sensors

MDPI

*Article*

# Complex Environment Path Planning for Unmanned Aerial Vehicles

**Jing Zhang [1], Jiwu Li [1], Hongwei Yang [1,*], Xin Feng [1,2] and Geng Sun [3]**

[1] College of Computer Science and Technology, Chang Chun University of Science and Technology, Changchun 130022, China; zhang_jing@cust.edu.cn (J.Z.); 2019100656@mails.cust.edu.cn (J.L.); fengxin@cust.edu.cn (X.F.)
[2] Chongqing Research Institute of Changchun, University of Science and Technology, Chongqing 401122, China
[3] College of Computer Science and Technology, Jilin University, Changchun 130012, China; sungeng@jlu.edu.cn
* Correspondence: yanghongwei@cust.edu.cn

**Abstract:** Flying safely in complex urban environments is a challenge for unmanned aerial vehicles because path planning in urban environments with many narrow passages and few dynamic flight obstacles is difficult. The path planning problem is decomposed into global path planning and local path adjustment in this paper. First, a branch-selected rapidly-exploring random tree (BS-RRT) algorithm is proposed to solve the global path planning problem in environments with narrow passages. A cyclic pruning algorithm is proposed to shorten the length of the planned path. Second, the GM(1,1) model is improved with optimized background value named RMGM(1,1) to predict the flight path of dynamic obstacles. Herein, the local path adjustment is made by analyzing the prediction results. BS-RRT demonstrated a faster convergence speed and higher stability in narrow passage environments when compared with RRT, RRT-Connect, P-RRT, 1-0 Bg-RRT, and RRT*. In addition, the path planned by BS-RRT through the use of the cyclic pruning algorithm was the shortest. The prediction error of RMGM(1,1) was compared with those of ECGM(1,1), PCGM(1,1), GM(1,1), MGM(1,1), and GDF. The trajectory predicted by RMGM(1,1) was closer to the actual trajectory. Finally, we use the two methods to realize path planning in urban environments.

**Keywords:** unmanned aerial vehicles; narrow passages; path planning; pruning; trajectory prediction



**Citation:** Zhang, J.; Li, J.; Yang, H.; Feng, X.; Sun, G. Complex Environment Path Planning for Unmanned Aerial Vehicles. *Sensors* **2021**, *21*, 5250. https://doi.org/10.3390/s21155250

Academic Editor: Weizhi Meng

Received: 23 June 2021
Accepted: 27 July 2021
Published: 3 August 2021



## 1. Introduction

Unmanned aerial vehicles (UAVs) have gradually spread from military to civilian use and can be used in different areas, such as climate detection [1], environmental research [2], intelligent transportation [3], and rescue and search operations [4], by embedding various devices. For UAVs, performing missions in complex urban environments is more difficult than open space missions. Two main influencing factors are noted. First, path planning is difficult because the space is divided by buildings.

Second, the uncertainty of the trajectory of dynamic flight obstacles affects the safety of UAVs. Herein, path planning in a complex environment can be divided into global path generation and local path adjustment. The goal of global path planning is to plan the most efficient path in the shortest time in the overall static space, and local path adjustment is used to avoid threatening dynamic flight obstacles. Path planning for UAVs in complex environment is promising for future development.

Many narrow passages are found in urban environment, and multiple divided spaces may be connected by several narrow passages. Performing global path planning in narrow passages that have a smaller volume than the overall space is difficult. The calculation of path planning algorithm based on graph search is complicated [5–7]. The artificial potential field method [8] easily falls into local shocks and fails at path planning. Swarm intelligence [9,10] and neural networks [11,12] require long iterations or training. The

sampling-based rapidly-exploring random tree (RRT) algorithm has shown outstanding performance.

RRT [13] was proposed for path planning with non-completion constraints of vehicle dynamics/kinematics. RRTs, which are efficient in solving complex spatial path solving problems, have no preprocessing. In addition, the RRT algorithm is simple to implement. For algorithms that are based on graph search, RRT calculations, which are also applicable to swarm intelligence and neural network algorithms, are simpler. However, the randomness of RRT enables the algorithm to generate more useless branch nodes, and they may spend considerable time in identifying and passing through unknown narrow passages.

Local path adjustment depends on the known trajectory information of dynamic flight obstacles. Therefore, a high-precision prediction model is a guarantee for UAVs to avoid dynamic obstacles. Regression analysis prediction model aims to make predictions by analyzing the functional relationship between observation data and statistical methods [14,15]. The autoregressive moving average model uses a mathematical model to approximately describe the data series generated by the object with the change in time [16,17]. The Markov prediction model uses the method of probability theory to study the change law of random events to predict the future state [18,19].

The neural network uses the steepest descent method to adjust the weights and thresholds through the back propagation of errors to predict the future trend [20,21]. Considering that more data will be processed, the time consumption when building the model is usually high. However, the grey prediction model [22] takes small data as the research object, and it has a relatively simple processing method for the original data. This model has high prediction accuracy. Related prediction models have been used in various aspects [23–25]. GM(1,1) is used as the representative model of the grey prediction model, which is especially suitable for this kind of forecasting. The contributions of this paper are presented as follows:

- In this paper, a BS-RRT algorithm is proposed to solve the global path planning problem. The algorithm converges quickly in the urban environment with narrow passages, and the algorithm has remarkable stability.
- To optimize the BS-RRT path further, a cyclic pruning algorithm is proposed to optimize the global programming path, and the effectiveness of the algorithm increases with the path's tortuous degree.
- Local path adjustment depends on the prediction model. To improve the accuracy of the prediction model, this paper improves the calculation of the background value of the GM(1,1) model. At the same time, combined with the idea of metabolism, (the new method is presented to optimize the GM(1,1) model in this paper), which improves the prediction accuracy of the model greatly.
- This paper provides a feasible path planning scheme for UAV flight in a complex environment.

## 2. Related Work

Thus far, RRTs have been widely used to solve path planning problems in many fields. However, RRTs grow themselves incrementally by random sampling point. Meanwhile, in the case of narrow passages in urban space, the small volume of narrow passages leads to a lower probability of random sampling points being generated in the narrow passageway compared with the volume of the overall space, which costs more time to sample wide open areas before passing the narrow passage. One solution is to vary the randomness of the sampling points, which means that sampling points occur with different probabilities in various areas. According to the exploration results of the RRT algorithm in Yershova, A. [26], the sampling area of the sampling point is limited.

Jaillet, L. et al. [27] used adaptive sampling to adjust the sampling interval dynamically according to the current situation. In [28], the author adjusted the probability of random points appearing at the end point to guide the growth of RRT. The algorithm has high efficiency in the environment with a narrow passage, but the efficiency of the algorithm

drops with the increase of the complexity of the environment. In [29], the algorithm grows in a favorable direction without collision by controlling the probabilistic diffusion and analyzing the results of the diffusion algorithm.

In [30], the author used the utility-oriented model to guide the random tree to allocate more computing resources in complex areas. Another approach is to generate multiple exploration trees simultaneously in space to deal with the narrow channel problem. In [31], the author proposed that two growing trees grow from the starting point and the end point. In [32], the author proposed an incremental path planning method that has obvious effect on solving local paths. In [33], the author proposed a parallel algorithm (C-FOREST) that is used to query the shortest path and grow multiple search trees simultaneously in parallel.

In [34], the authors extended the bidirectional tree to a multi-tree framework where any possible samples can be stored, and new trees are constructed. The aforementioned works have made some progress in RRT path planning. However, RRT expansion and connection is still a difficult problem for complex environments, and the time difference of each completion of path planning is large. Thus, the stability of the algorithm is low. The RRT algorithm reduces local re-sampling to a certain extent by changing the sampling strategy, which is prone to local oscillations before it passes through narrow passages, thereby, wasting certain computing resources.

In many cases, the whole space will be filled faster by the expansion of multiple trees, whereas the path search of narrow passage takes more sampling time. In this paper, BS-RRT is proposed to solve the problems of long time required to plan the global path of narrow passage; meanwhile, it has small time fluctuations in each planning. The initial RRT expansion is guided by a greedy mind to continuously expand toward the end point, which avoids wasting excessive computing resources in spacious space. Branch selection is conducted to pass the narrow passages when BS-RRT runs into obstacles. The optimal branch leaf node is selected as the root node to greed grow until connected to the end. Loop pruning is used to optimize the global path when connected from start to end.

When the global path is planned, the UAV needs to avoid local dynamic flight obstacles during the flight, thereby, changing the local path. The local path adjustment of UAV depends on the prediction results of the future trajectory of dynamic flight obstacles. the grey prediction model is suitable for predicting the trajectory of dynamic flight obstacles. The accuracy of background value estimation affects the accuracy of the model [35,36]. In [37], the author improved the accuracy and adaptability of the model by combining particle swarm optimization with the TWGM(1,1) algorithm to optimize the order and background value coefficient.

In [38], the GM (1,1) model was optimized using the Lagrange mean value theorem and interpolation coefficient method, and a background value that was correlated with the new variable K was constructed. Meanwhile, the accuracy of the model was improved by adding parameters. In [39], the author used an exponential curve, power function curve, polynomial curve, and interpolation function to optimize the background value, and the simulation results showed that these methods improved the accuracy of the model very well. In [40], the author improved the accuracy of the model by using piecewise cubic interpolation splines to reconstruct the background values while maintaining the monotonicity of the accumulated data.

In [41], the author used particle swarm optimization to optimize the development coefficient of the grey model, optimized the initial value of GM(1,1), and introduced the sliding window to improve the accuracy of the model. However, most models have low accuracy of long-term prediction performance because they do not consider the principle of new information priority. Swarm intelligence algorithm takes a long time to optimize the model. Runge phenomenon [42] may occur in the optimization method of high-order interpolation, which may affect the accuracy of the model.

In this paper, the GM(1,1) model is combined with the idea of metabolism, the principle of new information is fully considered priority, and the length selection of the model is discussed. At the same time, the 1-AGO dynamic sequence prediction model is combined

with the Romberg quadrature formula to recalculate the background value to improve the prediction accuracy of the model. The results of the predictive model are used by the UAV as a reference to adjust the local path. In this paper, the global path planning algorithm is combined with the prediction model to realize the autonomous flight of UAV in complex environment.

The remainder of this paper is organized as follows. In Section 3, we explain the principle of the BS-RRT algorithm and cyclic pruning algorithm, the optimization method of RMGM(1,1), and the scheme of local path adjustment. In Section 4, we compare the performance of BS-RRT and the cyclic pruning algorithm with RRT, RRT-Connect, P-RRT, 1-0 Bg-RRT [28], and RRT* through simulation experiments in a narrow passage, and we compare the prediction accuracy of RMGM(1,1) with ECGM(1,1), PCGM(1,1), GM(1,1), MGM(1,1), and GDF. Moreover, we verify the feasibility of the local path adjustment scheme. In Section 5, we present the summary and conclusions.

## 3. Principles of Path Planning in Complex Environment

The goal of path planning is to plan the most effective path in a short time. First, we briefly introduced the original RRT path planning algorithm. The principle of the improved algorithm BS-RRT and cyclic pruning algorithm for global path planning are described in detail in Section 3.1. Second, when the global path planning is finished, the UAV needs to predict the future trajectory of the captured dynamic flight obstacles in the process of flying along the trajectory.

We use the optimized RMGM(1,1) model to complete the prediction of the future flight trajectory of dynamic flight obstacles. The modeling process and error analysis of GM(1,1) and the modeling principle of RMGM(1,1) are introduced in Section 3.2. Finally, in Section 3.3, we describe a local path adjustment scheme that makes the path adjustment according to the predicted results of Section 3.2. We use a combination of global path planning and a local path adjustment scheme to achieve autonomous flight of the UAV.

### *3.1. Global Path Planning*

#### 3.1.1. Principle of RRT

The RRT constructs a tree-like data structure to incrementally explore unknown space by randomly sampling points. Figure 1 shows the schematic of the RRT extension. First, a random node is generated with random coordinate in space. Second, the tree is traversed by RRT to find the node closest to the sampling point. Third, a new node of one step along the direction that this node points to the sampling point. Finally, whether collision occurs from the nearest node to the new node is verified. Collision validation determines whether new nodes can be inserted into the tree. The above process is repeated until the algorithm finds a path that connects the starting point and the point goal.

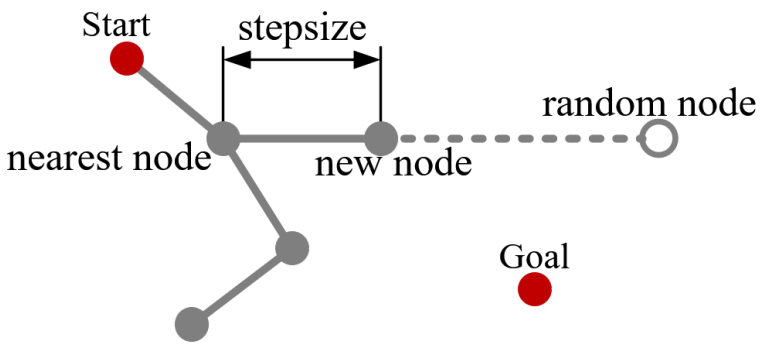



**Figure 1.** Principle of RRT extension.

#### 3.1.2. Principle of BS-RRT

The expansion process of BS-RRT is divided into three parts. First, the expanded tree grows greedily along the direction of the end point. Second, when the expanding

tree encounters an obstacle, it branches and grows to pass the obstacle. Finally, after the expanded tree passes the obstacle, the leaf node of the branch with the closest Euclidean distance to the end point among the branches is selected to grow again greedily. These three steps are circulated, in turn, to finally generate a path connecting the starting point and the goal point.

