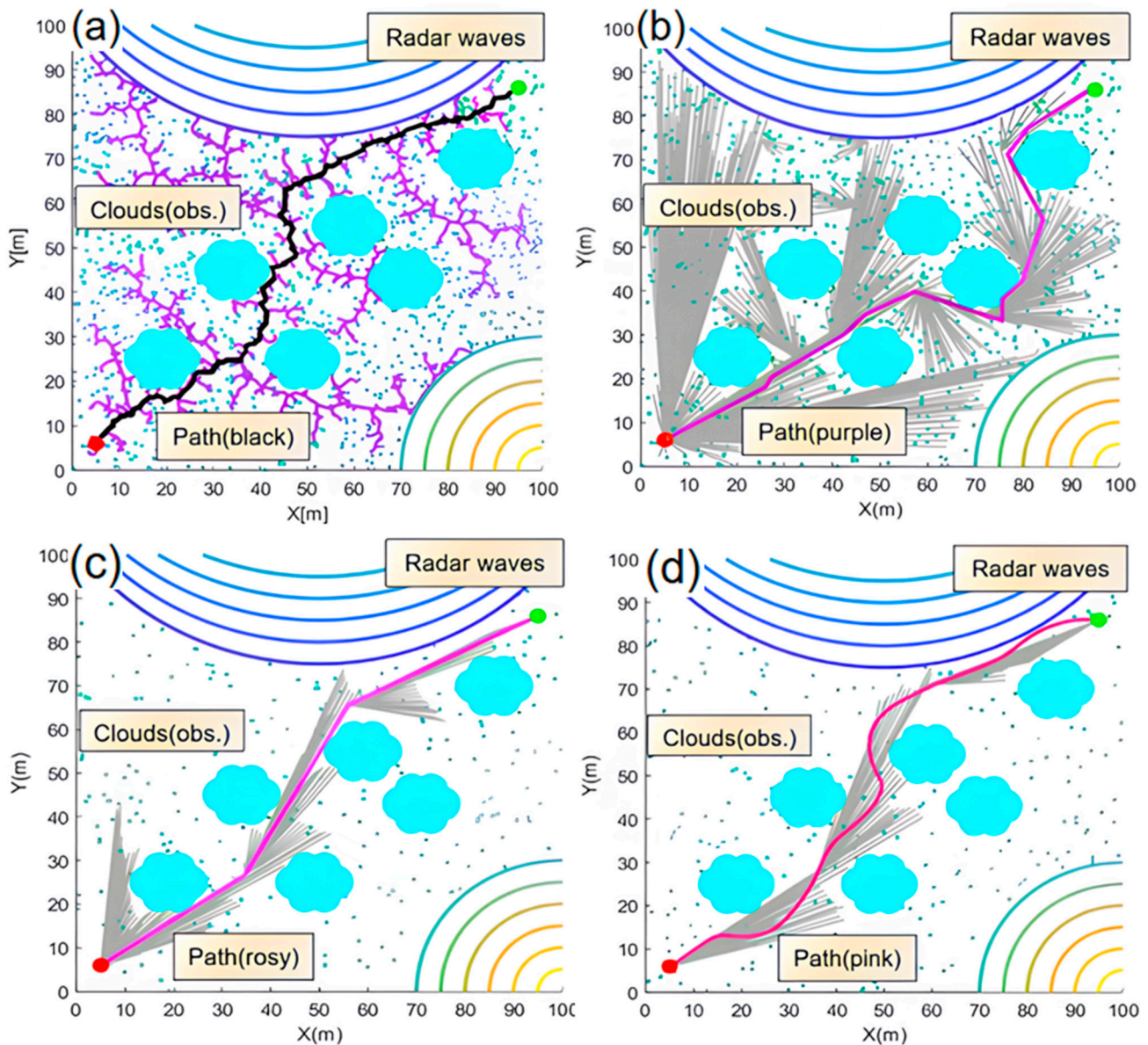
## 6. Algorithm Simulation and Analysis

To ascertain the performance of the improved fusion APF-IRRT\* algorithm, offline (static) and online (dynamic) path planning experiments were conducted under the  $100 \times 100$  m scale, simulated air obstacle map environment to obtain the final path map, time consuming, heading angles, angle of deflection, and other data. In offline planning, the obtained data are compared with the RRT algorithm, the RRT\* algorithm, and the improved RRT\* (IRRT\*) algorithm. Meanwhile, in the online planning, the data obtained are compared with the APF algorithm through the above comparison to verify its improvement of convergence speed and smoothness in static and dynamic environments, as well as better practicability.

