Research Article

Trajectory Planning of UAVs with Fault Tolerance Based on Monte Carlo Sampling

Jianwei Wu,1,2 Lin Chen,2 Yang Zhou,2 and Fuyun Liu1

1School of Mechanical and Electrical Engineering, Guilin University of Electronic Technology, Guilin 541004, China
2School of Mechanical Engineering, Southeast University, Nanjing 211189, China

Correspondence should be addressed to Fuyun Liu; liufuyun310@aliyun.com

Received 11 December 2021; Revised 27 March 2022; Accepted 30 March 2022; Published 11 July 2022

1. Introduction

Trajectory planning of UAVs refers to generating an optimal flight trajectory between the starting and ending points, considering factors such as energy consumption, maneuverability, arrival time, terrain, and threat [1–3]. It is one of the key technologies for UAVs and helps to accomplish many missions such as reconnaissance [4, 5], surveillance [6, 7], road patrol [8], aerial photo [9], object detection [10, 11], and cargo transfer [12].

It is a very important research topic to realize trajectory planning of a UAV fast in complex environments. Due to the limitation of the UAV’s structure and the influence of the complex environments, its positioning system cannot be precisely located [13]. Once the positioning errors accumulate to a certain extent, the task of a UAV may fail. At present, there is a technology of error correction [14–16], which is used for eliminating UAV’s errors when it flies into the vicinity of some points (called correction points) arranged in advance. The technology can ensure that a UAV flies along the planned trajectory and performs its task successfully. However, affected by complex environments such as weather and terrain, these points could be invalid even after the positioning errors of a UAV are corrected by them. Accordingly, the real challenge is how to ensure that the optimal trajectory for UAVs has the ability of fault tolerance.

The past decades have witnessed many studies in trajectory planning for UAVs. To realize the trajectory planning for UAVs, these studies constructed single or multiple objective optimization with multiconstraints by considering the energy consumption, security (for example, the fewer turning operations), and the shortest path, under constraints such as threat, obstacle, terrain, and arrival time. Researchers have
studied its modeling, optimization, and solution in detail, and a large number of algorithms have been presented as follows: Voronoi diagram method [17], A* algorithm [18], improved A* algorithm [19], artificial potential field method [20], algorithm based on disturbed fluid and trajectory propagation [21], genetic algorithm [22, 23], particle swarm optimization (PSO) algorithm [24], neural network algorithm [25, 26], simulated annealing algorithm [27], and so on. These algorithms having their respective characteristics provide us with a basis for selecting them according to different tasks. For example, graph-based approaches are usually very accurate, but since some missions require several constraints on the resultant path, they face some difficulties, such as a long convergence time. Artificial potential field method has the advantages of simple principle and small amount of computation, yet there may be some oscillation of the trajectory if it enters into the narrow gap between two obstacles; heuristic algorithms, such as genetic algorithm and particle swarm optimization algorithm, are also used for multiobjective optimization but do not guarantee that an optimal solution will be found because there is no an analytical proof [28].

In spite of so many studies on trajectory planning for UAVs, they mainly focus on improving performances such as trajectory smoothness, trajectory length, obstacle crossing ability, and computation efficiency. In fact, few studies focus on the trajectory planning for UAVs considering the error correction, let alone on the trajectory planning with fault tolerance. Because there are many correction points arranged in advance and performing the error correction is conditional by them, the trajectory planning of the UAVs considering the error correction is more difficult than without considering it, especially under the three premises that UAVs are with minimal turning radius (i.e., considering the kinematical constraints), that the flying distance is short enough, and that the points used for error correction are as few as possible. Moreover, since correction points have a failure probability, the flight errors could not be entirely eliminated by them. Therefore, a greater difficulty is how to design a trajectory that can satisfy the above premises to arrive at the destination successfully, even if the error correction fails. Maybe “it is for these reasons” that the current studies can still not break through the stage that realizes the trajectory planning of UAVs with fault tolerance.

To plan an optimal trajectory considering error correction with fault tolerance, this paper proposes two trajectory planning methods based on the definite rule (i.e., minimizing optimization objectives designed) and Monte Carlo sampling. By constructing five calculation models of the two methods in detail, designing the algorithms appropriately, and developing computational programs based on MATLAB, trajectory planning of UAVs with fault tolerance is achieved. Simulation example illustrates the methods’ feasibility and results’ rationality.

The main contributions of this study are as follows. (i) This paper proposes two trajectory planning methods for UAVs with error correction, which overcomes the limitation in traditional methods that cannot deal with the long-distance trajectory planning for UAVs needed to consider error correction. (ii) The proposed method based on Monte Carlo sampling can perform the trajectory planning with error correction existing in failure, which further expands its application and has practical value. (iii) A set of solution programs in MATLAB 2018a programming environment are developed based on the trajectory planning methods, and the optimal trajectory is obtained by a practical problem.

The rest of this paper is organized as follows. Section 2 provides the necessary background. Section 3 describes in detail the trajectory planning methods with fault tolerance based on definite rule and Monte Carlo sampling and respectively establishes the following five calculation models: the calculation model of the trajectory length considering the turning radius, the optimization model of the maximum spherical surface, the optimal correction point model considering the weighted distance, the model of updating errors, and the model of extracting preferable correction points. The reasons for selecting optimization algorithms and steps of the implementation are given in Section 4. A simulation example from a practical problem is presented in Section 5, and the corresponding simulation results and their discussions are described, including the optimized trajectories under the different weights and the feasible trajectories, the optimal trajectory, and the comparison with different methods. Finally, the conclusions are drawn in Section 6.