- Greedy growth
  In the initial phase, the sampling strategy is changed to speed up the tree expansion, and the sampling point for each sampling coincides with the goal point. In Figure 2, the BS-RRT continues to expand toward the goal point. The algorithm goes to the next stage when the BS-RRT fails to grow.

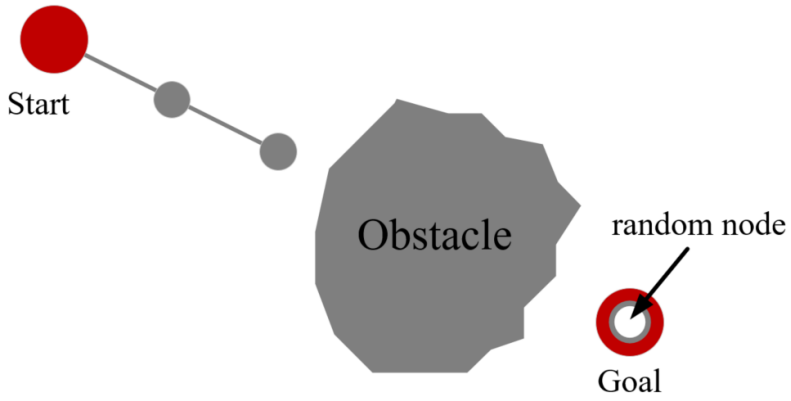



**Figure 2.** Greedy growth.

- Branch growth
  When the greedy growth stops, the BS-RRT tree will enter the branch growth stage to avoid obstacles. Point *A* in Figure 3a represents that point *A* failed to extend the next greedy point, and thus the point enters the branch, and point *A* is set as the root point of the branch. The line from the root point to the goal point divides the space into left and right regions. In the left region, the sampling point gradually moves away from the goal point in a clockwise trend each time, and the sampling stops until point *C* is expanded successfully. The sampling point in the right region gradually moves away from the goal point counterclockwise each sampling, the sampling stops until point *B* is extended successfully. In Figure 3b, points *B* and *C* cannot grow greedily because obstacles are observed in the direction from point *B* and *C* to the goal point. Hence, they continue to branch.
  The sampling point from point *C* to the direction of the goal point moves clockwise away from the goal point for each sampling until point *E* is expanded successfully. Similarly, the sampling point from point *B* points to the direction of the goal point moves counterclockwise away from the goal point each sampling until point *D* is expanded successfully. At this point, points *E* and *D* can grow greedily, and branch growth ends. Notably, in the branching stage, the length of the extension of all nodes is equal to the initial step size.
  Other branch nodes only expand one child node in addition to the original root point *A*. In the regions divided by root point *A*, the sampling point will only shift the sampling clockwise in the left region or counterclockwise sampling on the right, such as points C and E expand clockwise, meanwhile, points B and D expand counterclockwise. The branch growth in the corresponding region deviated to the direction of the goal point pointing to the leaf point to stop until the sampling point.

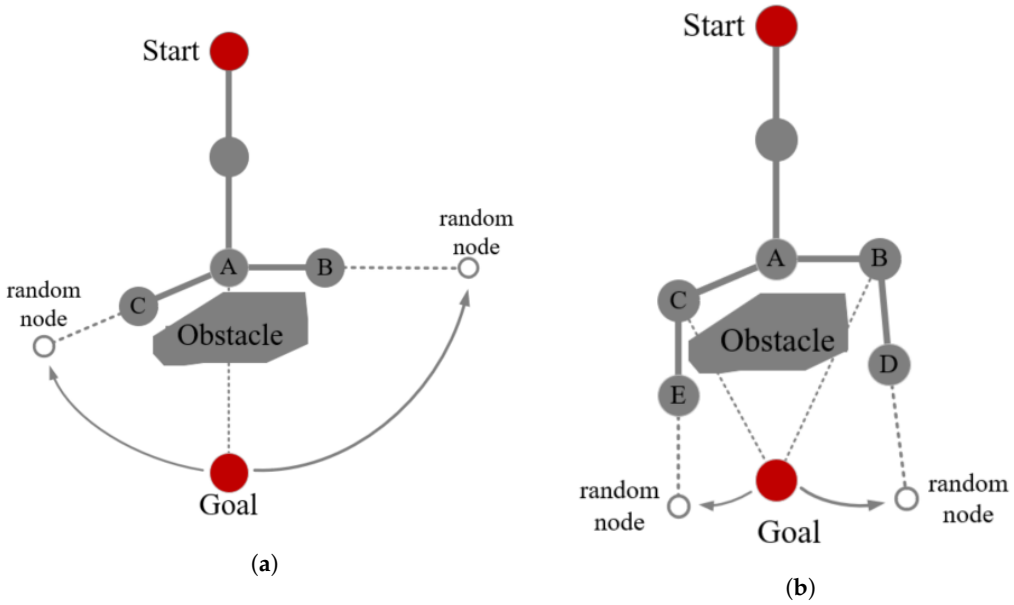



**Figure 3.** (**a**) shows the first branch growth. The point A is the root node, the point C and the point B are the leaf nodes; (**b**) shows the second branch growth. The point C and the point B are the root nodes, the point E and the point D are the leaf nodes.

- Branch selection
  When the branches stop growing, this indicates that BS-RRT has passed the barrier and enters the branch selection stage. If one branch fails to grow, then another branch will be selected directly. The overall BS-RRT algorithm will fail when both branches fail. The algorithm selects the branch with a short Euclidean distance from the goal point when the branches grow successfully. The selected leaf point of the branch as the starting point goes to the first step of greedy growth.

#### 3.1.3. Cyclic Pruning Algorithm

The aim of cyclic pruning algorithm is to optimize the path generated by BS-RRT, and the shortest path from the start point to the end point will be obtained accordingly. The generated path will be pruned twice by a cyclic pruning algorithm. Each point in the path from the start point to the end point is denoted from 1 to $n$. The first pruning will significantly reduce the number of unnecessary nodes as well as the path length, and the second pruning will shorten the path length while keeping the number of nodes unchanged.

- The first pruning
  We define the following set: set $V$={$v_1$,$v_2$,$v_3$,..., $v_{n-1}$}, set $P$={$p_1$, $p_2$, $p_3$,..., $p_{n-1}$}, and empty set $U$. Set $V$ traverses every point in the path from the start point to the $n-1$th point, set $P$ traverses each point in the path from the end point $n$th to the third point in reverse, and the point in the set records the coordinate information of every point in the path. Based on the above definition, the point connecting to itself will be avoided. The first pruning of the path is performed as follows. First, we determine whether the point $v_i$ in the set $V$ is connected to any point in set $P$. If the point $v_i$ and the point $p_j$ can be connected with no obstacle, then the points from $v_i$ to $v_{n-j+1}$ are deleted from the set $V$, the points from $p_{n-i+1}$ to $p_j$ are deleted from the set $P$ as well. Meanwhile, the points $v_i$ and $p_j$ are added to set $U$.
  Repeat the above process until the set $V$ is empty, point $p_1$ will be added to the set $U$ if the point $p_1$ is not in the set $U$. The set $U$ is the final result set after the first pruning. An example is shown in Figure 4 to describe the process of the first pruning in detail. There are seven points in the path generated by BS-RRT, where the hollow circles represent points that are not in sets $V$ and $P$, dashed lines represent two points fail

to connect due to the existence of obstacles, and solid lines represent two points can be connected directly. Thus, set $V = \{v_1,v_2,v_3,v_4,v_5\}$ and set $P = \{p_1,p_2,p_3,p_4,p_5\}$ can be constructed. The first pruning operation is divided into three-steps in Figure 4.

First, in determining whether the point $p_j$ in the set $P$ can be connected to point $v_1$, the initial value of $j$, $j$ is set as 1. Figure 4I shows that point $v_1$ and $p_5$ can be connected directly. Then, points $v_1$ and $v_2$ are deleted from set $V$, point $p_5$ is deleted from set $P$, and point $v_1$ and $p_5$ are added to set $U$.

Second, we determine whether the point $p_j$ in the set $P$ can be connected to point $v_3$ in turn, and point $v_3$ and $p_2$ can be connected directly as shown in Figure 4II. Points $v_3$, $v_4$, and $v_5$ are deleted from set $V$, points $p_2$, $p_3$, and $p_4$ are deleted from set $P$, and points $v_3$ and $p_2$ are added to set $U$.

Finally, the set $V$ is empty in Figure 4III, the point $p_1$ will be added to the set $U$ if $p_1$ is not in the set $U$. Thus, the set $U = \{v_1,p_5,v_3,p_2,p_1\}$. All of points in the set $V$ can be connected in turn, and the first pruning is finished.

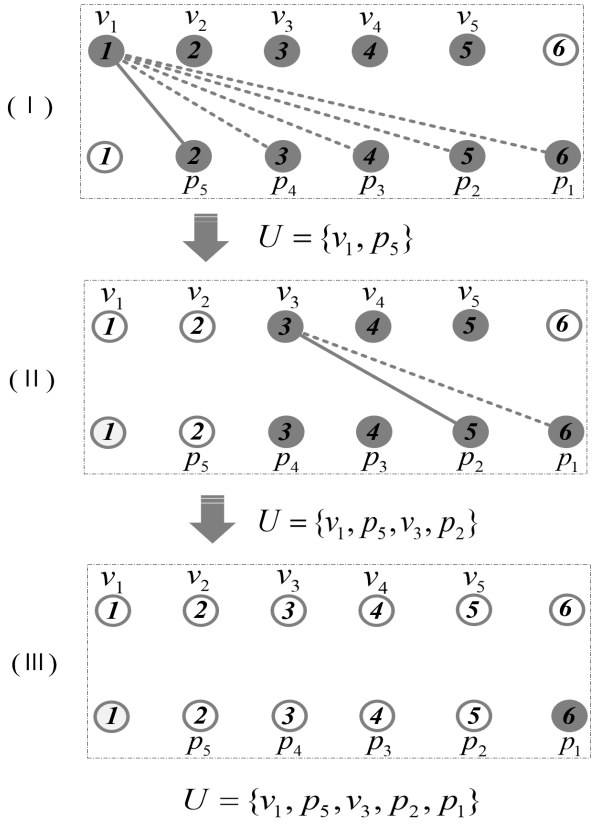



**Figure 4.** First pruning of the path, (**I**) shows the point $v_1$ and the point $p_5$ are connected successfully; (**II**) shows the point $v_3$ and the point $p_2$ are connected successfully; (**III**) shows no point can connect successfully.

- The second pruning

  Assume that there are m points in the set $U$ after the first pruning, we define the following sets, set $C = \{c_1,c_2,c_3,\ldots, c_m\}$, set $Z = \{z_1,z_2,z_3,\ldots, z_{m-2}\}$. The points in set C represent the points in set $U$ starting from $u_m$ to $u_1$. The points in set $Z$ represent the points in set $U$ starting from point $u_m$ to point $u_3$. The missing points $u_1$ and $u_2$ in set $Z$ ensures that the connection of the last two points is not on the same line segment.

  The second pruning operation is performed as follows. Determining whether the first point $z_1$ in the set $Z$ can be connected to the midpoint of the two points in the set $C$, that is, the midpoint of $c_i$ and $c_{i+1}$ connecting $z_{i-1}$. If the connection is successful, then the midpoint of $c_i$ and $c_{i+1}$—namely node $c_{i+0.5}$—will replace the point $c_i$ in set $C$.

Point $z_{i+1}$ will be determined in turn. After all the points in the point set $Z$ are tested, the points in the set $C$ forms the final path, and the cyclic pruning algorithm ends.
An example is shown in Figure 5 to describe the process of the second pruning in detail. There are five points in the set $U$, the hollow circles represent points that are not in sets $Z$, dashed lines represent the failed connection between two points, and solid lines represent successful connections between two points. Set $C = \{c_1,c_2,c_3,c_4,c_5\}$ and set $Z = \{z_1,z_2,z_3\}$ are constructed accordingly.
Similarly, the second pruning operation is divided into three steps in Figure 5. First, node $z_1$ is connected to node $c_{2.5}$ successfully, and then point $c_{2.5}$ is added to the set $C$ to replace point $c_2$. Second, point $c_{3.5}$ will not add to the set $C$ because point $z_2$ failed to connect to point $c_{3.5}$. Finally, point $c_{4.5}$ will replace point $c_4$ in the set C because point $z_3$ can be connected to node $c_{4.5}$ with no obstacle. The cyclic pruning algorithm ends and the set $C$ is the final pruning result.

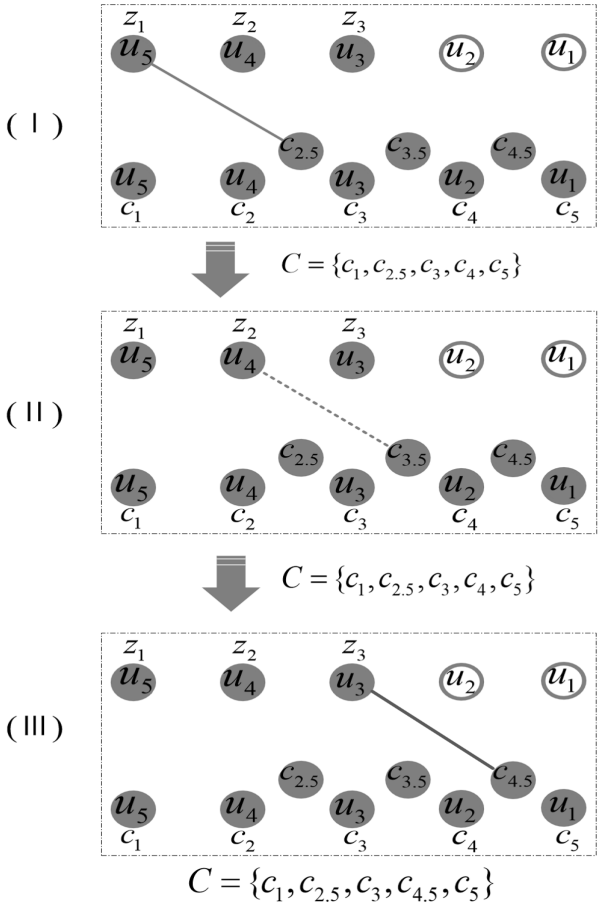



**Figure 5.** Second pruning of the path, (**I**) shows the point $z_1$ and the point $c_{2.5}$ are connected successfully; (**II**) shows the point $z_2$ and the point $c_{3.5}$ fail to connect; (**III**) shows the point $z_3$ and the point $c_{4.5}$ are connected successfully.