### 6.1. Offline Path Planning Experiment

The algorithm in this paper is simulated with MATLAB 2022(b) software in the Window11 system environment. The hardware configuration of the experimental platform is CPU i9-10980XE, GPURT2080Ti, memory 128GB. First, establish a  $100 \times 100$  m simulation map and set parameters: the starting point coordinates are (5, 6), the target point coordinates are (95, 86),

the fixed search *step* is 0.5, the attractive coefficient  $\zeta = 1$ , the repulsive coefficient  $\eta = 5$ , the obstacle influence range  $dist^* = 3$ , the initial value of the sampling range  $0 < \theta < 15^\circ$ , the maximum number of samples  $COUNT\_MAX = 30,000$ , and the adaptive coefficient  $K = 2.81$ . The time, heading angles, angle of deflection, and  $\gamma_{max}$  of the obtained path are compared in the given constraint range  $r_1 = 5$ . In this map, color ripples represent the interference source, blue clouds represent cloud obstacles, red is the starting point, and green is the target point. The final paths obtained by the four algorithms are represented by black (Figure 8a), purple (Figure 8b), rose (Figure 8c), and pink (Figure 8d), respectively.



**Figure 8.** Comparison of offline planning paths. (a) the RRT algorithm; (b) the RRT\* algorithm; (c) the IRRT\* algorithm; (d) the APF-IRRRT\* fusion algorithm.

From the above figures, it can be observed intuitively in Figure 8b that the RRT\* algorithm fared better than the traditional RRT algorithm, but it is still far from the UAV flight requirement. As is glaringly revealed in Figure 8c, the IRRT\* algorithm has the most petite inflection point; meanwhile, the instantaneous turning angle is too large and still exists. However, it seems to appear more concise because of constraints on search range. The fused APF-IRRRT\* algorithm in Figure 8d is optimal in terms of path smoothness, with

no inflection point, continuous angle, and smooth path. The charts of heading angle change and deflection angle change of the four algorithms are displayed in Figures 9 and 10.

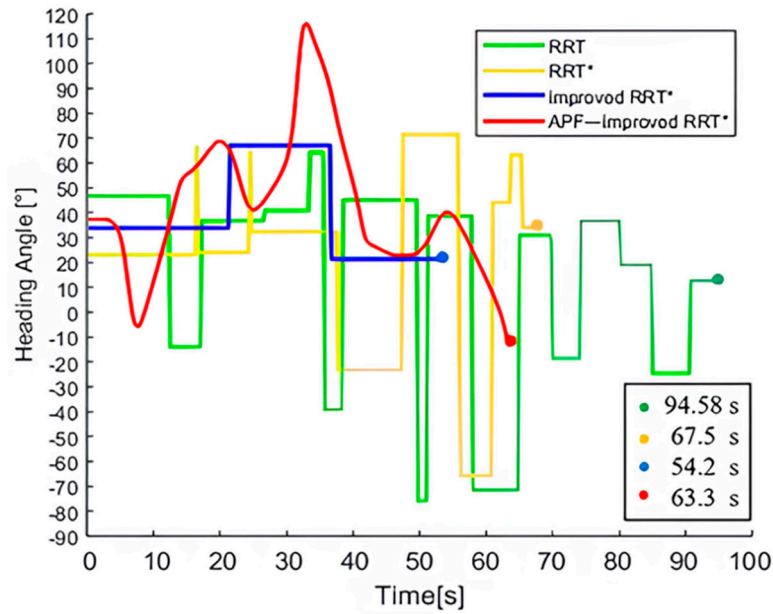


Figure 9. Heading angle change.