The framework of this paper is shown in Figure 1.

2. Background

Figure 2 shows the flight space of UAVs with the correction points used for eliminating its positioning errors. However, affected by complex environments such as weather and terrain, some correction points could be invalid even after the positioning errors of a UAV are corrected by them. These points with the failure possibility are expressed in squares, as shown in Figure 2. In the figure, A and B are, respectively, the starting and ending points expressed in red, C^2 is the first
correction point searched, the vertical and horizontal correction points are shown in green and blue dots, respectively, and the black curve represents a planned trajectory. During planning a trajectory from A to B, the following constraints will be taken into account.

(1) A UAV needs real-time positioning during flight, and its positioning accuracy includes vertical and horizontal errors. When the UAV reaches the ending point, if both the vertical and horizontal errors are less than \( \theta \) m (called constraint \( \theta \) in this paper), it is regarded that the UAV successfully arrives at the destination. For every 1 m flying, it is assumed that both the vertical and horizontal errors of a UAV will be increased by \( \delta \) m, respectively, where \( \delta \) is a rough estimator based on the previous positioning performance.

(2) A UAV needs to correct the positional error during flight. The correction points include two types, horizontal and vertical corrections. When a UAV reaches a correction point, the vertical or horizontal error can be eliminated based on its type. It is assumed that the positions of correction points in the flight space can be obtained before planning flight trajectory and that these points determined by the terrain of flight space are not a uniform law. If both the vertical and horizontal errors of a UAV are corrected in time, it can fly along the planned trajectory and finally arrive at the intended destination.

(3) At the starting point A, the vertical and horizontal errors of a UAV are both zero.

(4) The vertical error of a UAV will be eliminated after it is successfully corrected by vertical correction points, but its horizontal error will remain unchanged.

(5) The horizontal error of a UAV will be eliminated after it is successfully corrected by horizontal correction points, but its vertical error will remain unchanged.

(6) Vertical correction of a UAV can be performed when its vertical and horizontal errors are no more than \( \alpha_1 \) and \( \alpha_2 \), respectively.

(7) Horizontal correction of a UAV can be performed when its vertical and horizontal errors are no more than \( \beta_1 \) and \( \beta_2 \), respectively.

(8) A UAV has a minimum turning radius \( r \). Limited by the structure and control system, it cannot finish an immediate turning (i.e., the direction of a UAV cannot be changed abruptly) and hence has a minimum turning radius.

(9) The probability of a UAV being successfully corrected is \( \sigma \), when it passes a correction point with the failure probability.

(10) Flight error of a UAV, not more than \( \zeta \), will be leftover rather than completely eliminated, when the error correction fails by correction points with the failure probability. Namely, \( e_f = \min \{ e_r, \zeta \} \), where \( e_f \) and \( e_r \) are the errors before and after being corrected, respectively.

From the above descriptions, it can be known that the aim of this study is to find the optimal trajectory for a UAV considering all the above constraints. Even if error corrections fail, the trajectory planning methods with failure tolerance still can ensure that the UAV successfully arrives at the destination. Surely, the optimal trajectory needs to achieve the objectives that trajectory length is as short as possible and that the number of corrections the UAV passes is as small as possible.

3. Trajectory Planning Methods with Failure Tolerance

During the flight of a UAV from the starting point to the ending point, for eliminating its positioning errors, it needs to pass some correction points one by one and reaches the destination at last. However, there exists the failure probability after being corrected by certain points. To ensure UAVs have the ability of fault tolerance, there is a
conservative handling method assuming that the error corrections all fail by the correction points with the failure probability. Under this assumption, without doubt, if a UAV can fly along a series of trajectories, if one of them subject to all the constraints in Section 2 is the shortest, it must be the optimal trajectory along which the UAV can successfully arrive at the destination.

Based on the handling method, the key is to search the best correction points by means of a definite rule, in order to obtain the optimal trajectory considering the error correction. To achieve the purpose, “the definite rule” mentioned in this paper is to construct the optimization models and to minimize their objective function under constraints. As a result, the trajectory planning of UAVs with fault tolerance becomes an optimization problem, and its objectives include that the trajectory length is the shortest and that the correction points UAV passes are as small as possible.

Therefore, the trajectory planning model with the error correction will be established by using the handling method and by considering the optimization objectives and used for obtaining a series of the best correction points. To achieve trajectory planning considering the error correction with fault tolerance, we establish the calculation model of the trajectory length considering the minimal turning radius, the optimization model of the maximum spherical surface, the optimal correction point model considering the weighted distance, the model of updating errors, and the model of extracting preferable correction points. The trajectory planning methods with failure tolerance for UAVs can be formed by integrating these models, and the optimal trajectory with failure tolerance will be obtained by solving them.