### *3.2. Trajectory Prediction of Dynamic Obstacles*

We assume that the coordinate sequence of a dynamic obstacle is $P(k) = (x(k), y(k), z(k))$, $k = 1, 2, ..., n$. $x(k), y(k)$ and $z(k)$ are coordinates for the $X, Y$, and $Z$ axes at time $k$. The map is constructed in the first quadrant to make the values of the coordinate sequence non-negative. Three sets of data sequences $(x(1), x(2), ..., x(n))$, $(y(1), y(2), ..., y(n))$, and $(z(1), z(2), ..., z(n))$ are obtained by splitting the original coordinate sequence $P(k)$. Three predicted values $\hat{x}(n+1)$, $\hat{y}(n+1)$, and $\hat{z}(n+1)$ are obtained by building a prediction model for the three sets data series respectively, and the predicted coordinates $\hat{p}(n+1) = (\hat{x}(n+1), \hat{y}(n+1), \hat{z}(n+1)$ are obtained.

In this part, the X axis coordinate sequence $(x(1), x(2), ..., x(n))$ is used as an example to demonstrate the main modeling steps of the GM(1,1) model, and then the error of the model is analyzed. We combine the first-order cumulative dynamic sequence prediction model and Romberg's numerical integration formula to recalculate the background value to reduce the error of the model. We also rebuild the model with the idea of metabolism.

3.2.1. Principle and Error Analysis of GM(1,1) Modeling

- Modeling of Grey Model GM(1,1)
  Let the original non-negative sequence data be

$$X^{(0)} = \left(x^{(0)}(1), x^{(0)}(2), \ldots, x^{(0)}(n)\right). \tag{1}$$

From Equation (1), $x^{(0)}(k) = x(k)$, and digital (0) indicates that the initial data sequence. The $X^{(1)}$ is given as follows

$$X^{(1)} = \{x^{(1)}(1), x^{(1)}(2), ..., x^{(1)}(n)\}. \tag{2}$$

where

$$x^{(1)}(k) = \sum_{i=0}^{k} x^{(0)}(k) = x^{(1)}(k-1) + x^{(0)}(k), k = 1, 2, \cdots, n. \tag{3}$$

Equation (3) is first-order accumulated generation operating (1-AGO) series of $X^{(0)}$. From Equation (3), we suppose that sequence $X^{(1)}$ meets the following first-order grad forecasting differential equation

$$\frac{\mathrm{d}x^{(1)}(t)}{\mathrm{d}t} + ax^{(1)}(t) = b. \tag{4}$$

The solution of Equation (4) with the initial condition $\hat{x}^{(1)}(1) = x^{(1)}(1)$ is presented as follows

$$\hat{x}^{(1)}(k+1) = \left(x^{(0)}(1) - \frac{b}{a}\right)\mathrm{e}^{-ak} + \frac{b}{a}, k = 1, 2, 3, \ldots, n. \tag{5}$$

Thus, we obtained the following grey prediction equation

$$\begin{aligned} \hat{x}^{(0)}(k+1) &= \hat{x}^{(1)}(k+1) - \hat{x}^{(1)}(k) \\ &= (1 - \mathrm{e}^{a})\left(x^{(0)}(1) - \frac{b}{a}\right)\mathrm{e}^{-ak}, k = 1, 2, 3, \ldots, n. \end{aligned} \tag{6}$$

Here, $k = n$, $\hat{x}^{(1)}(n+1)$ is the predicted coordinate on the X axis. To obtain the prediction model of the raw data sequence, we need to determine the grey development coefficient $a$ and the grey control parameter $b$ in Equation (4). For this purpose, we performed the integral accumulation on both sides of Equation (4) for every contiguous interval, and then we can obtain

$$x^{(1)}(k+1) + a\int_{k}^{k+1} x^{(1)}(t)dt = b. \tag{7}$$

Let the background value be

$$z^{(1)}(k+1) = \int_{k}^{k+1} x^{(1)}(t)dt. \tag{8}$$

Consequently, to estimate the values of $a$ and $b$, we must use some methods to estimate the background value $z^{(1)}(k+1)$. We yield the estimated background value $z^{(1)}(k+1)$ as follows

$$\mathrm{z}^{(1)}(k+1) = \frac{1}{2}[x^{(1)}(k) + x^{(1)}(k+1)]. \tag{9}$$

The method of solving $a, b$ parameters is expressed as

$$\hat{\alpha} = (a, b)^{\mathrm{T}} = \left(B^{\mathrm{T}}B\right)^{-1}B^{\mathrm{T}}Y, \tag{10}$$

$$B = \begin{bmatrix} -z^{(1)}(2) & 1 \\ -z^{(1)}(3) & 1 \\ \vdots & \vdots \\ -z^{(1)}(n) & 1 \end{bmatrix}, Y = \begin{bmatrix} x^{(0)}(2) \\ x^{(0)}(3) \\ \vdots \\ x^{(0)}(n) \end{bmatrix}. \tag{11}$$

- Error analysis
  Equation (8) shows that the classical GM(1,1) model uses the average of adjacent values to estimate the background value $z^{(1)}(k)$. Its geometric meaning is that the trapezoidal area, which is based on the edge of exponential curve $x^{(1)}(t)$, has been replaced by the area of straight ladder. This method has significant error when the 1-AGO data sequence varies greatly. As shown in Figure 6, ΔS is the error existing in the model.

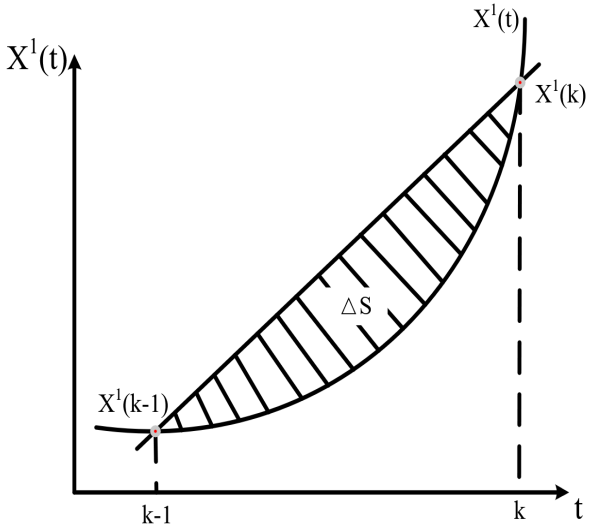



**Figure 6.** Reasons for GM(1,1) model error.

According to the error analysis results, the curve $x^{(1)}(t)$ has fitting by first-order accumulation (1-AGO) dynamic sequence prediction model [43], and the trapezoidal area, which is based on the edge of the exponential curve $x^{(1)}(t)$, was recalculated by the Romberg numerical integration formula. Meanwhile, the optimized model was constructed by combining the idea of metabolism, named RMGM(1,1).

3.2.2. Principle of RMGM(1,1) Modeling

In this part, the X axis coordinate sequence $(x(1), x(2), ..., x(n))$ is used as an example in the same way to describe the principles of the 1-AGO dynamic sequence prediction model briefly and Romberg's numerical integration formula, as well as the construction of the RMGM(1,1) model.

- 1-AGO dynamic sequence prediction model [43]
  Assume that Equations (1) and (2) meet the following law: If $x^{(0)}(k)$ has the form of homogeneous exponential growth $x^{(0)}(k)$=$ce^{a(k-1)}$, then the 1-AGO sequence is in the form of non-homogeneous exponential sequence (i.e., Equation (12)) and vice versa.

$$x^{(1)}(k) = Ae^{\alpha(k-1)} + B, k = 1, 2, \cdots, n. \tag{12}$$

The solving formula of each parameter is expressed as follows

$$\alpha = \ln \frac{x^{(0)}(k)}{x^{(0)}(k-1)}, \tag{13}$$

$$A = \frac{x^{(0)}(k)}{\left(\frac{x^{(0)}(k)}{x^{(0)}(k-1)}\right)^{k-1}\left(1 - \frac{x^{(0)}(k-1)}{x^{(0)}(k)}\right)}, \tag{14}$$

$$B = x^{(0)}(k_1) - A = x^{(0)}(k_1) - \frac{x^{(0)}(k)}{\left(\frac{x^{(0)}(k)}{x^{(0)}(k-1)}\right)^{k-1}\left(1 - \frac{x^{(0)}(k-1)}{x^{(0)}(k)}\right)}. \tag{15}$$

Equation (12) is called the 1-AGO dynamic sequence prediction model.

- Romberg quadrature formula

The Romberg quadrature formula is also called the successively divided and semi-accelerated method. As an extrapolation algorithm, this formula improves the accuracy of the result of integration without increasing the amount of calculation. The Romberg quadrature formula has a more accurate integral approximate value by the weighted average of the approximate values of the trapezoidal formula.

The initial parameter is set at $k = 1$, and the integral is calculated roughly as

$$r_{1,1} = \frac{b-a}{2}[f(a) + f(b)]. \tag{16}$$

The value of the next position is calculated as

$$r_{k+1,1} = \frac{1}{2}\left[r_{k,1} + \frac{b-a}{2^{k-1}}\sum_{n=1}^{M} f\left(a + (2n-1)\frac{b-a}{2^k}\right)\right], M = 2^{k-1}. \tag{17}$$

A recursive formula is used for the calculation

$$r_{k+1,m} = \frac{4^{m-1}r_{k+1,m-1} - r_{k,m-1}}{4^{m-1} - 1}. \tag{18}$$

If the difference between $r_{k+1,k+1}$ and $r_{k,k}$ meets the predetermined accuracy $\varepsilon$, then the calculation stops. Otherwise, $k$ goes up by 1, and then we proceed to Equation (17).

$$R = \begin{bmatrix} r_{1,1} & & & & & \\ r_{2,1} & r_{2,2} & & & & \\ r_{3,1} & r_{3,2} & r_{3,3} & & & \\ r_{4,1} & r_{4,2} & r_{4,3} & r_{4,4} & & \\ r_{5,1} & r_{5,2} & r_{5,3} & r_{5,4} & r_{5,5} & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}. \tag{19}$$

The Romberg algorithm is expressed as the lower triangular matrix of Equation (18), and it can be extended infinitely downward and backward. The optimal approximate solution for a definite integral is $r_{k,k}$, which is the lower right item calculated in Equation (19). The calculation of the Romberg quadrature formula recurs in Equations (17) and (18).

- Building a metabolic model

In the traditional modeling process, the prediction accuracy of the model is reduced continuously as the system status changes by selecting a fixed length of the original data to build a model. Therefore, the higher prediction accuracy of the model should be ensured by updating the modeling data. This process of building a model is dynamic. The new data will be added to the modeling data sequence, and the oldest data in the modeling data will be deleted to reconstruct the model when a new data appears, the length of the modeling data of the construction of model remains unchanged. The optimal length of the modeling data will be discussed in the experiment.

- Construction of RMGM (1,1) model

The RMGM(1,1) model reduces error by calculating the area of curve of the 1-AGO dynamic sequence model in each interval by using the Romberg quadrature formula.
**Step 1** : Select a fixed-length raw data sequence $x^{(0)}(k)$ and calculate 1-AGO sequence $x^{(1)}(k)$, $k = 1, 2, 3, ..., n$.
**Step 2** : Let $a = 1, b = 2$, locate the interval $[a, b]$, combine Equations (13)–(15) in this interval, and calculate the required parameters $\alpha$, $A$, and $B$ of Equation (12) to obtain the formula $f(k) = x^{(1)}(k) = Ae^{\alpha(k-1)} + B$.
**Step 3** : Calculate $r$ of the integral area of the interval $[a, b]$ in accordance with Equations (16)–(18). The ordinate value of the background value curve required by Equations (16)–(18) is given by Step 2, Let $z^{(1)}(k) = r$.
**Step 4** : if $b < n$, then proceed to Step 2, and let $a$ and $b$ increase by 1. The sequence of the background values $z^{(1)}(k)$ is obtained until $b = n$.
**Step 5** : The time response function is obtained by combining Equations (7)–(10).
**Step 6** : Newly generated data are added to the original data sequence $x^{(0)}(k)$, and the first old data are deleted at the same time. A new data sequence $x^{(0)}(k)$ is generated by keeping the length unchanged, and then return to Step 1.

The predicted value $\hat{x}(n+1)$, $\hat{y}(n+1)$, and $\hat{z}(n+1)$ is obtained by establishing the RMGM(1,1) model in three coordinate series $(x(1), x(2), ..., x(n))$, $(y(1), y(2), ..., y(n))$, and $(z(1), z(2), ..., z(n))$. The prediction coordinates $\hat{p}(n+1) = (\hat{x}(n+1), \hat{y}(n+1), \hat{z}(n+1)$ are obtained accordingly.

### *3.3. Local Path Adjustment*

The local path of the UAV is adjusted according to the prediction results of the dynamic flight obstacle in the previous step. Path alignment schemes are divided into two approaches. First, the UAV tries to keep to the original planned flight path when it encounters threats, that is, the UAV only changes its speed. At this point, the path of the dynamic obstacle is not coincident from that of the UAV, and they will collide at the intersection point. The UAV makes a local deviation to avoid dynamic obstacles when the deceleration strategy is not feasible. This means that the path of the UAV and dynamic obstacle are coincident, and many points are closer and threatening on both of the paths. The UAV returns to the original path when it eludes the dynamic obstacles successfully.