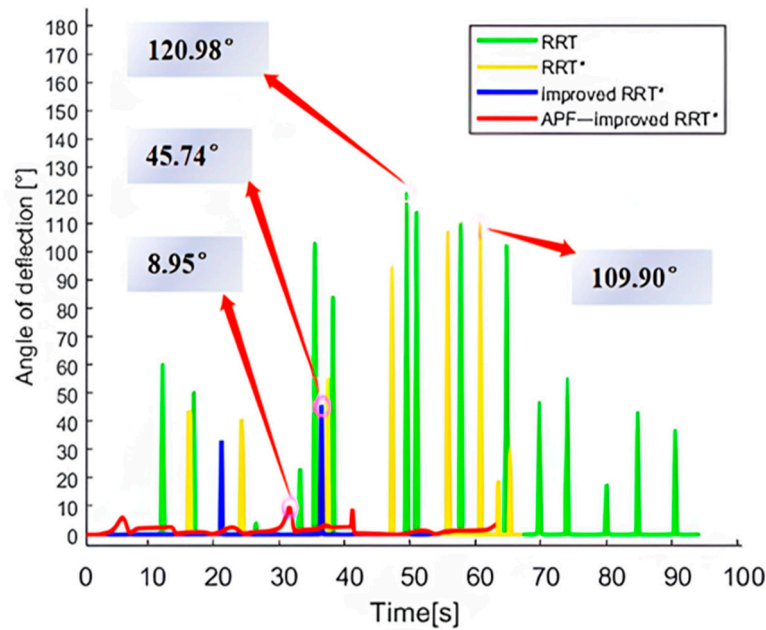


Figure 10. Deflection angle change.

The data obtained are summarized in Table 1.

Table 1. Comparison of offline planning simulation data.

Algorithms	Time (s)	$\gamma_{max}$ (°)
RRT	94.58	120.98
RRT*	67.5	109.90
IRRT*	54.2	45.74
APF-IRRT*	63.3	8.95

It is noticeable from Figure 9 that the heading angle of the APF-IRRT\* algorithm is always in a continuously changing attitude, and there is no dramatically instantaneous change of the angle, which effectively improves the safety of the UAV during actual flight. Table 1 clearly shows that in terms of time consumption, the RRT algorithm takes 94.58 s, which is the longest among the four algorithms. The shortest algorithm is the IRRT\* algorithm, which takes 54.2 s; meanwhile, the convergence speed is improved by 40.38% and 19.7%, compared to the traditional RRT algorithm. The APF-IRRT\* algorithm takes 63.3 s, ranking second. In Figure 10, the maximum rotation angle of the RRT algorithm is 120.98°, followed by the RRT\* algorithm at 109.90°. The improved RRT algorithm has a relatively obvious exactness, with a maximum rotation angle of 45.74°. For the fusion algorithm, although it takes 9.1 s more than the IRRT\* algorithm, its  $\gamma_{max}$  was only 8.95°, which was 80.43% higher than perfusion. Remarkably, the APF-IRRT\* algorithm obtains a safer and more effective flight path with a satisfying convergence rate in static offline planning. Hence, it has better practicability.

6.2. Online Path Planning Experiment

Because of the complexity of the actual disaster environment, the offline planning of UAVs often cannot safely reach the terminal for rescue, so the dynamic online path planning of the algorithm is essential. In the above-simulated map environment, when all parameters remain unchanged, different speeds and directions (random speeds and fixed directions) are assigned to each obstacle, which can more closely simulate the rapidly changing battlefield environment and more effectively test the reliability of the algorithm. Figure 11 depicts the moving direction of each obstacle.