3.1. Modeling Based on the Definite Rule

3.1.1. Calculation Model of the Trajectory Length considering Turning Radius. The calculation model considering turning radius is shown in Figure 3, where $C^{i}$, $C^{i+1}$, and $C^{i+1}$ denote the previous starting point, the current starting point (i.e., the best correction point searched at the previous iteration), and a new correction point searched, respectively. It can be known that the following flight trajectory between starting point to the correction point can ensure the flight distance is small, when the UAV flies into point $C^{i+1}$, its flight direction is just tangent to straight line $C^{i}C^{i+1}$; namely, during the flying from the point $C^{i}$ to $C^{i+1}$, the UAV flies first with the minimum radius $r$ and then with a large maximum radius $R$. It should be noted that the new correction point is just the ending point $B$, when the trajectory errors $e_{x}$ and $e_{y}$ from the current starting point to ending point $B$ are both less than $\theta$.

In Figure 3, points $O^{i}$ and $O^{i+1}$ are the centers of the circle corresponding to radii $r$ and $R$, respectively. Straight line $O^{i}Q$ is perpendicular to straight line $C^{i}C^{i+1}$. $N$ is the point of intersection between straight lines $O^{i}O^{i+1}$ and $C^{i}C^{i+1}$, and $P$ is the tangent point of the two circles. $\alpha$ is the angle between straight lines $C^{i}O^{i}$ and $C^{i}C^{i+1}$, and $\beta$ is the angle between straight lines $O^{i+1}O^{i}$ and $O^{i}Q$.

By the geometrical relationship shown in Figure 3, the following expressions can be obtained:

$$
y = \arccos \left( \frac{\vec{C}^{i}C^{i+1} \cdot \vec{C}^{i+1}C^{i+2}}{|\vec{C}^{i}C^{i+1}| |\vec{C}^{i+1}C^{i+2}|} \right) - \frac{\pi}{2},
$$

$$
|\vec{C}^{i}C^{i+1}| = |\vec{C}^{i}Q| + |\vec{Q}C^{i+1}| = r \cos(y) + (R + r) \sin(\psi),
$$

$$
R = |\vec{O}^{i+1}N| \cos(\psi) = \left( (R + r) - |\vec{O}^{i}N| \right) \cos(\beta)
= (R + r) \cos(\psi) - r \sin(\psi) \Rightarrow,
$$

$$
|\vec{O}^{i+1}C^{i+1}| = r \frac{\cos(\psi) - r \sin(\psi)}{1 - \cos(\psi)}.
$$

By substituting equations (3) into (2), the calculation formula of the $|\vec{C}^{i}C^{i+1}|$ can be given as

$$
|\vec{C}^{i}C^{i+1}| = r \left( \cos(y) + \frac{1 - \sin(y)}{1 - \cos(\psi)} \sin(\psi) \right).
$$

According to equation (3), $\psi = \arccos(R + r \sin(y)/R + r)$, which manifests that angle $\psi$ belongs to the first quadrant. Since equation (3) is an algebraic equation, it is not hard to get angle $\psi$ by solving it.

Next, by using the calculated $\psi$ and relationship between the arc length and central angle, the trajectory length of a UAV from the point $C^{i}$ to $C^{i+1}$ can be written as

$$
|\vec{C}^{i}C^{i+1}| = r \left( \frac{\pi}{2} - y + \psi \right) + r \left( \frac{\cos(\psi) - \sin(\psi)}{1 - \cos(\psi)} \right),
$$

where $|\vec{C}^{i}C^{i+1}|$ denotes the sum of the arc lengths $\vec{C}^{i}P$ and $\vec{PC}^{i+1}$.

3.1.2. Optimization Model of the Maximum Spherical Surface. To find a best correction point at each iteration, a set of correction points satisfying constraints can be determined first. The optimization model of the maximum spherical surface is established by considering constraints (6) and (7). Because the optimization model is preliminarily to obtain a set of correction points, accordingly, it will neglect the influence of the turning radius, but the optimal
correction point model (described in Section 3.1.3) considering the weighted distance will not.

The optimization model of the maximum spherical surface, including two types of constraint functions caused by the vertical and horizontal correction points, is given as follows:

1. Define design variable
   \[ X = (x_1, x_2, x_3)^T. \]

2. Determine objection function
   \[ \min f(X) = |\overrightarrow{CX}|, \]
   where \( C_i \) is the starting point at iteration \( i \) (i.e., the best correction point searched at iteration \( i-1 \)), and \(|\overrightarrow{CX}|\) denotes the length of straight line from the point \( C_i \) to \( X \) whose coordinates in the three-dimensional space are \( x_1, x_2, \) and \( x_3 \).

3. Define two constraint functions according to the types of correction points.

The constraints (constraint (6) mentioned in Section 2) of the maximum errors allowed to be vertically corrected are given by

\[ \begin{aligned}
   g_1(X) &= e_i^y + |\overrightarrow{C_iX}| \delta - \alpha_1 \leq 0, \\
   g_2(X) &= e_i^x + |\overrightarrow{C_iX}| \delta - \alpha_2 \leq 0,
\end{aligned} \]
where \( e_i^y \) and \( e_i^x \) denote the vertical and horizontal errors (i.e., the errors after being corrected at iteration \( i-1 \)) of the starting point at the iteration \( i \), respectively.