In Figure 7, the space is divided into two regions by making vertical lines perpendicular to the trajectory of the UAV according to the collision position. The dynamic flight obstacle in Region 1 is called the co-directional obstacle, and the dynamic flight obstacle in Region 2 is called the contralateral obstacle. The coordinates of 1 s indicate where the UAV has arrived in the path, and the coordinates of 2 s in the path represent where the UAV is going to be. According to the results of the prediction model, the UAV will collide with the dynamic flight obstacle at the position of 2 s. The coordinate of 2's is added to the path instead of the coordinate of 2 s after the local path is adjusted. According to the different flight trajectories of dynamic flight obstacles, we analyze four collision situations and suggest a local path adjustment strategy for each collision.

- In Figure 7a, the UAV verifies whether the adoption of a deceleration strategy is feasible, that is, the coordinate of 2's is added to the original path instead of the coordinate of 2 s. $R$ refers to the distance recorded between the UAV and the obstacle in every second. $R$ should be greater than the sum of the radius of the envelopment circle of UAV and dynamic flight obstacle if no collision occurs. The path of the codirectional obstacle is not coincident with the path of the UAV and the $R$ value is far greater than the sum of the radius of the envelope circle of UAV and the dynamic flight obstacle at 2's. Therefore, the UAV will adopt this strategy wherein the position at 2's replaces the position at 2 s.
- In Figure 7b, the path of the dynamic obstacle is coincident of the UAV. If the deceleration strategy is adopted, then the $R$ value is less than the sum of the radius of the envelope circle of the UAV and the dynamic flight obstacle, and the collision will still occur. A decentralized strategy is adopted, the position at the 2 s is offset at the 2's

while keeping $R$ greater than the sum of the radius of the envelope circle of the UAV and the dynamic flight obstacle. The UAV returns to the original path at 3 s after avoiding the dynamic obstacle. For the opposite dynamic obstacle, verifying whether the deceleration strategy is feasible first is necessary.

- In Figure 7c, if the deceleration strategy is adopted, then $R$ is greater than the sum of the radius of the UAV and the dynamic flight obstacle envelope circle at the 2's, and the position at the 2's instead of the position at the 2 s add the path. Hence, obstacle avoidance is successful.
- In Figure 7d, the path of the UAV and dynamic obstacle are coincident, and $R$ will be less than the sum of the radius of the enveloping circle of UAV and dynamic flight obstacle if the deceleration strategy is adopted. Thus, the decentralized strategy is carried out. The position at the 2's is the offset position, thereby, making the $R$ value greater than the sum of the radius of the envelopment circle of the UAV and the dynamic flight obstacle. The UAV returns to the original path at 3 s after successful obstacle avoidance.

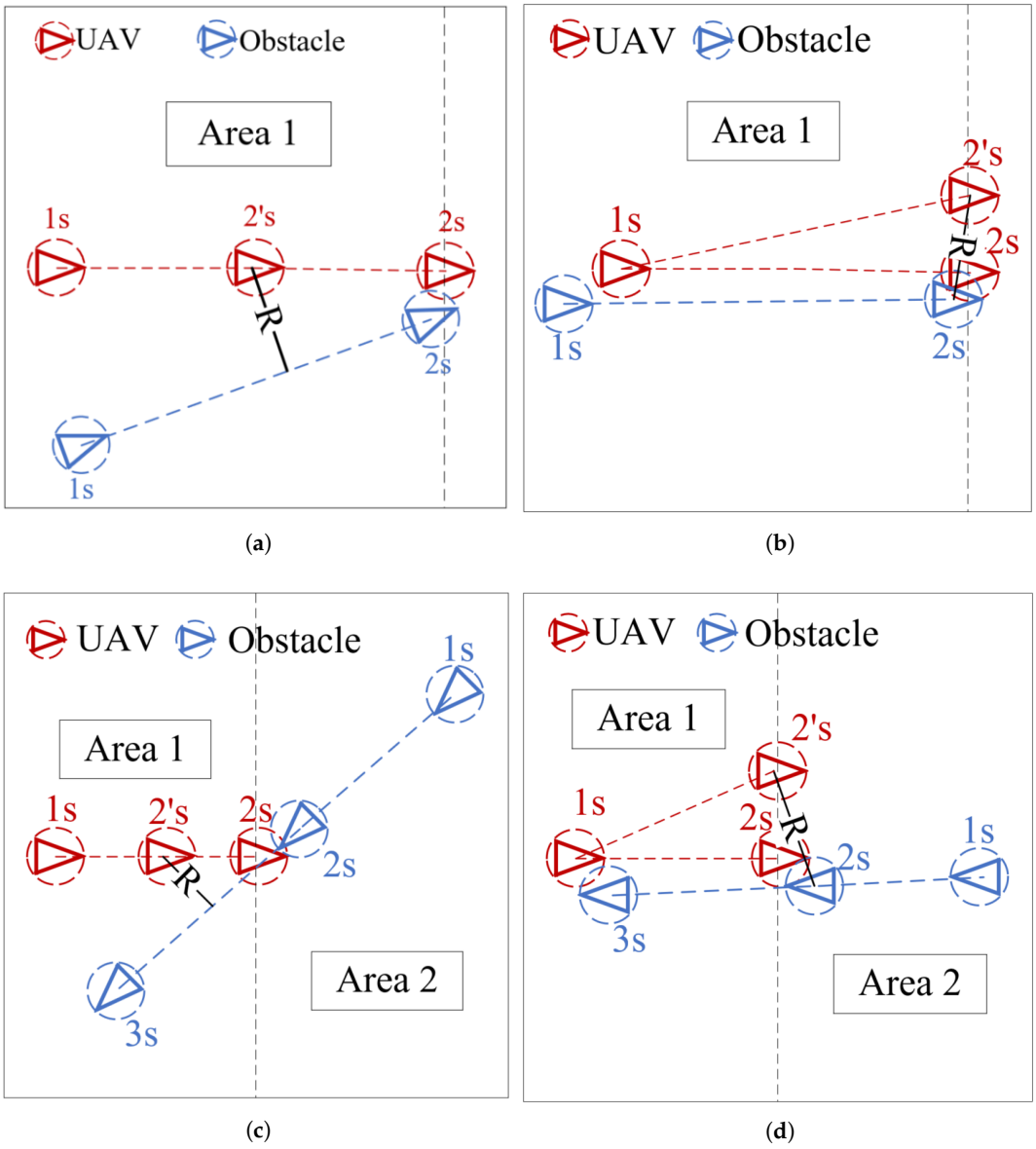



**Figure 7.** $R$ refers to the distance recorded between the UAV and the obstacle in every second. (**a**,**b**) show the obstacle avoidance of UAV when the obstacle is co-directional; (**c**,**d**) show the obstacle avoidance of UAV when the obstacle is contralateral.

## 4. Experiments and Discussions

In this section, simulation experiments are carried out on an IntelCPU2.8G 16 GB memory computer using a MATLAB-R2016B environment. UAVs are low-energy IoT devices, which can save energy through effective energy management methods [44,45]. Energy saving can also be achieved by reducing unnecessary maneuver of UAVs. The three dimensional space is transformed into two dimensional plane to avoid the change of UAVs in vertical direction [46] in this paper. This section is divided into three parts.

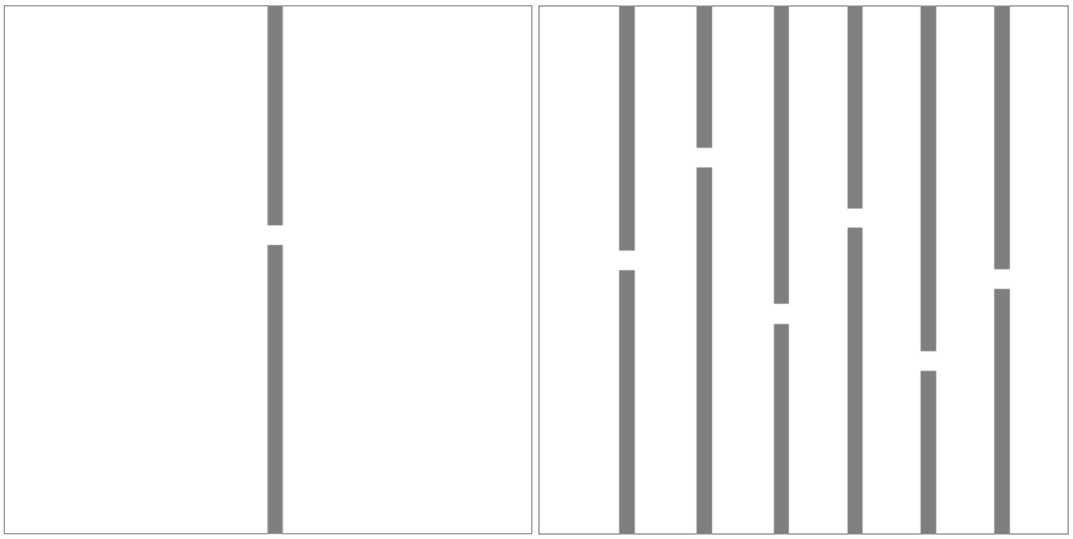
In the first part, three different maps were created with the size 800 × 800 (Figure 8). The start and goal point coordinates were set to (50,50) and (750,750), respectively. The step size of BS-RRT and narrow passage width were set to 30. The global path was generated in a map with narrow passages by the BS-RRT algorithm. Then, the cyclic pruning algorithm was used to optimize the generated global path. We validate the performance of BS-RRT algorithm and cyclic pruning algorithm by comparing BS-RRT with RRT, RRT-Connect, P-RRT, 1-0 Bg-RRT [28], and RRT$^*$. We set $P = 0.5$ in P-RRT. For the RRT$^*$ algorithm, the earliest time to finish the feasible path planning was set as the criterion.

In the second part, we predicted the trajectories of dynamic flight obstacles by using RMGM(1,1) models to prepare for local path adjustments. The accuracy of the RMGM(1,1) model was compared with other models. First, the ECGM(1,1) model of exponential curve optimization, PCGM(1,1) model of polynomial curve optimization in [39], and original GM(1,1) model were compared with RMGM(1,1). Second, the metabolic MGM(1,1) and GDF [47] were compared with RMGM(1,1). Third, the local path adjustment scheme was implemented according to the prediction results of the RMGM(1,1) model.

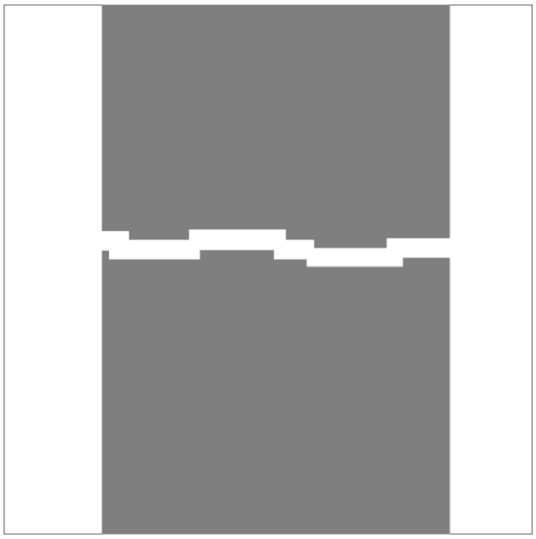
In this section, the dynamic flight obstacle was set as the same model as the UAV, and the radius of the envelope ball is 0.5. Considering the unforeseeable factors in the actual UAV flight and the errors caused by the prediction model, the setting of the $R$ value was generally greater than the sum of the radius of the envelope circle between the UAV and the dynamic obstacle, and $R$ was set to 2. Some of the necessary parameters are summarized in Table 1.

**Table 1.** The main parameters in the experiment.

| Experiment | Parameters |
|---|---|
| Runtime environment | IntelCPU2.8G 16 GB memory computer using MATLAB-R2016B |
| Map size | 800 × 800 |
| Number of maps | 3 |
| Start point in the map | (50,50) |
| Goal point in the map | (750,750) |
| Narrow passage width | 30 |
| Envelope sphere radius of UA V and dynamic flight obstacle | 0.5 |
| Narrow passage width | 30 |
| Distance R | 2 |



(**a**) Map 1 (**b**) Map 2



(**c**) Map 3

**Figure 8.** Three different maps with narrow passages. Map 2 has five more narrow passages than Map 1, and the narrow passage in Map 3 is longer than that in Map 2.

### *4.1. BS-RRT and the Circular Pruning Algorithm*

More detailed data on Figure 9 are documented in Table 2. The narrow passages in Figure 9c are few and short. The path generated by the BS-RRT algorithm was close to the optimal path before pruning, and a path length reduction of 5.606% was achieved after pruning by using the cyclic pruning algorithm. As shown in Figure 9f,i, the number and length of narrow channels increased, and the path generated by the BS-RRT algorithm was more tortuous and different from the optimal path. The effect of the cyclic pruning algorithm is obvious, and the path reductions were 24.898% and 17.077%.