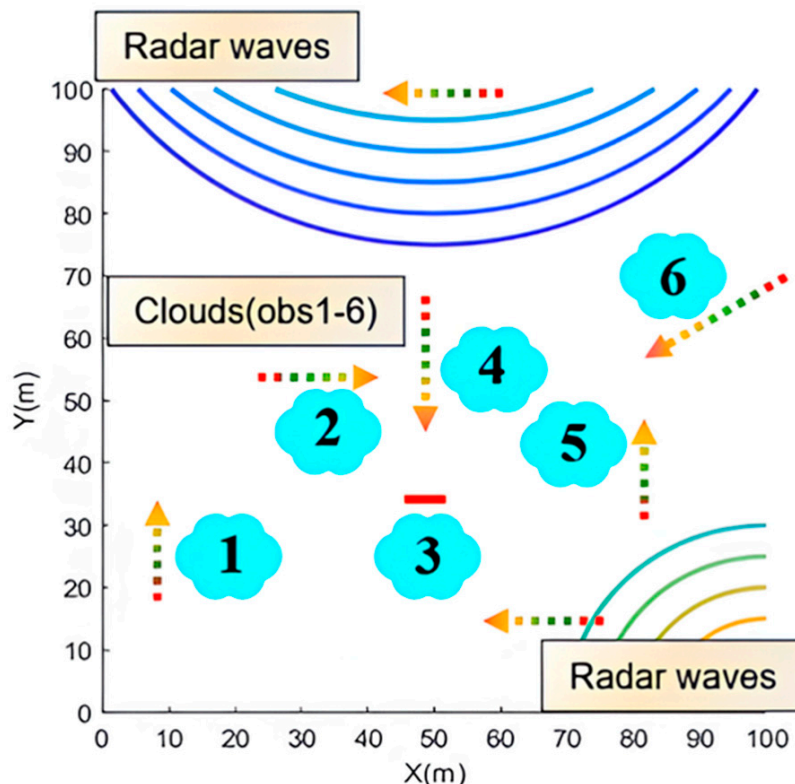
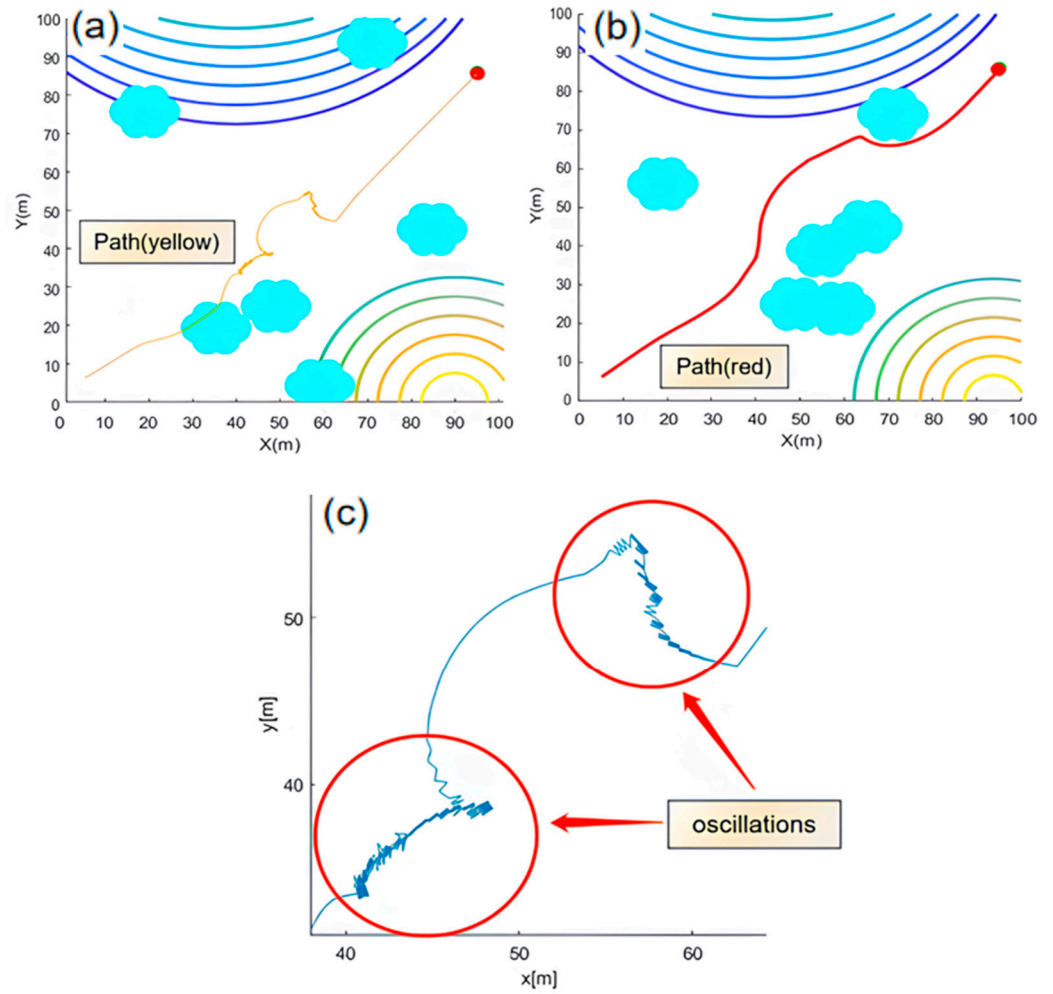


Figure 11. Schematic diagram of the dynamic environment.