The constraints (constraint (7) mentioned in Section 2) of the maximum errors allowed to be horizontally corrected are given by

\[ \begin{aligned}
   g_3(X) &= e_i^y + |\overrightarrow{C_iX}| \delta - \beta_1 \leq 0, \\
   g_4(X) &= e_i^x + |\overrightarrow{C_iX}| \delta - \beta_2 \leq 0.
\end{aligned} \]

The maximum radii are \( R_{i1} \) and \( R_{i2} \) by solving the optimization model above with the two constraint functions. Then, we will select a set of correction points belonging to the maximum radius from \( R_{i1} \) and \( R_{i2} \) so that each flight distance of a UAV is farther before its flight errors are corrected as shown in Figures 4 and 5. Obviously, the selection method can ensure that the correction points UAV passes will be as few as possible. In the two figures, the green and blue points represent vertical and horizontal correction points, respectively. After determining the type of correction points and judging whether the distance from the point \( C_i \) to a correction point is less than the chosen maximum radius, a set of correction points can be obtained. In the next section, the set of correction points will be used for searching the best correction point based on the optimal correction point model considering the weighted distance.

3.1.3. Optimal Correction Point Model considering the Weighted Distance. The weighted distance model is shown in Figure 6 where point \( C_{i+1} \) is the best point being searched, and point \( D \) is the foot of the perpendicular from the point \( C_{i+1} \) to straight line \( CB \). In Figure 6, point \( X_i^k \) (\( k = 1, 2, \ldots, n \)) denotes one of the correction points, and \( n \) denotes the total number of the correction points selected by the optimization model of the maximum spherical surface at iteration \( i \).

To realize the optimization objectives that trajectory length is as short as possible and that the correction points a UAV passes are as few as possible, on the one hand, the flight direction should be toward the ending point so that the trajectory length shortens, that is, the distance \( |X_i^kD| \) is as small as possible. That the distance \( |X_i^kB| \) shortens, on the other hand, will result in the decrease of total correction points a UAV passes. Therefore, by taking into account the weight of distances \( |X_i^kD| \) and \( |X_i^kB| \), the optimal correction point model can be established as follows:

1. Define design variable \( x \)
   \[ x = 1, 2, \ldots, n. \]

2. Design objective function
Mathematical Problems in Engineering

3.1.4. Model of Updating Errors. After obtaining the best correction point \( C^{i+1} \), the current starting point will be replaced by it as a new starting point, and a UAV’s flight errors will be updated for the next iteration. The handling method assumes that error corrections all fail by those correction points with failure probability. Accordingly, updating errors also needs to consider the failure label of the point \( C^{i+1} \) besides its type, and the formulas for updating errors can be expressed as follows.

Case 1: \( C^{i+1} \) is a vertical correction point

\[
\begin{align*}
\epsilon_x^{i+1} &= \tau \cdot \min \{ \epsilon_x^i + \hat{C} \cdot C^{i+1} \cdot \delta, \zeta \}, \\
\epsilon_z^{i+1} &= \epsilon_z^i + \hat{C} \cdot C^{i+1} \cdot \delta,
\end{align*}
\]

where \( \epsilon_x^{i+1} \) and \( \epsilon_z^{i+1} \) denote the vertical and horizontal errors of the starting point at iteration \( i + 1 \), respectively, and \( \tau \)—either “0” or “1”—represents failure label of point \( \epsilon_x^{i+1} \).

Case 2: \( C^{i+1} \) is a horizontal correction point

\[
\begin{align*}
\epsilon_x^{i+1} &= \epsilon_x^i + \hat{C} \cdot C^{i+1} \cdot \delta, \\
\epsilon_z^{i+1} &= \tau \cdot \min \{ \epsilon_z^i + \hat{C} \cdot C^{i+1} \cdot \delta, \zeta \}.
\end{align*}
\]

3.2. Modeling Based on Monte Carlo Sampling

3.2.1. Motivation. Generally, using the definite rule by minimizing the designed objective functions can obtain the optimal trajectory whose length and searched points of the planned trajectory are as small as possible. However, affected by the distribution and number of correction points, any feasible trajectory cannot be found based on the definite rule, and the simulation example in Section 5 will give the illustration. Actually, any correction point searched at a certain iteration has a great influence on the next iteration, such as the number, type, failure label of correction points allowed, and trajectory feasibility (i.e., whether a UAV flying along the trajectory can arrive at the destination success-
3.2.2. Model of Extracting Preferable Correction Points.

To avoid that no correction point can be found based on the definite rule, we use not only the best correction point but also others at each iteration. In this way, the role of weight $\omega$ mentioned in equation (11) will be weakened. Accordingly, the distance $|X_i^j|B|$ can be directly taken as the objective function for extracting preferable correction points by sorting the distances $|X_i^j|B|$ ($i = 1, 2, 3, \ldots, m$). $m$ denotes the number of correction points satisfying the constraints from equations (12) and (13).

The model of extracting preferable correction points is shown in Figure 7, and it extracts $k$ correction points from $m$ by distances $|X_i^j|B|$ ($i = 1, 2, 3, \ldots, m$) from small to large. It can be regarded as an improvement for the optimal correction point model mentioned in Section 3.1.3, and one of the $k$ points will be randomly selected based on Monte Carlo sampling.

It should be noted: if $m < k$, in the next step, one point will be randomly selected from such correction points less than $k$; if $m = 0$ and the trajectory errors (from the updated starting point to the ending point) do not satisfy constraint $\theta$, a UAV cannot successfully arrive at the destination when along the trajectory, so it is necessary to automatically return the initial states to restart.