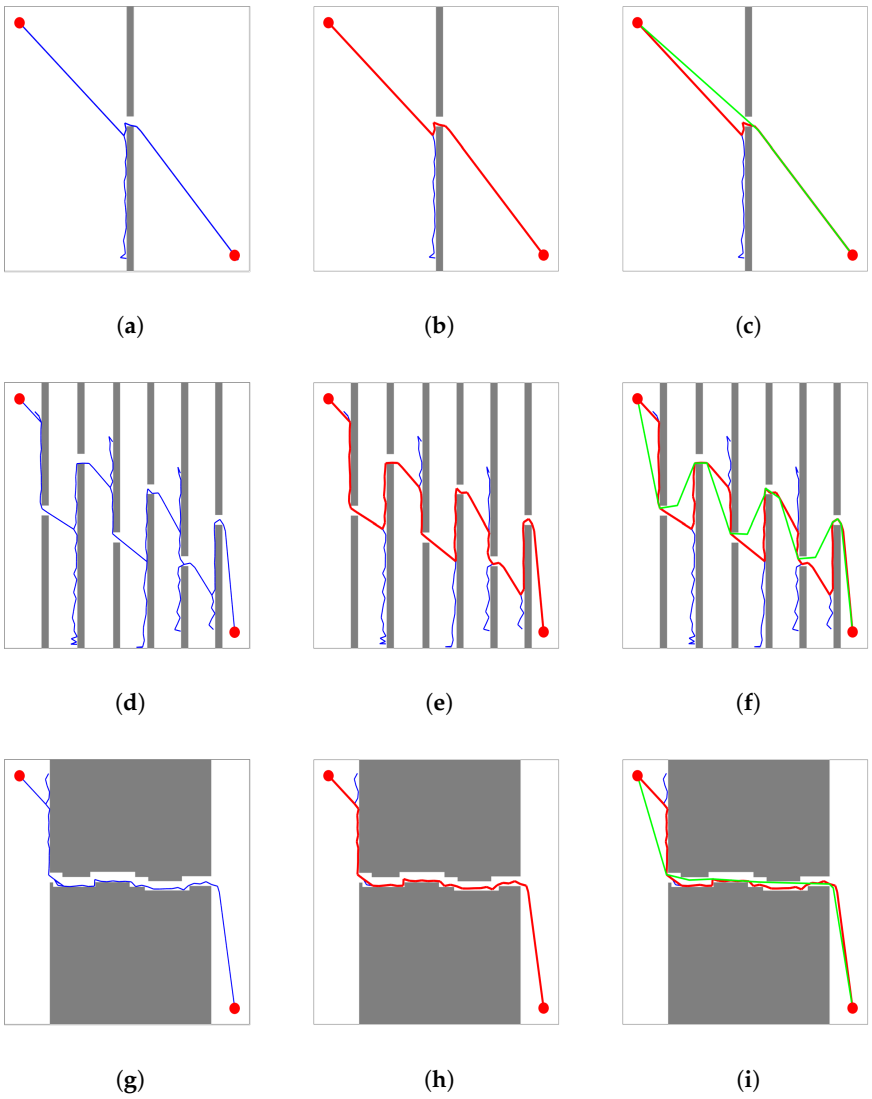



**Figure 9.** Planning results of BS-RRT and cyclic pruning algorithms in the three maps. The blue line refers to the path of the BS-RRT algorithm, the red line represents the backtracking path after BS-RRT connects the starting point and the end point, and the green line corresponds to the final path after cyclic pruning. (**a**,**d**,**g**) show the running results of BS-RRT algorithm in three maps; (**b**,**e**,**h**) show the backtracking path of BS-RRT algorithm in three maps; (**c**,**f**,**i**) show the final path generated after pruning in three maps.

**Table 2.** Comparison of path changes before and after pruning with cyclic pruning algorithms.

| Figure | Before Pruning | After Pruning | Percentage Reduction |
|---|---|---|---|
| Figure 9c | 1054.345 | 995.238 | 5.606% |
| Figure 9f | 2369.049 | 1779.210 | 24.898% |
| Figure 9i | 1505.181 | 1248.134 | 17.077% |

In Figure 10, RRT, RRT-Connect, P-RRT, 1-0 Bg-RRT, and RRT* carry out global path planning in Maps 1, 2, and 3. The final backtracking paths are relatively tortuous when the number of narrow passages increases.

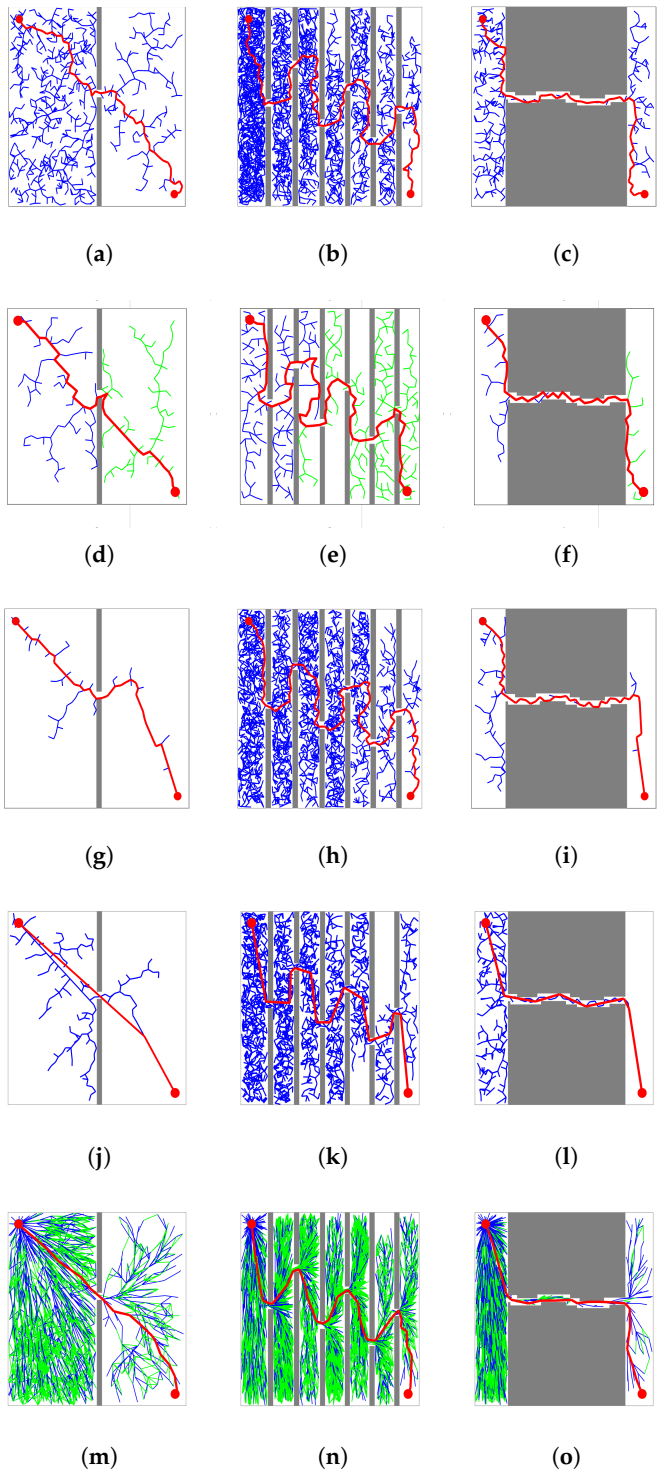



**Figure 10.** (**a**–**c**) show the path planning results of RRT in three maps; (**d**–**f**) are the path planning results of RRT-Connect in three maps; (**g**–**i**) are the path planning results of P-RRT in three maps; (**j**–**l**) are the path planning results of 1-0 Bg-RRT in three maps; and (**m**–**o**) are the path planning results of RRT* in three maps. The red line corresponds to the final path.

In Figure 11a–c, each run time is recorded for BS-RRT, RRT, RRT-Connect, P-RRT, 1-0 Bg-RRT, and RRT* of path planning in Maps 1, 2, and 3 in 50 experiments. Intuitively, except for BS-RRT, the stability of the other algorithms is poor, and the time fluctuates greatly with respect to the path planning of each experiment. Moreover, as shown in Figure 11b, when the narrow passages increase, the path planning time of the five algorithms extends

significantly, except for BS-RRT, and the fluctuation range of the algorithm planning time is larger than that of BS-RRT.

In Figure 11d, the average path length of 50 times the path planning of the six algorithms is recorded. The paths planned by the BS-RRT algorithm are all optimal. Table 3 is drawn based on computer simulation results. According to Table 3, in terms of the average planning time, the BS-RRT algorithm had the fastest convergence rate. In the three maps, the convergence times were 0.722, 1.159, and 0.757 s. With the increase in the number of narrow passages, the convergence time of BS-RRT rises slightly, and the length of the narrow passages does not affect the convergence time of the algorithm. The convergence time of the other algorithms varies greatly with the increase in the number of narrow passages.

In Map 2, the mean convergence times of RRT, RRT-Connect, P-RRT, 1-0 Bg-RRT, and RRT* were 20.02, 9.152, 33.216, 21.898, and 42.886 s. In terms of the maximum time difference of the algorithm, the time fluctuations of BS-RRT algorithm were the least among the three maps, which were 0.097, 0.162, and 0.136 s. Similarly, the volatility of other algorithms increased with the increase in the number of narrow passages. The maximum fluctuation times of the path planning with other algorithms in Map 2 were 34.189, 18.563, 91.061, 28.304, and 89.655s. Although the 1-0 Bg-RRT planning path was more efficient in the state space with a narrow passage, the efficiency of the algorithm decreased with the increase of the number of narrow passages. In terms of the average planned path length, the path was optimized by the circular pruning algorithm to be the shortest in the three maps.

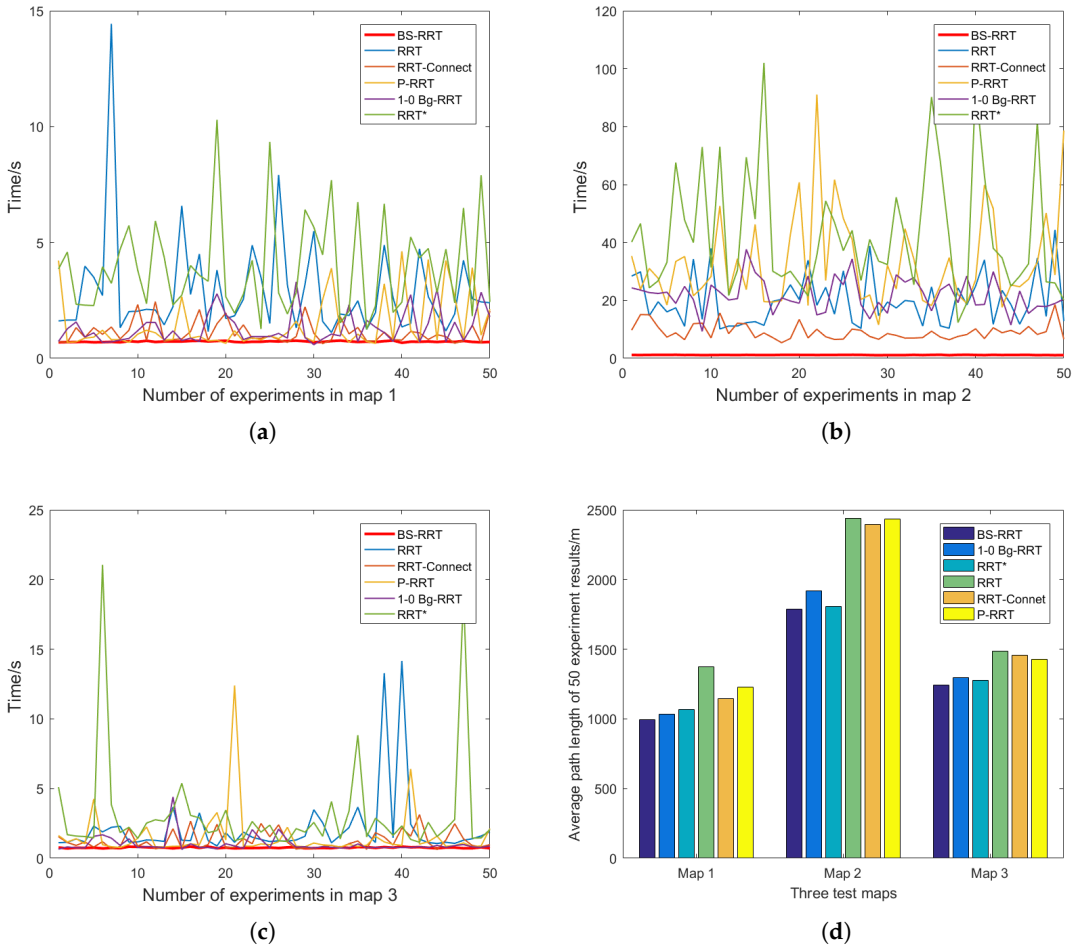



**Figure 11.** (**a**–**c**) show the run time for each of the 50 experiments for BS-RRT, RRT, RRT-Connect, P-RRT, 1-0 Bg-RRT, and RRT* in the three maps; (**d**) shows the average path length of 50 experiments using the six algorithms.

**Table 3.** Time data from 50 experiments for the six algorithms.

| Map | Algorithm | Average Planning Time/s | Maximum Time Difference/s | Average Path Length |
|---|---|---|---|---|
| Map 1 | BS-RRT | 0.722 | 0.097 | 994.858 |
| | RRT | 2.905 | 13.324 | 1374.577 |
| | RRT-Connect | 1.121 | 2.449 | 1146.323 |
| | P-RRT | 1.466 | 4.621 | 1225.728 |
| | 1-0 Bg-RRT | 1.304 | 2.628 | 1033.617 |
| | RRT* | 3.951 | 9.050 | 1068.216 |
| Map 2 | BS-RRT | 1.159 | 0.162 | 1787.360 |
| | RRT | 20.020 | 34.189 | 2439.182 |
| | RRT-Connect | 9.152 | 18.563 | 2396.566 |
| | P-RRT | 33.216 | 91.061 | 2433.546 |
| | 1-0 Bg-RRT | 21.898 | 28.304 | 1920.761 |
| | RRT* | 42.886 | 89.655 | 1807.956 |
| Map 3 | BS-RRT | 0.757 | 0.136 | 1242.084 |
| | RRT | 2.127 | 13.274 | 1486.449 |
| | RRT-Connect | 1.270 | 3.126 | 1457.493 |
| | P-RRT | 1.567 | 12.392 | 1429.350 |
| | 1-0 Bg-RRT | 1.008 | 3.768 | 1298.581 |
| | RRT* | 3.115 | 19.911 | 1274.754 |

Table 4 records the time taken for BS-RRT to plan 50 times in the four maps shown in Figure 12. The convergence time and fluctuation of the algorithm slightly increased when the number of obstacles gradually rose, and the maximum fluctuation was 0.2754 s. The average convergence times of the algorithm in the four maps were 0.8169, 0.8556, 0.9452, and 0.9456 s. These results indicate that, in ordinary maps, BS-RRT also maintains good convergence speed and stability.