The paths generated by the APF algorithm and the APF-IRRT\* algorithm are shown in yellow and red, respectively, in Figure 12a,b. The simulation diagram of the online planning experiment results is as follows.



**Figure 12.** Comparison of online planning paths. (a) the APF algorithm (b) the APF-IRRT\* fusion algorithm (c) Oscillation graph of the APF algorithm.

Judging from the above two figures, it is interesting to recognize that in a dynamic environment, when the APF algorithm encounters a moving obstacle, it falls into a local minimum and generates oscillation, as shown in Figure 12c, which significantly affects the safety and reliability of UAV flight. In contrast, the path obtained by the APF-IRRT\* algorithm is relatively smooth and stable. The charts of heading angle change and deflection angle change of the two algorithms are presented in Figures 13 and 14.

The data obtained are summarized in Table 2.

**Table 2.** Comparison of online planning simulation data.

Algorithms	Time (s)	$\gamma_{max} (^{\circ})$
APF	84.17	159.99
APF-IRRT*	72.67	23.94

It is apparent from Table 2 that the heading angle of the APF algorithm has a long time and large amplitude oscillation, and the angle close to  $160^{\circ}$  makes it impossible for the UAV to operate in the virtual environment. In addition, although the APF-IRRT\* algorithm is not stable in the static environment under the dynamic environment, and the heading angle changes in several cases, the maximum turning angle is only  $23.94^{\circ}$ , which is acceptable. The required time is 72.67 s, which is only 13.67% shorter than the APF algorithm, but the smoothness is improved by 85.03%. The fused APF-IRRT\* algorithm also performs well in a dynamic environment.

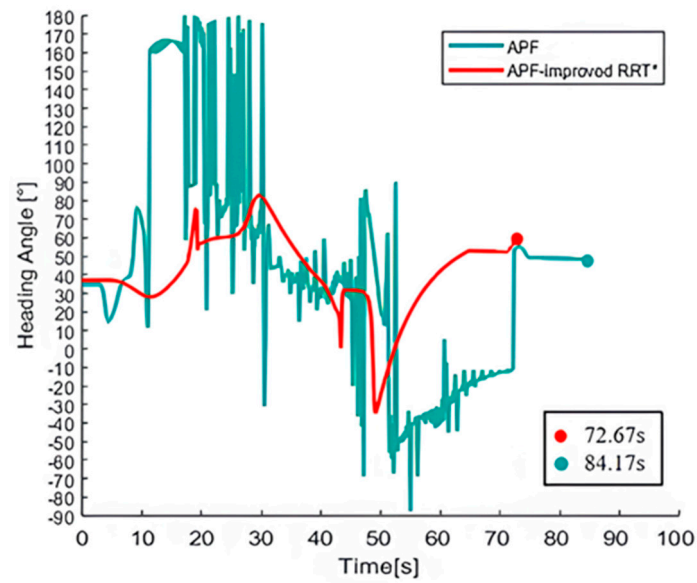


Figure 13. Heading angle change.