4. Algorithms and Implementation

4.1. Algorithms. To solve the optimization models, sequential quadratic programming (SQP) and the exhaustive search will be adopted, mainly based on the following reasons.

1. The optimization model of the maximum spherical surface, whose objective and constraint functions are quadratic, belongs to a quadratic programming problem. On the one hand, the trajectory planning considering error correction involves many optimizations at each iteration. That is to say, the computational cost should be given priority; otherwise, the real-time performance cannot be ensured in practice. On the other hand, our main work is to emphasize the modeling of trajectory planning considering error correction, so solving it can directly call the SQP algorithm provided by MATLAB. As described in the literature [38–41], SQP, being a high-efficiency method, is considered to be one of the most efficient methods to solve the nonlinearly constrained optimization problems, particularly suitable for the quadratic programming problem. Accordingly, "the SQP method will be adopted to solve the optimization model of the maximum spherical surface."

2. The optimal correction point model considering the weighted distance will employ the exhaustive search algorithm by programming based on the MATLAB because its feasible points are very finite after a set of correction points obtained by the optimization model of the maximum spherical surface. By using the barbaric algorithm, the best correction point can be found under an assigned weight.

4.2. Implementation. The flowchart of trajectory planning methods with fault tolerance is shown in Figure 8, and the implementation steps are given as follows.

Step 1. Obtain the starting and ending points, as well as all of the correction points in the input data. That means acquiring their information, such as their coordinates in the three-dimensional space, types, and failure labels.

Step 2. Judge whether the flight errors (which can be obtained by the calculation model of the trajectory length considering the turning radius) from the starting point to the ending point satisfy the condition $\theta$ and determine whether to continue the next iteration.
Step 3. Determine the type of correction points from the two radii by selecting the maximum one and obtain a set of the correction points within the spherical surface of the selected radius, according to the optimization model of the maximum spherical surface with its two constraint functions.

Step 4. Perform the following steps separately. Cases 1 and 2 represent the trajectory planning methods based on the definite rule and Monte Carlo sampling, respectively.

Case 1: Obtain the best correction point from the set of the correction points, according to the optimal correction point model considering the weighted distance.

Case 2: Extract $k$ correction points considering the distances $|BX_i|$ at each iteration.

Step 5.

Case 1: Take the best correction point as a new starting point, update the flight errors after the UAV is corrected, and continue the next iteration.

Case 2: Monte Carlo sampling. Randomly select one from the $k$ correction points (preferable correction points) as a new starting point, update the flight error...
after the UAV is corrected, and continue the next iteration.

Step 6. Repeat Steps 1–5, until Step 2 not to search next correction point. That is, the vertical and horizontal errors caused by the flight distance between the new starting point and ending point B are both less than \( \theta \) m.

5. Simulation Example

In this section, we apply the trajectory planning methods proposed in Section 3 to solve a practical problem. This problem is very typical because the number of correction points is small and the requirements are very strict, which may lead to the failure of the definite rule method.

In MATLAB 2018a programming environment, a set of solution programs is developed based on the trajectory planning methods, and the optimal trajectory is obtained. It is noted that the developed program is not limited to this problem to be solved, if these data of a flight space change, and the new results of trajectory planning for UAVs can be obtained after importing the new data.

5.1. Problem to Be Solved. Table 1 shows the partial data of a flight space, which includes the coordinates of the starting point, ending point, and correction points. The serial numbers 0 and 326 are the starting point and ending point, respectively, and the rest are correction points. In the column containing the type of correction points, "0" and "1" represent the horizontal and vertical error correction points, respectively. In the last column, "0" and "1" represent the correction points with and without the failure probability, respectively; the probability of a successful correction is only \( \sigma \) when the flight error is corrected by those having the failure probability.

For this problem, the parameters mentioned in Section 2 are given as follows: \( \alpha_1 = 20 \, \text{m} \), \( \alpha_2 = 10 \, \text{m} \), \( \beta_1 = 15 \, \text{m} \), \( \beta_2 = 20 \, \text{m} \), \( \theta = 20 \, \text{m} \), \( \delta = 0.001 \, \text{m} \), \( \sigma = 80\% \), \( \xi = 5 \, \text{m} \), and \( r = 200 \, \text{m} \).

Table 1: Partial data of a flight space [42].

<table>
<thead>
<tr>
<th>Serial numbers</th>
<th>Coordinates</th>
<th>Type of correction points</th>
<th>Failure label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X (m)</td>
<td>Y (m)</td>
<td></td>
</tr>
<tr>
<td>Starting point A</td>
<td>0</td>
<td>0.00</td>
<td>50000.00</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>76009.85</td>
<td>9788.11</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2448.20</td>
<td>71599.88</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>27800.93</td>
<td>15218.07</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>50056.73</td>
<td>91668.09</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>25311.15</td>
<td>51225.79</td>
</tr>
<tr>
<td>Correction points</td>
<td>324</td>
<td>78846.70</td>
<td>3906.98</td>
</tr>
<tr>
<td></td>
<td>325</td>
<td>63645.56</td>
<td>6263.13</td>
</tr>
<tr>
<td></td>
<td>326</td>
<td>100000.00</td>
<td>5499.61</td>
</tr>
<tr>
<td>Ending point B</td>
<td>324</td>
<td>78846.70</td>
<td>3906.98</td>
</tr>
<tr>
<td></td>
<td>325</td>
<td>63645.56</td>
<td>6263.13</td>
</tr>
</tbody>
</table>