**Table 4.** Time data from 50 experiments for BS-RRT algorithms in common maps.

| Figure | Maximum Time Difference/s | Minimum Planning Time/s | Average Planning Time/s | Maximum Planning Time/s |
|---|---|---|---|---|
| Figure 12a | 0.1277 | 0.7388 | 0.8169 | 0.8665 |
| Figure 12b | 0.1367 | 0.7632 | 0.8556 | 0.8999 |
| Figure 12c | 0.1530 | 0.8429 | 0.9452 | 0.9959 |
| Figure 12d | 0.2754 | 0.8387 | 0.9456 | 1.1141 |

Table 5 records the detailed data of the path changes before and after pruning using the circular pruning algorithm in the four maps in Figure 12. The path length and the tortuous degree generated by BS-RRT increased with the number of obstacles. In Figure 12c, the zigzag degree of the path was the largest when the obstacle was 50, and the pruning effect, which was reduced by 11.46%, was the most obvious. As shown in Figure 12a, the path generated by BS-RRT was close to the optimal path with a low tortuous degree, and the path length was reduced by 5.33% after pruning. In Figure 12b,d, the path tortuous degree is similar in the two figures, and the path length decreased by 9.37% and 9.75%, respectively.

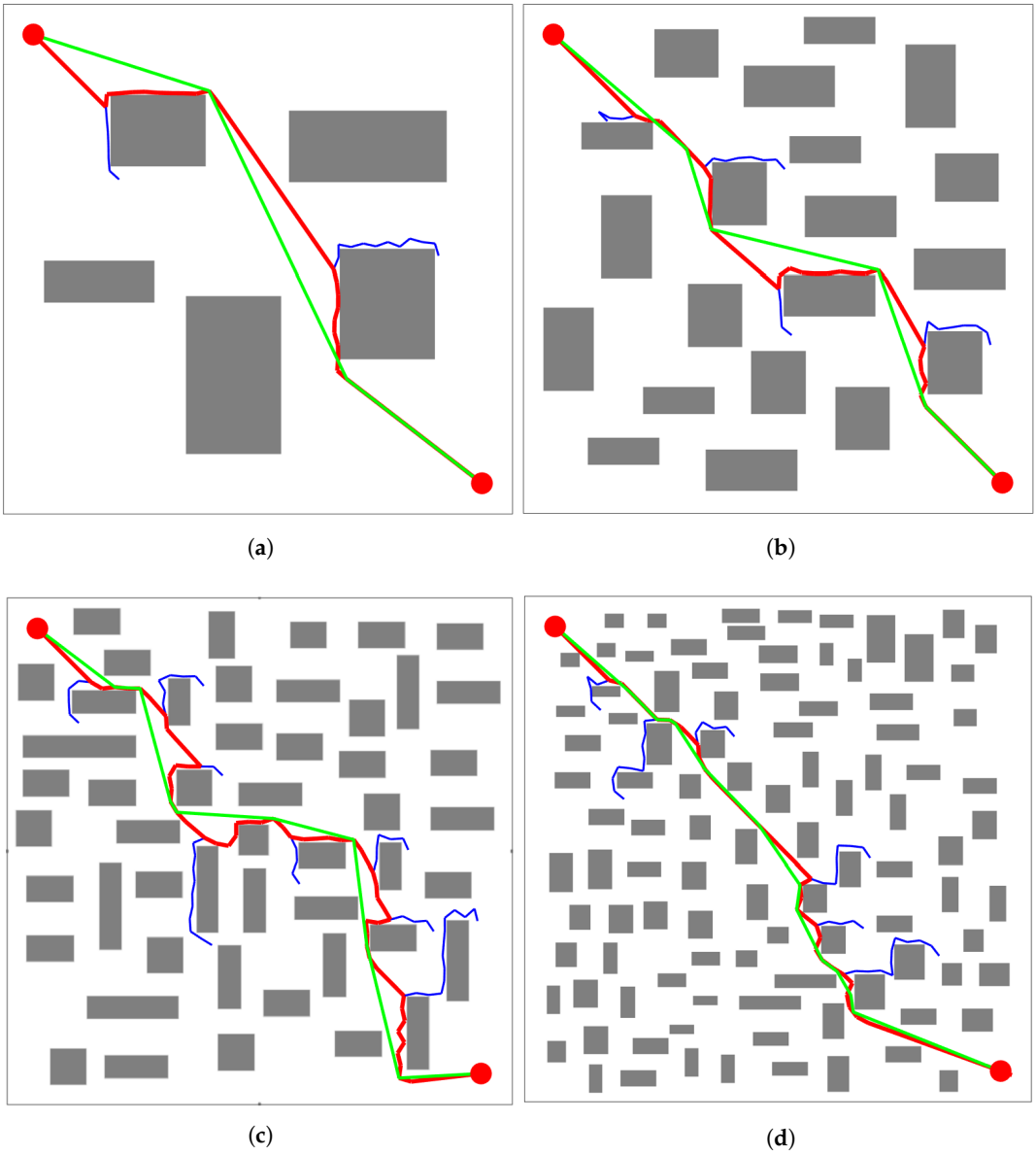


**Figure 12.** Path planning of the BS-RRT algorithm in common maps and there are 5, 20, 50 and 100 obstacles in (**a**–**d**) respectively.

**Table 5.** Comparison of path changes before and after pruning with cyclic pruning algorithms.

| Figure | Before Pruning | After Pruning | Percentage Reduction |
|---|---|---|---|
| Figure 12a | 1105.35 | 1046.38 | 5.33% |
| Figure 12b | 1168.98 | 1059.44 | 9.37% |
| Figure 12c | 1199.60 | 1062.18 | 11.46% |
| Figure 12d | 1228.31 | 1108.53 | 9.75% |

### *4.2. RMGM*(1,1) *Model Accuracy Comparison*

To intuitively express the accuracy of the model, we define the following formula. Three axes of coordinates are split to build the original data sequence

$$P^{(0)}(k) = (X^{(0)}(k), Y^{(0)}(k), Z^{(0)}(k)). \tag{20}$$

where

$$\begin{cases} \text{Axis}x : X^{(0)} = \left\{ x^{(0)}(k) \right\} \\ \text{Axis}y : Y^{(0)} = \left\{ y^{(0)}(k) \right\} \\ \text{Axis}z : Z^{(0)} = \left\{ z^{(0)}(k) \right\} \end{cases}, k = 1, 2, ..., n. \tag{21}$$

The coordinates obtained by the prediction of the three coordinate sequences are the fitting values

$$\hat{P}(k+t) = \left( \hat{X}^{(0)}(k+t), \hat{Y}^{(0)}(k+t), \hat{Z}^{(0)}(k+t) \right). \tag{22}$$

where

$$\begin{cases} \text{Axis}x : \hat{X}^{(0)} = \left\{ \hat{x}^{(0)}(k+t) \right\} \\ \text{Axis}y : \hat{Y}^{(0)} = \left\{ \hat{y}^{(0)}(k+t) \right\} \\ \text{Axis}z : \hat{Z}^{(0)} = \left\{ \hat{z}^{(0)}(k+t) \right\} \end{cases}, t = 1, 2, ..., m. \tag{23}$$

To evaluate the accuracy of the prediction effect of the model, we define the compound position error as

$$E_{\mathrm{p}} = \sqrt{E_x^2 + E_y^2 + E_z^2}. \tag{24}$$

$E_x, E_y, E_z$ are the position errors in the three directions and are defined as follows

$$\begin{cases} E_x = |\hat{X} - X| \\ E_y = |\hat{Y} - Y| \\ E_z = |\hat{Z} - Z| \end{cases}. \tag{25}$$

$\widehat{X}, \widehat{Y}$, and $\widehat{Z}$ are the coordinate components of the three directions calculated by the model and are the three directions of the corresponding real coordinates for the moving target

$$\begin{cases} E_{\max} = \max\limits_{1}^{N}(E_{\mathrm{p}}) \\ E_{\min} = \min\limits_{1}^{N}(E_{\mathrm{p}}) \\ E_{\mathrm{eq}} = \sum\limits_{1}^{N}(E_{\mathrm{p}}) \end{cases}. \tag{26}$$

The model length $n$ determines the metabolic rate and accuracy of the model. Thus, discussing the model length $n$ is necessary. According to [48,49], the length of the model is usually between 3 and 5. In Figure 13, we predict 20 sets of coordinates and calculate the error by comparing them with the real coordinates. The red dot is closest to the origin, and the prediction error is the smallest when the model length is 3. Therefore, we chose the model length of $n$ = 3 as the optimal parameter value.

The X axis in Figure 14 shows the real-time position of the moving object in the X direction. Similarly, the Y and Z axes represent the real-time position of the moving object in the Y and Z directions, respectively. The trajectories predicted by RMGM, ECGM, PCGM, and GM, as well as the actual trajectories of moving objects, are shown in Figure 14. Except for the red curve that represents RMGM, the other curves have large deviations from the real trajectory. To show the trajectory prediction accuracy of the different methods clearly, Figure 15 depicts the combined position errors calculated by RMGM, ECGM, PCGM, and GM.

Yellow, red, green, and blue bars represent the RMGM, ECGM, PCGM, and GM errors, respectively. The errors of RMGM are far less than those of the other models. Table 6 is drawn according to the computer simulation results, According to Table 6, the composite position errors calculated by RMGM were much smaller than those of the other models. The average errors of ECGM, PCGM, and GM were 120.17-, 120.32-, and 121.38-times that of RMGM, respectively.

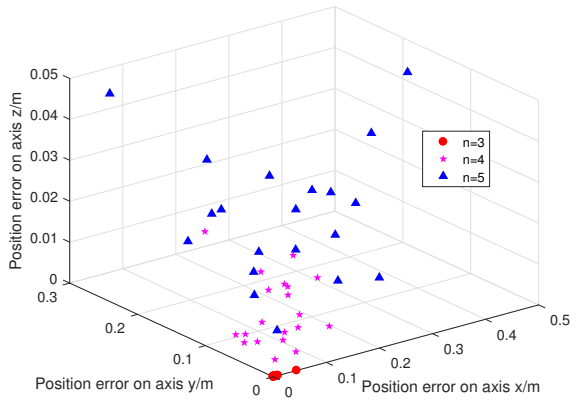



**Figure 13.** The error between the predicted and the actual coordinates at n = 3, 4, and 5.

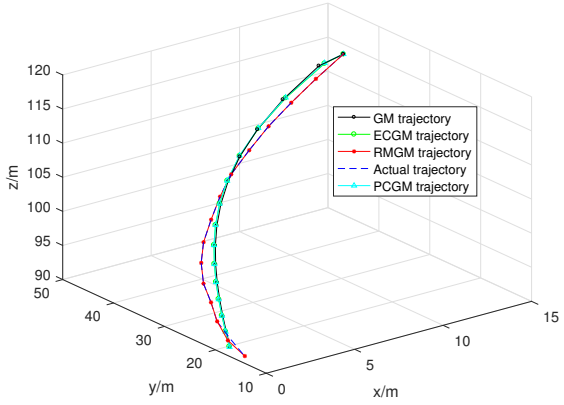



**Figure 14.** Predicted trajectories of the GM, ECGM, PCGM, and RMGM models.

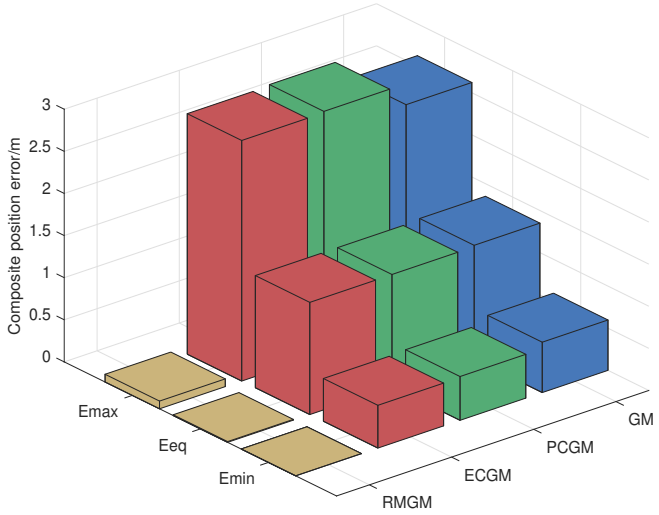



**Figure 15.** Composite position error calculated by the GM, ECGM, PCGM, and RMGM models in the experiment.

**Table 6.** Composite position error.

| Method | Emin | Eeq | Emax |
|---|---|---|---|
| RMGM | 0.0002 | 0.0112 | 0.1238 |
| ECGM | 0.4897 | 1.3459 | 2.6561 |
| PCGM | 0.4985 | 1.3476 | 2.6816 |
| GM | 0.5676 | 1.3595 | 2.5258 |

In Figure 16, we compare the trajectory prediction performance of RMGM, GDF, and MGM models, and the prediction trajectories of the three models are close to the real trajectory. Distinguishing the advantages from the disadvantages in the trajectory diagram is difficult. In Figures 17 and 18, the fitting position errors of the three models can be shown clearly. Furthermore, the errors of RMGM are the minimum. In Figure 17, the variation trend of the RMGM error is opposite to that of GDF. The error of the GDF model increased when the background value error was small because the GDF used an error correction term to reduce the model error.