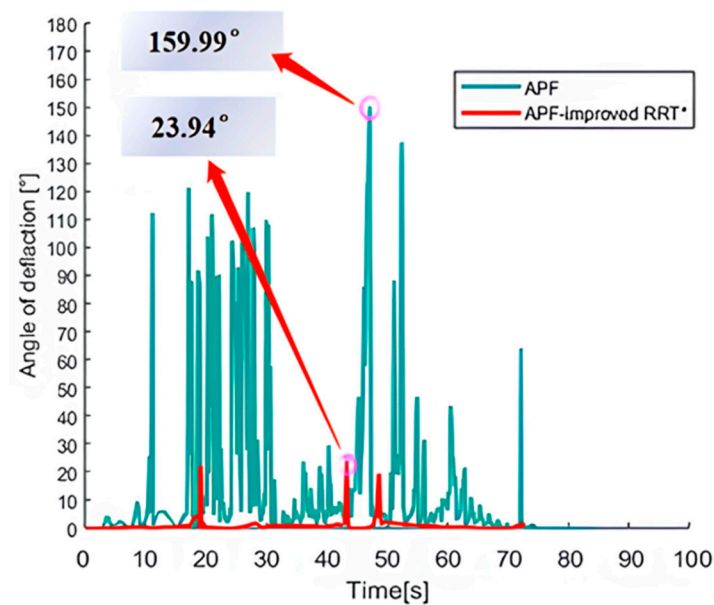


Figure 14. Deflection of angle change.

## 7. Conclusions

This paper proposed a new APF-IRRT\* fusion algorithm for rescue and disaster relief UAVs. This algorithm firstly improves the RRT\* algorithm; it not only reduces the calculation time, but also greatly improves the safety. Secondly, the introduction of the APF algorithm made it not only suitable for ordinary static obstacle environments, but also possesses better obstacle avoidance capabilities in changing dynamic environments to ensure its safety, which has effectively solved the problem of dangerous and complex surroundings of rescue drones. The path planning problem in the environment provides an effective option for the global path planning of the UAV in a specific environment. The algorithm mainly presents the characteristics of simple modeling, with few parameters to be adjusted, not being affected by the shape of obstacles, simple calculation, and easy implementation. Based on the algorithm in this paper, the global optimization path of the robot can be quickly and effectively planned. To ensure computational efficiency and prediction correctness, the target directivity of the extended node was given, which

effectively overcomes the inherent blindness of the RRT algorithm. The step-size adaptive and angle-increasing sampling methods were also utilized to limit the sampling range, thus reducing the tortuosity of the generated path and the time consumed.

To further enhance the path's smoothness, the generated path was optimized by calculating the angle value between the line connecting the new node's parent node and the original path. The simulation results demonstrate that the fusion APF-IRRT\* algorithm dramatically improves the issues of slow convergence speed and tortuous path of the traditional RRT algorithms in the static environment. Meanwhile, the APF-IRRT\* takes 63.3 s, ranking second. Despite it taking 9.1 s more than the IRRT\* algorithm, the maximum instantaneous steering angle recorded was only  $8.95^\circ$ , an increase of 80.43%, which greatly improved its flight stability and safety. In the dynamic environment, in spite of the fusion algorithm, the UAV was less stable in the static environment, and the change of heading angle also had several cases of turns that were too fast. The maximum turning angle registered was only  $23.94^\circ$ , which was within the acceptable range. The required time was 72.67 s, which was 13.67% shorter than the APF algorithm and 85.03% higher in smoothness.

Despite an enhanced performance and high practicability in static and dynamic environments of the fused APF-IRRT\* algorithm, UAVs need more sensitive reaction speeds and smooth paths to maintain high mobility in the disaster environment. Therefore, the next research step is to optimize the mathematical model and algorithm to overcome the bottleneck so that the convergence speed will not change when it is reduced to a certain extent. More importantly, obstacles can be in various forms, while this paper only uses the shape of clouds—which needs to be improved in future research.

**Author Contributions:** Conceptualization, J.Z.; methodology, J.Z. and Q.D.; investigation, M.L.; resources, M.L. and J.Y.; writing—original draft preparation, Q.D.; writing—review and editing, Q.D. and J.Y.; supervision, J.Z.; project administration, J.Z.; funding acquisition, J.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by Key research and development project of Jiangsu Province is Subject 1: Key technology research and development of micro-ecological hydropower energy storage equipment: BE2019009-1.