5.2. Results and Discussion Based on the Definite Rule. After considering that certain correction points have the failure probability and assuming that all corrections fail by these points at each iteration, the results can be obtained by running the programs developed in this paper. However, the simulation results show that, no matter which weight we select, the serial numbers of the correction points searched are all [163, termination]. This is a special phenomenon, which presents that after serial number 163 is searched, solving the optimal correction point model cannot find any correction point at the next iteration. This causes the cease of the programs unexpectedly. In other words, if there exists the failure probability on certain correction points, no feasible trajectory can be found based on the definite method for the practical problem whose distributions on correction points are relatively rigorous. "It is for this reason that we further propose a trajectory planning method based on Monte Carlo Sampling," and the simulation results are as shown in Section 5.3.

To illustrate the simulation process and results based on the definite rule, the restriction of the correction points for the problem will be loosened, and it is assumed that all correction points will not fail because most problems fall into this situation. In other words, in equations (14) and (15), \( r \) will remain 0, and the corresponding results are as follows.

5.2.1. Optimized Trajectories under the Different Weights. Because correction points are finite and not uniformly distributed in the whole space, weight \( \omega \) from 0 to infinite can be discretionally picked to optimize trajectories as shown in Table 2.

If weight \( \omega \) is very small, the best correction points searched mainly depend on the distance \([BX_i]\). To minimize the objective function (i.e., equation (11)), a correction point farthest from the starting point will be found. In other words, a UAV will fly a longer distance at each iteration, which may cause that the optimal correction point model cannot find any correction point because flight errors are so large that they cannot satisfy the constraints from equations (12) and (13) at next iteration. When weight \( \omega \) belongs to [0, 0.2], the series number of the correction points searched just illustrates the point as shown in Table 2. It should be noted that the "termination" shown in Table 2 presents that after serial number 320 is searched, solving the optimal correction point model cannot find any correction point at the next iteration, which causes the cease of the programs unexpectedly.
With the increase of weight $\omega$, the importance of $|B_i\times x| \to 0$ shown in Figure 6 is gradually weakened. Under a larger weight $\omega$, it is easy to understand that if there are three or more correction points (satisfying constraints (6) and (7)) nearby the straight line between the starting and ending points, the optimal correction point model may always search the same points so that running the programs will fall into an endless loop. When $\omega > 49$, the series number of the correction points searched just illustrates the point as shown in Table 2.

It can be seen from Table 2 that, the optimal trajectory is obtained when weight $\omega$ belongs to $[6, 25]$; in this case, trajectory length is the shortest, and the correction points UAV passes are very few. If there is no failure of all the correction points, therefore, the total length of the optimal trajectory is 111.7787 km, and the total number of the best correction points searched is 12.

5.2.2. Optimal Trajectory. Table 3 shows the flight data of the UAV along the optimal trajectory, which includes the correction points, vertical and horizontal errors before being corrected, type of correction points, and flight distance from the starting point updated continually to the best correction point searched at each iteration, respectively.

Table 2: Optimized trajectory under the different weights.

<table>
<thead>
<tr>
<th>Weight $\omega$</th>
<th>Total number of the best correction points</th>
<th>Trajectory length (km)</th>
<th>Serial numbers of the points searched</th>
</tr>
</thead>
<tbody>
<tr>
<td>0~0.2</td>
<td>—</td>
<td>—</td>
<td>[163, 114, 8, 309, 255, 148, 90, 156, 320, termination]</td>
</tr>
<tr>
<td>0.3~0.5</td>
<td>14</td>
<td>137.7666</td>
<td>[163, 114, 8, 309, 255, 148, 90, 156, 13, 164, 50, 187, 61, 292, 326]</td>
</tr>
<tr>
<td>0.6~0.7</td>
<td>14</td>
<td>136.0649</td>
<td>[163, 114, 8, 309, 255, 148, 90, 156, 13, 164, 50, 323, 61, 292, 326]</td>
</tr>
<tr>
<td>0.8~3</td>
<td>14</td>
<td>135.0177</td>
<td>[163, 114, 8, 309, 255, 148, 90, 156, 33, 289, 50, 323, 61, 292, 326]</td>
</tr>
<tr>
<td>4~5</td>
<td>14</td>
<td>127.5342</td>
<td>[163, 114, 8, 309, 121, 123, 49, 160, 138, 19, 50, 322, 326, 61, 292, 326]</td>
</tr>
<tr>
<td>6~25</td>
<td>12</td>
<td>111.7787</td>
<td>[163, 114, 8, 309, 121, 123, 49, 160, 92, 93, 61, 292, 326]</td>
</tr>
<tr>
<td>27~29</td>
<td>15</td>
<td>123.6882</td>
<td>[163, 114, 234, 188, 222, 230, 225, 170, 123, 93, 61, 292, 326]</td>
</tr>
<tr>
<td>30~30.5</td>
<td>15</td>
<td>122.4127</td>
<td>[163, 114, 234, 188, 228, 227, 309, 121, 123, 49, 160, 92, 93, 61, 292, 326]</td>
</tr>
<tr>
<td>31~48</td>
<td>16</td>
<td>120.0604</td>
<td>[163, 114, 234, 188, 238, 227, 309, 305, 123, 45, 160, 92, 93, 61, 292, 326]</td>
</tr>
<tr>
<td>&gt;49</td>
<td>No solution</td>
<td></td>
<td>[163, 114, 234, 188, 114, 234, 188, 114, 234, 188, 114, 234, 188, 114, 234, 188, 114, 234, 188, 114, 234, 188, ...]</td>
</tr>
</tbody>
</table>

Table 3: Flight data of the UAV along the optimal trajectory.