Moreover, given that the error correction term is a prediction model, the variation trend of the GDF model's error was hysteretic compared with the RMGM. In Figure 18, considering that the RMGM model only optimized the background value error in the modeling process, the error followed the same trend as that of the MGM model. Figure 19 and Table 7 show a comparison of the compound position errors calculated by the three methods. The average synthetic position errors of MGM, GDF, and RMGM were 0.03437, 0.02435, and 0.00273, respectively. The average position errors of MGM and GDF models were 12.59 times and 8.92 times the RMGM models, respectively. Clearly, RMGM had better trajectory prediction performance than MGM and GDF.

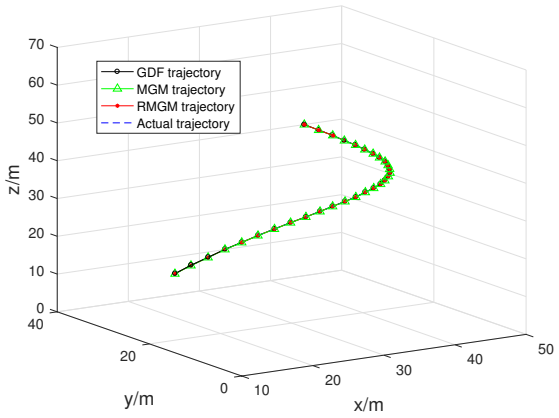



**Figure 16.** Trajectory prediction of the GDF, MGM, and RMGM models.

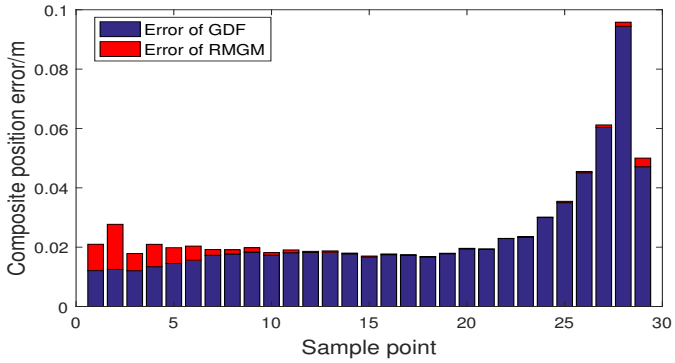



**Figure 17.** Combined position errors of GDF and RMGM.

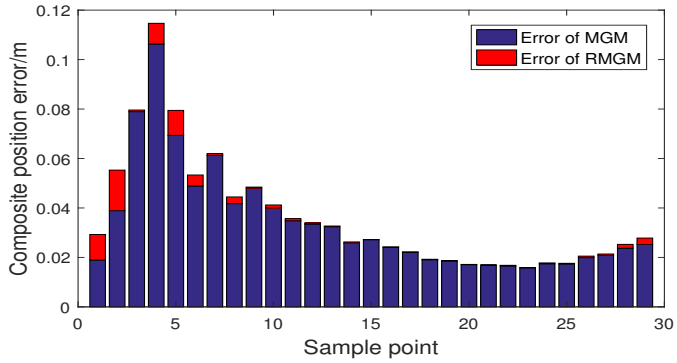



**Figure 18.** Combined position errors of MGM and RMGM.

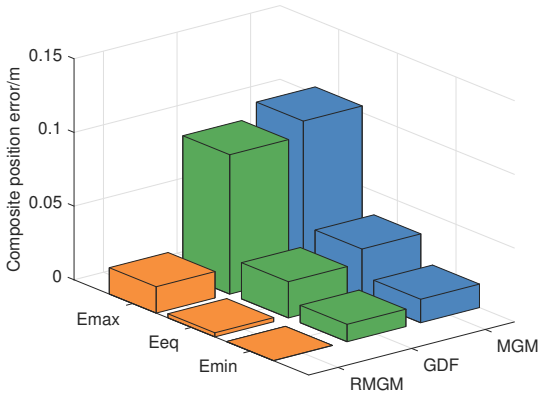



**Figure 19.** Comparison of the prediction errors of GDF, MGM, and RMGM.

**Table 7.** Comparison of the composite position error calculated by three methods.

| Method | Emin | Eeq | Emax |
|---|---|---|---|
| MGM | 0.01531 | 0.03437 | 0.11419 |
| GDF | 0.01163 | 0.02435 | 0.09092 |
| RMGM | 0.00005 | 0.00273 | 0.01587 |

### *4.3. Local Path Adjustment*

In Figure 20, The local path adjustment strategy is demonstrated on an 80 × 80 size map. In Figure 20c, the local path adjustment of the same direction obstacle and the opposite obstacle are classified into one category because they have the same deceleration strategy, and the local adjustment of the opposite obstacle is taken as an example.

Table 8 records the coordinate information of the UAV and the dynamic flying obstacle in Figure 20a at every moment. The coordinate of the next moment about the dynamic flying obstacle was recorded by using the RMGM(1,1) model. $R1$ refers to the distance between the predicted coordinates of the dynamic flying obstacle and the UAV in the next moment. $R2$ is the distance between the dynamic flying obstacle and UAV after the UAV adopts the deceleration strategy in the next moment.

The adjusted coordinates indicated that the original coordinates at the next moment are replaced by the adjusted coordinates. $R3$ refers to the distance between the coordinates of the dynamic flying obstacle and coordinates after the UAV adopts a change strategy. The safe distance of $R1$, $R2$, and $R$ was 2. According to the prediction results, $R1$ was 1.2 when the UAV flies to the coordinates at 2 s, that is, collision occurred between the UAV and the dynamic flying obstacle at 3 s.

First, a deceleration strategy was validated. $R2$ was 0 if the UAV adopted the deceleration strategy, suggesting that the path of the UAV coincided with the trajectory of the dynamic obstacle. A decentralized strategy was adopted instead of the deceleration strategy. The distance between the offset coordinates of the UAV and the predicted coordinates of the dynamic flight obstacle was greater than 2. Thus, the coordinate (36, 40) in the path was replaced by the coordinate (37.2, 38). $R3$ was 2.3 at 3 s, and the obstacle avoidance was successful.

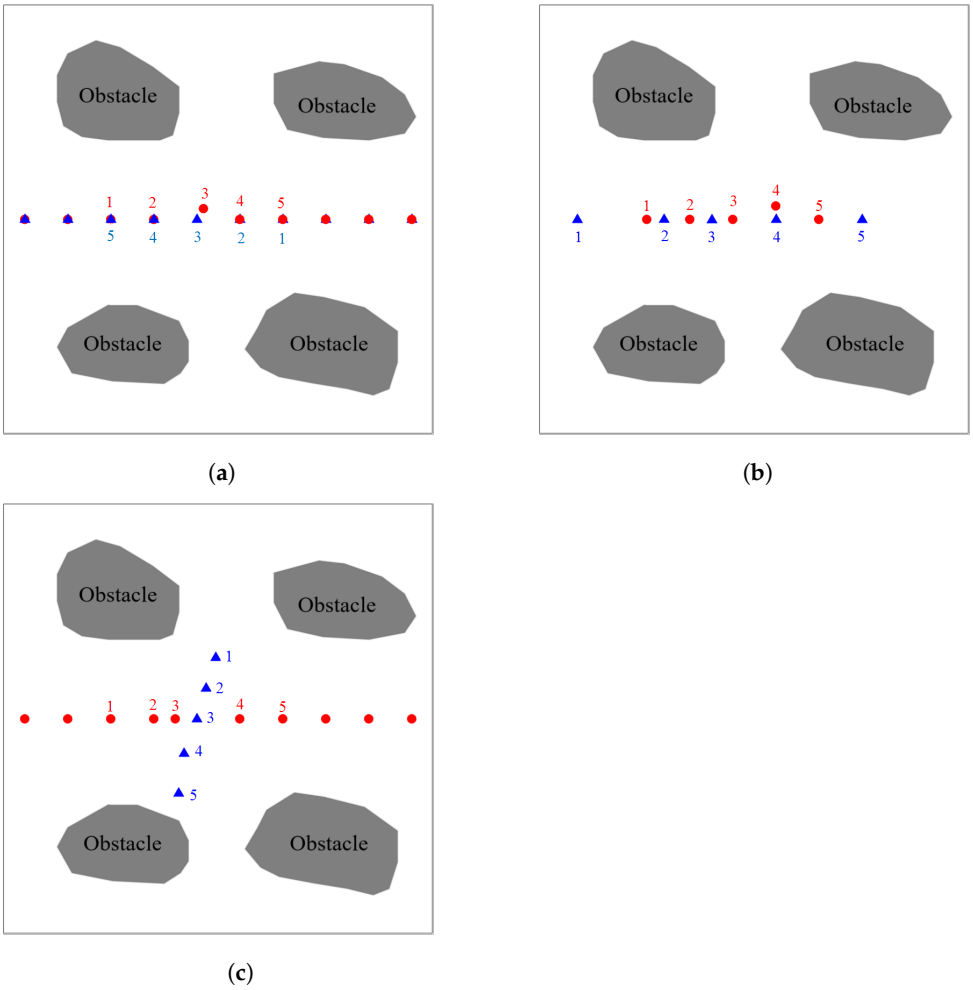



**Figure 20.** Dynamic flight obstacles are represented as blue triangles, the red circle is the UAV, and the number of the corresponding color is the coordinate point of the corresponding time. (**a**,**b**) show the local path adjustment for the opposite and codirectional obstacles, respectively. (**c**) shows the deceleration strategy of the UAV.

The coordinate information of the UAV and dynamic flight obstacle in Figure 20b at each moment is recorded in Table 9. $R1$ was 0.2 at 3 s, suggesting that a collision between the UAV and dynamic flight obstacle occur at 4 s. First, the feasibility of a deceleration strategy was verified. $R2$ was 0 if the UAV adopted the deceleration strategy. Thus, a decentralized strategy was adopted, and the coordinates (43.8, 38) were added to the path of the UAV instead of (44, 40) at 4 s. $R3$ was 2.3 when the UAV flew to the coordinates (43.8, 38) at 4 s, and the obstacle avoidance was successful.

The coordinate information of teh UAV and dynamic flight obstacle in Figure 20c at each moment is recorded in Table 10. Considering that the trajectories of UAVs and dynamic flying obstacles are not coincident, the UAV only needs to avoid dynamic flight obstacles by adopting a deceleration strategy. Oncoming obstacles are used as an example.

Collision occureed at 3 s when $R1$ was 1. The feasibility of the deceleration strategy was verified. $R2$ was 4.02 at 3 s when the deceleration strategy was carried out. Hence, $t$ no collisions occurred at 3 s. Then, the coordinates were added into the path after the deceleration strategy, and the obstacle avoidance was successful.

**Table 8.** Coordinate information of the local path adjustment in Figure 20a.

| Time | UAV Coordinates | Obstacle Coordinates | Prediction Coordinates | R1 | R2 | Adjusted Coordinate | R3 |
|---|---|---|---|---|---|---|---|
| 1 s | (20,40) | (52,40) | (45,40) | 17 | | | |
| 2 s | (28,40) | (44,40) | (37.2,40) | 1.2 | 0 | (37.2,38) | 2.3 |
| 3 s | (36,40) | (36,40) | (29.4,40) | 14.6 | | | |
| 4 s | (44,40) | (28,40) | (21.8,40) | 30.2 | | | |
| 5 s | (52,40) | (20,40) | (14.3,40) | | | | |

**Table 9.** Coordinate information of the local path adjustment in Figure 20b.

| Time | UAV Coordinates | Obstacle Coordinates | Prediction Coordinates | R1 | R2 | Adjusted Coordinate | R3 |
|---|---|---|---|---|---|---|---|
| 1 s | (20,40) | (7,40) | (23.6,40) | 4.4 | | | |
| 2 s | (28,40) | (23,40) | (32.2,40) | 3.8 | | | |
| 3 s | (36,40) | (32,40) | (43.8,40) | 0.2 | 0 | (43.8,38) | 2.3 |
| 4 s | (44,40) | (44,40) | (59.2,40) | 7.2 | | | |
| 5 s | (52,40) | (60,40) | (112,40) | | | | |

**Table 10.** Coordinate information of the local path adjustment in Figure 20c.

| Time | UAV Coordinates | Obstacle Coordinates | Prediction Coordinates | R1 | R2 | Adjusted Coordinate | R3 |
|---|---|---|---|---|---|---|---|
| 1 s | (20,40) | (39.5,28.6) | (37.7,34.3) | 11.25 | | | |
| 2 s | (28,40) | (37.7,34.3) | (35.9,41) | 1 | 4.02 | (32,40) | |
| 3 s | (36,40) | (36,40) | (34.3,46.6) | 11.73 | | | |
| 4 s | (44,40) | (33.6,46.4) | (31.4,53.8) | 24.80 | | | |
| 5 s | (52,40) | (32.6,53.8) | (31.6,62.3) | | | | |

## 5. Conclusions

In this paper, we presented an autonomous flight scheme of UAVs in complex urban environments. This scheme is a composite method that includes global path generation and local path adjustment. In the global path planning, we presented a branching selection RRT(BS-RRT) algorithm to plan the path in urban environment with many narrow passages. The experimental results showed that BS-RRT could plan global paths quickly by branching selection continually. BS-RRT converged faster in narrow passage environments compared with RRT, RRT-Connect, P-RRT, 1-0 Bg-RRT, and RRT$^*$.