**Data Availability Statement:** Not applicable.

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

## References

1. Fraga-Lamas, P.; Ramos, L.; Mondejar-Guerra, V.; Fernandez-Carames, T.M. A review on iot deep learning uav systems for autonomous obstacle detection and collision avoidance. *Remote Sens.* **2019**, *11*, 2144. [[CrossRef](#)]
2. Singla, A.; Padakandla, S.; Bhatnagar, S. Memory-based deep reinforcement learning for obstacle avoidance in uav with limited environment knowledge. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 107–118. [[CrossRef](#)]
3. Wei, Z.; Meng, Z.; Lai, M.; Wu, H.; Han, J.; Feng, Z. Anti-collision technologies for unmanned aerial vehicles: Recent advances and future trends. *IEEE Internet Things J.* **2021**, *9*, 7619–7638. [[CrossRef](#)]
4. Xu, G.; Zhang, S.; Liu, H. Adaptive Optimal Control of UAV Formation Based on Policy Iteration. In Proceedings of the 2022 34th Chinese Control and Decision Conference (CCDC), Hefei, China, 15–17 August 2022; pp. 4145–4150.
5. Fransen, K.; van Eekelen, J. Efficient path planning for automated guided vehicles using a\* (astar) algorithm incorporating turning costs in search heuristic. *Int. J. Prod. Res.* **2023**, *61*, 707–725. [[CrossRef](#)]
6. Husain, Z.; Al Zaabi, A.; Hildmann, H.; Saffre, F.; Ruta, D.; Isakovic, A.F. Search and rescue in a maze-like environment with ant and dijkstra algorithms. *Drones* **2022**, *6*, 273. [[CrossRef](#)]
7. Pan, Y.; Yang, Y.; Li, W. A deep learning trained by genetic algorithm to improve the efficiency of path planning for data collection with multi-UAV. *IEEE Access* **2021**, *9*, 7994–8005. [[CrossRef](#)]
8. Wu, Z.; Meng, Z.; Zhao, W.; Wu, Z. Fast-RRT: A RRT-based optimal path finding method. *Appl. Sci.* **2021**, *11*, 11777. [[CrossRef](#)]
9. Wang, J.; Li, B.; Meng, M.Q.H. Kinematic Constrained Bi-directional RRT with Efficient Branch Pruning for robot path planning. *Expert Syst. Appl.* **2021**, *170*, 114541. [[CrossRef](#)]
10. Zekui, Q.; Rui, W.; Xiwang, D.; Qingdong, L.; Dongyang, F.; Zhang, R. Three-dimensional path planning for unmanned aerial vehicles based on the developed RRT algorithm. In Proceedings of the 2018 IEEE CSAA Guidance, Navigation and Control Conference (CGNCC), Xiamen, China, 10–12 August 2018; pp. 1–5.



11. Keyu, L.; Yonggen, L.; Yanchi, Z. Dynamic obstacle avoidance path planning of UAV Based on improved APF. In Proceedings of the 2020 5th International Conference on Communication, Image and Signal Processing (CCISP), Chengdu, China, 13–15 November 2020; pp. 159–163.
12. Bai, X.; Jiang, H.; Cui, J.; Lu, K.; Chen, P.; Zhang, M. UAV Path Planning Based on Improved A and DWA Algorithms. *Int. J. Aerosp. Eng.* **2021**, *2021*, 4511252. [[CrossRef](#)]
13. Zhao, C.; Zhu, Y.; Du, Y.; Liao, F.; Chan, C.-Y. A novel direct trajectory planning approach based on generative adversarial networks and Rapidly-exploring random tree. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 17910–17921. [[CrossRef](#)]
14. Li, B.; Chen, B. An adaptive Rapidly-exploring random tree. *IEEE-CAA J. Autom. Sin.* **2022**, *9*, 283–294. [[CrossRef](#)]
15. Pan, Z.; Zhang, C.; Xia, Y.; Xiong, H.; Shao, X. An improved artificial potential field method for path planning and formation control of the multi-uav systems. *IEEE Trans. Circuits Syst. II Express Briefs* **2022**, *69*, 1129–1133. [[CrossRef](#)]
16. Zhang, X.; Zhu, T.; Du, L.; Hu, Y.; Liu, H. Local path planning of autonomous vehicle based on an improved heuristic bi-rrt algorithm in dynamic obstacle avoidance environment. *Sensors* **2022**, *22*, 7968. [[CrossRef](#)] [[PubMed](#)]
17. Fan, J.; Chen, X.; Liang, X. UAV trajectory planning based on bi-directional apf-rrt\* algorithm with goal-biased. *Expert Syst. Appl.* **2023**, *213*, 119137. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.