<table>
<thead>
<tr>
<th>Serial numbers</th>
<th>Vertical error before being corrected</th>
<th>Horizontal error before being corrected</th>
<th>Type of correction points</th>
<th>Flight distance of each segment (km)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Starting point A</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>13.2879</td>
</tr>
<tr>
<td>163</td>
<td>13.2878</td>
<td>13.2879</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>114</td>
<td>18.6295</td>
<td>5.3416</td>
<td>1</td>
<td>5.346</td>
</tr>
<tr>
<td>8</td>
<td>13.9573</td>
<td>19.2989</td>
<td>0</td>
<td>13.9573</td>
</tr>
<tr>
<td>309</td>
<td>19.5215</td>
<td>5.5642</td>
<td>1</td>
<td>5.5642</td>
</tr>
<tr>
<td>121</td>
<td>11.2523</td>
<td>16.8166</td>
<td>0</td>
<td>11.2523</td>
</tr>
<tr>
<td>49</td>
<td>16.6633</td>
<td>5.4110</td>
<td>1</td>
<td>5.4110</td>
</tr>
<tr>
<td>123</td>
<td>11.798</td>
<td>17.2090</td>
<td>0</td>
<td>11.7980</td>
</tr>
<tr>
<td>160</td>
<td>18.3219</td>
<td>6.5239</td>
<td>1</td>
<td>6.5239</td>
</tr>
<tr>
<td>92</td>
<td>5.7878</td>
<td>12.3117</td>
<td>0</td>
<td>5.7878</td>
</tr>
<tr>
<td>93</td>
<td>15.273</td>
<td>9.4852</td>
<td>1</td>
<td>9.4852</td>
</tr>
<tr>
<td>61</td>
<td>9.8347</td>
<td>19.31988</td>
<td>0</td>
<td>9.8347</td>
</tr>
<tr>
<td>292</td>
<td>16.389</td>
<td>6.5543</td>
<td>1</td>
<td>6.5543</td>
</tr>
<tr>
<td>326</td>
<td>6.9805</td>
<td>13.5348</td>
<td>—</td>
<td>6.9805</td>
</tr>
</tbody>
</table>

Total trajectory length

= 111.7787 km

With the increase of weight $\omega$, the importance of $|BX_i\times x|$ shown in Figure 6 is gradually weakened. Under a larger weight $\omega$, it is easy to understand that if there are three or more correction points (satisfying constraints (6) and (7)) nearby the straight line between the starting and ending points, the optimal correction point model may always search the same points so that running the programs will fall into an endless loop. When $\omega > 49$, the series number of the correction points searched just illustrates the point as shown in Table 2.

It can be seen from Table 2 that, the optimal trajectory is obtained when weight $\omega$ belongs to $[6, 25]$; in this case, trajectory length is the shortest, and the correction points UAV passes are very few. If there is no failure of all the correction points, therefore, the total length of the optimal trajectory is 111.7787 km, and the total number of the best correction points searched is 12.

5.2.2. Optimal Trajectory. Table 3 shows the flight data of the UAV along the optimal trajectory, which includes the correction points, vertical and horizontal errors before being corrected, type of correction points, and flight distance from the starting point updated continually to the best correction point searched at each iteration, respectively.

Figure 9 shows the optimal trajectory in the space. The views in the two coordinate planes projected by the optimal trajectory are shown in Figures 9 and 10. The two figures clearly show the locations of the best correction points searched at each iteration. It is noted that in reality, the planned trajectory is a curve; however, Figures 9–11, the purpose of which is to describe the locations of the best correction points and to illustrate the process of searching correction points at each iteration, hence use a series of straight lines to express the optimal trajectory. Actually, owing to the small turning radius, it is not apparent to express the optimal trajectory with a curve.

It can be seen from Table 3 that the two types (“0 and 1”) of the best correction points searched appear alternately, and the flight distance of each segment appears with the long and short one. That is coincident with the principle that the best correction points are as few as possible. In other words, if a best correction point searched is the vertical one, the next must be the horizontal one in order to maximize the flight distance before being corrected.

Table 3: Flight data of the UAV along the optimal trajectory.
Figure 9 clearly shows that the flight trajectory is near the straight line AB when its three axes have the same range [0, 100 km]. Compared to the length of straight line AB being 103.0451 km, the optimal trajectory whose total length is only 111.7787 km illustrates that it is quite short. As shown in Table 3, all the flight errors are limited to the allowed range of the vertical and horizontal correction before being corrected. So, few are the best correction points searched, additionally, that it is sure that the optimal trajectory of the UAV is reasonable.