The BS-RRT still maintained a fast convergence speed and high stability in the ordinary map with different numbers of obstacles after many experiments. We also proposed a cyclic pruning algorithm to optimize the path generated by BS-RRT. The simulation results show that the cyclic pruning algorithm shortened the path, and the path after pruning could reach the optimal path. The effect of the cyclic pruning algorithm was enhanced with the increase in the tortuous degree of the path. Forecasting and decision making were included in the local path adjustment scheme, and the RMGM(1,1) model was proposed to predict the trajectory of the dynamic flight obstacles.

The real-time performance of the RMGM(1,1) model was guaranteed by constantly adding new coordinates and eliminating old coordinates. The optimal model length was selected to improve the prediction accuracy. Then, RMGM(1,1) was compared with the other trajectory prediction methods (e.g., GM(1,1), ECGM(1,1), PCGM(1,1), MGM(1,1),

and GDF) in the computer simulation. The simulation results show that the trajectory prediction performance of RMGM was superior to the other models. The decision of the UAV was based on the prediction results of the RMGM model. The decision section contained two path adjustment strategies.

If the coordinate position after deceleration was far from the trajectory of the dynamic flight obstacle, then the deceleration strategy was feasible. The use of the deceleration strategy was preferred when the trajectory of the dynamic flight obstacle did not coincide with the path of the UAV. The decentralized strategy was used when the trajectory of the dynamic flight obstacle coincided with the path of the UAV. The results of the prediction model are the premise for both kinds of decision making. The experimental results demonstrated the effectiveness of the local path adjustment scheme.

**Author Contributions:** Conceptualization, J.Z. and J.L.; methodology, J.Z. and J.L.; software, J.Z. and J.L.; validation, J.Z., J.L. and G.S.; formal analysis, J.Z. and J.L.; investigation, J.L.; resources, J.Z.; writing—original draft preparation, J.Z. and J.L.; writing—review and editing, J.Z., J.L. and G.S.; supervision, H.Y., X.F. and G.S.; 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 Jilin Province Education Department projects, grant number JJKH20200802KJ and JJKH20200791KJ.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Acknowledgments:** The authors are grateful to the reviewers in enhancing the clarity and completeness of this article.

**Conflicts of Interest:** The authors have no conflict of interest to declare that are relevant to the content of this article.

## References

1. Garcin, M.; Desmazes, F.; Lerma, A.N.; Gouguet, L.; Météreau, V. Contribution of Lightweight Revolving Laser Scanner, HiRes UAV LiDARs and photogrammetry for characterization of coastal aeolian morphologies. *J. Coast. Res.* **2020**, *95*, 1094–1100. [CrossRef]
2. Tripolitsiotis, A.; Prokas, N.; Kyritsis, S.; Dollas, A.; Papaefstathiou, I.; Partsinevelos, P. Dronesourcing: A modular, expandable multi-sensor UAV platform for combined, real-time environmental monitoring. *Int. J. Remote Sens.* **2017**, *38*, 2757–2770. [CrossRef]
3. Wan, M.; Gu, G.; Qian, W.; Ren, K.; Maldague, X.; Chen, Q. Unmanned aerial vehicle video-based target tracking algorithm using sparse representation. *IEEE Internet Things J.* **2019**, *6*, 9689–9706. [CrossRef]
4. Zheng, Y.; Du, Y.; Sheng, W.; Ling, H. Collaborative human-UAV search and rescue for missing tourists in nature reserves. *INFORMS J. Appl. Anal.* **2019**, *49*, 371–383. [CrossRef]
5. Hart, P.E.; Nilsson, N.J.; Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107. [CrossRef]
6. Stentz, A. The focussed D* algorithm for real-time replanning. In Proceedings of the 14th International Joint Conference on Artificial Intelligence, Montreal, QC, Canada, 20–25 August 1995; Volume 2, pp. 1652–1659
7. Koenig, S.; Likhachev, M. Fast replanning for navigation in unknown terrain. *IEEE Trans. Robot.* **2005**, *21*, 354–363. [CrossRef]
8. Khatib, O. Real-time obstacle avoidance for manipulators and mobile robots. In *Autonomous Robot Vehicles*; Springer: Berlin/Heidelberg, Germany, 1986; pp. 396–404.
9. Liu, A.; Jiang, J. Solving path planning problem based on logistic beetle algorithm search-pigeon-inspired optimisation algorithm. *Electron. Lett.* **2020**, *56*, 1105–1108. [CrossRef]
10. Ok, S.H.; Seo, W.J.; Ahn, J.H.; Kang, S.; Moon, B. An ant colony optimization approach for the preference-based shortest path search. *J. Chin. Inst. Eng. Trans. Chin. Inst. Eng. Ser. A* **2011**, *34*, 181–196. [CrossRef]
11. Kurdi, M.M.; Dadykin, A.K.; Elzein, I.; Ahmad, I.S. Proposed system of artificial Neural Network for positioning and navigation of UAV-UGV. In Proceedings of the 2018 Electric Electronics, Computer Science, Biomedical Engineerings' Meeting (EBBT), Istanbul, Turkey, 18–19 April 2018; pp. 1–6
12. Zhang, Y.; Zhang, Y.; Liu, Z.; Yu, Z.; Qu, Y. Line-of-Sight Path Following Control on UAV with Sideslip Estimation and Compensation. In Proceedings of the 2018 37th Chinese Control Conference, Wuhan, China, 25–27 July 2018; pp. 4711–4716
13. LaValle, S.M. Rapidly-exploring random trees: A new tool for path planning. *Comput. Sci. Dept. Oct.* **1998**, *98*. Available online: https://www.cs.csustan.edu/~xliang/Courses/CS4710-21S/Papers/06%20RRT.pdf (accessed on 31 July 2021)

14. Liu, J.; Shi, G.; Zhu, K. Online Multiple Outputs Least-Squares Support Vector Regression Model of Ship Trajectory Prediction Based on Automatic Information System Data and Selection Mechanism. *IEEE Access* **2020**, *8*, 154727–154745. [CrossRef]
15. Chiou, J.; Yang, Y.; Chen, Y. Multivariate functional linear regression and prediction. *J. Multivar. Anal.* **2016**, *146*, 301–312. [CrossRef]
16. Sun, S.; Chen, J.; Sun, J. Traffic congestion prediction based on GPS trajectory data. *Int. J. Distrib. Sens. Netw.* **2019**, *15*, 1550147719847440. [CrossRef]
17. Qian, L.P.; Feng, A.; Yu, N.; Xu, W.; Wu, Y. Vehicular Networking-Enabled Vehicle State Prediction via Two-Level Quantized Adaptive Kalman Filtering. *IEEE Internet Things J.* **2020**, *7*, 7181–7193. [CrossRef]
18. Wang, H.; Yang, Z.; Shi, Y. Next location prediction based on an adaboost-markov model of mobile users. *Sensors* **2019**, *19*, 1475. [CrossRef] [PubMed]
19. Ding, J.; Liu, H.; Yang, L.T.; Yao, T.; Zuo, W. Multiuser Multivariate Multiorder Markov-Based Multimodal User Mobility Pattern Prediction. *IEEE Internet Things J.* **2019**, *7*, 4519–4531. [CrossRef]
20. Ding, C.; Wang, W.; Wang, X.; Baumann, M. A Neural Network Model for Driver's Lane-Changing Trajectory Prediction in Urban Traffic Flow. *Math. Probl. Eng.* **2013**, *2013*, 1–8. [CrossRef]
21. Liu, M.; Shi, J. A cellular automata traffic flow model combined with a BP neural network based microscopic lane changing decision model. *J. Intell. Transp. Syst.* **2019**, *23*, 309–318. [CrossRef]
22. Julong, D. Control problems of grey systems. *Syst. Control Lett.* **1982**, *1*, 288–294. [CrossRef]
23. Zeng, B.; Duan, H.; Bai, Y.; Meng, W. Forecasting the output of shale gas in China using an unbiased grey model and weakening buffer operator. *Energy* **2018**, *151*, 238–249. [CrossRef]
24. Ma, X.; Mei, X.; Wu, W.; Wu, X.; Zeng, B. A novel fractional time delayed grey model with Grey Wolf Optimizer and its applications in forecasting the natural gas and coal consumption in Chongqing China. *Energy* **2019**, *178*, 487–507. [CrossRef]
25. Wu, W.; Ma, X.; Zhang, Y.; Wang, Y.; Wu, X. Analysis of novel FAGM (1, 1, t $\alpha$) model to forecast health expenditure of China. *Grey Syst.* **2019**, *9*, 232–250. [CrossRef]
26. Yershova, A.; Jaillet, L.; Siméon, T.; LaValle, S.M. Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain. In Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Barcelona, Spain, 18–22 April 2005; pp. 3856–3861
27. Jaillet, L.; Yershova, A.; La Valle, S.M.; Siméon, T. Adaptive tuning of the sampling domain for dynamic-domain RRTs. In Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, Edmonton, AB, Canada, 2–6 August 2005; pp. 2851–2856
28. Gan, Y.; Zhang, B.; Ke, C.; Zhu, X.; He, W.; Ihara, T. Research on Robot Motion Planning Based on RRT Algorithm with Nonholonomic Constraints. *Neural Process. Lett.* **2021**, *53*, 3011–3029. [CrossRef]
29. Dalibard, S.; Laumond, J.P. Control of probabilistic diffusion in motion planning. In *Algorithmic Foundation of Robotics VIII*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 467–481.
30. Burns, B.; Brock, O. Single-query motion planning with utility-guided random trees. In Proceedings of the 2007 IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April 2007; pp. 3307–3312
31. Kuffner, J.J.; LaValle, S.M. RRT-connect: An efficient approach to single-query path planning. In Proceedings of the 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings, San Francisco, CA, USA, 24–28 April 2000; Volume 2, pp. 995–1001
32. Li, T.; Shie, Y. An incremental learning approach to motion planning with roadmap management. In Proceedings of the 2002 IEEE International Conference on Robotics and Automation, Washington, DC, USA, 11–15 May 2002; Volume 4, pp. 3411–3416.
33. Otte, M.; Correll, N. C-FOREST: Parallel shortest path planning with superlinear speedup. *IEEE Trans. Robot.* **2013**, *29*, 798–806. [CrossRef]
34. Strandberg, M. Augmenting RRT-planners with local trees. In Proceedings of the IEEE International Conference on Robotics and Automation, New Orleans, LA, USA, 26 April–1 May 2004; Volume 4, pp. 3258–3262
35. Tan, G. The structure method and application of background value in grey system GM(1, 1) model (I). *Syst. Eng. Theory Pract.* **2000**, *20*, 98–103.
36. Guanjun, T. The Structure Method and Application of Background Value in Grey System GM(1, 1) model (II). *Syst. Eng. Theory Pract.* **2000**, *5*, 125–132.
37. Sun, J.; Li, H.; Zeng, B.; Zhao, X.; Wang, C. Parameter Optimization on the Three-Parameter Whitenization Grey Model and Its Application in Simulation and Prediction of Gross Enrollment Rate of Higher Education in China. *Complexity* **2020**, *2020*, 1–10.
38. Yuhong, W.; Jie, L. Improvement and application of GM(1, 1) model based on multivariable dynamic optimization. *J. Syst. Eng. Electron.* **2020**, *31*, 593–601. [CrossRef]
39. Cheng, M.; Shi, G. Improved methods for parameter estimation of gray model GM(1, 1) based on new background value optimization and model application. *Commun. Stat. Simul. Comput.* **2019**, 1–23. [CrossRef]
40. Zhu, Y.; Jian, Z.; Du, Y.; Chen, W.; Fang, J. A New GM(1, 1) Model Based on Cubic Monotonicity-Preserving Interpolation Spline. *Symmetry* **2019**, *11*, 420. [CrossRef]
41. Li, K.; Liu, L.; Zhai, J.; Khoshgoftaar, T.M.; Li, T. The improved grey model based on particle swarm optimization algorithm for time series prediction. *Eng. Appl. Artif. Intell.* **2016**, *55*, 285–291. [CrossRef]

42. Chen, Y. High-order polynomial interpolation based on the interpolation center's neighborhood the amendment to the runge phenomenon. In Proceedings of the 2009 WRI World Congress on Software Engineering, Xiamen, China, 19–21 May 2009; Volume 2, pp. 345–348.
43. Wang, Z.; Dang, Y.; Liu, S.; Zhou, J. The optimization of background value in GM(1, 1) model. *J. Grey Syst.* **2007**, *10*, 69–74. [CrossRef]
44. Ju, Q.; Li, H.; Zhang, Y. Power management for kinetic energy harvesting IoT. *IEEE Sens. J.* **2018**, *18*, 4336–4345. [CrossRef]
45. Ju, Q.; Zhang, Y. Predictive power management for internet of battery-less things. *IEEE Trans. Power Electron.* **2017**, *33*, 299–312. [CrossRef]
46. Lin, Y.; Saripalli, S. Sampling-based path planning for UAV collision avoidance. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 3179–3192. [CrossRef]
47. Wang, Q.; Zhang, Z.; Wang, Z.; Wang, Y.; Zhou, W. The trajectory prediction of spacecraft by grey method. *Meas. Sci. Technol.* **2016**, *27*, 085011. [CrossRef]
48. Xia, X.; Chen, X.; Zhang, Y.; Wang, Z. Grey bootstrap method of evaluation of uncertainty in dynamic measurement. *Measurement* **2008**, *41*, 687–696. [CrossRef]
49. Luo, Z.; Wang, Y.; Zhou, W.; Wang, Z. The grey relational approach for evaluating measurement uncertainty with poor information. *Meas. Sci. Technol.* **2015**, *26*, 125002. [CrossRef]