Furthermore, some correction points, which satisfy all constraints, can be randomly selected to generate many

**Figure 9:** Optimal trajectory without the failure probability in the space.

**Figure 10:** Optimal trajectory without the failure probability in the XOY plane.

**Figure 11:** Optimal trajectory without the failure probability in the XOZ plane.
Therefore, the optimal trajectory length is 184.0919 km and exists the failure probability on certain correction points, points the UAV passes are very few. Considering that there case, trajectory length is the shortest, and the correction Apparently, “trajectory 1” is the optimal trajectory; in this points searched are, the larger the trajectory length becomes. Feasible trajectories; it can be verified that, regardless of the trajectory length or number of correction points, they both are greater than the optimal trajectory’s counterparts.

From what has been analyzed and discussed in the three paragraphs above, it is worth believing that the results obtained by the trajectory planning method based on the definite rule are feasible and reasonable, and this work provides an effective solution for the trajectory planning of UAVs considering the error correction when there is no failure on the correction points.

5.3. Results and Discussions Based on Monte Carlo Sampling. In this section, the trajectory planning method based on Monte Carlo sampling will be adopted, considering that there is a failure on certain correction points, and the error corrections all fail by these points. By repeatedly running programs whose flowchart is shown in Figure 8, a series of feasible trajectories can be obtained based on Monte Carlo sampling. The optimal trajectory can be selected from them by considering the trajectory length and number of correction points a UAV passes.

5.3.1. Feasible Trajectories. Some feasible trajectories obtained by Monte Carlo sampling are shown in Table 4; in fact, many others are not listed due to the large trajectory lengths.

It can be seen from Table 4 that the more the correction points searched are, the larger the trajectory length becomes. Apparently, “trajectory 1” is the optimal trajectory; in this case, trajectory length is the shortest, and the correction points the UAV passes are very few. Considering that there exists the failure probability on certain correction points, therefore, the optimal trajectory length is 184.0919 km and its total number of correction points searched is 22. When a UAV flies along the optimal trajectory, even if error corrections fail, it can still arrive at the destination successfully. “It means that the trajectory planning method based on Monte Carlo sampling has the ability of fault tolerance.”

5.3.2. Optimal Trajectory. Table 5 shows the flight data of a UAV along the optimal trajectory with the failure probability on certain correction points. The flight data include the correction points, vertical and horizontal errors before and after being corrected, type of correction points, failure label, and flight distance from the starting point updated continually to the point searched at each iteration.

Figure 12 shows the optimal trajectory in the space with the failure probability on certain correction points. The views in the two coordinate planes projected by the optimal trajectory are shown in Figures 12 and 13. The two figures clearly show the locations of correction points searched at each iteration. Like Figures 9–11, Figures 12–14 also use a series of straight lines to express the optimal trajectory.

It can be seen from Table 5 that the two types (“0 and 1”) of correction points searched alternately appear like Table 3, which is in conformity with the principle that correction points are as few as possible. From the vertical and horizontal errors before and after being corrected, it is easy to verify that the optimal trajectory satisfies all the constraints. That is, before flight errors are corrected by vertical or horizontal correction points, they satisfy the constraint ((12) or (13)), respectively. Among correction points searched, only one point, whose serial number is 161, has the failure probability. That is reasonable because those points with the failure probability, not completely eliminating flight errors, should be maximally avoided.

![Table 4: Feasible trajectories obtained by Monte Carlo sampling.](image-url)

<table>
<thead>
<tr>
<th>Number of correction points searched</th>
<th>Trajectory length (km)</th>
<th>Serial numbers of points searched (including the ending point 326)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>22</td>
<td>[252, 266, 270, 89, 236, 133, 3, 55, 144, 69, 298, 117, 85, 76, 6, 12, 219, 16, 71, 141, 291, 161, 326]</td>
</tr>
<tr>
<td>2</td>
<td>23</td>
<td>[252, 266, 270, 89, 236, 133, 3, 281, 197, 112, 143, 250, 9, 253, 76, 6, 12, 219, 16, 71, 141, 291, 161, 326]</td>
</tr>
<tr>
<td>4</td>
<td>26</td>
<td>[169, 266, 270, 89, 236, 147, 278, 144, 55, 197, 112, 143, 250, 86, 73, 207, 70, 211, 321, 16, 282, 141, 291, 84, 124, 326]</td>
</tr>
<tr>
<td>6</td>
<td>28</td>
<td>[169, 266, 270, 89, 236, 133, 3, 281, 144, 55, 197, 112, 143, 250, 9, 253, 249, 274, 12, 219, 16, 101, 141, 291, 84, 124, 326]</td>
</tr>
<tr>
<td>7</td>
<td>29</td>
<td>[169, 266, 270, 89, 236, 133, 3, 281, 144, 55, 69, 298, 117, 85, 76, 6, 211, 44, 191, 160, 324, 19, 50, 177, 87, 246, 313, 210, 212, 326]</td>
</tr>
</tbody>
</table>
It can be shown from Figure 12 that the optimal trajectory is not better near straight line AB when its three axes keep the same range [0, 100 km]. Compared to the length of straight line AB being 103.0451 km, the optimal trajectory (with the failure probability on certain correction points) whose length is 184.0919 km has a large increase in length, further revealing that for the practical problem, the correction points with the failure probability is so severe so that the shorter trajectories cannot avoid them.

It is true that both the length and number of correction points belonging to the optimal trajectory are relatively large. In spite of this, the optimal trajectory satisfies all the constraints, and its length and number of correction points are smaller than other trajectories obtained by Monte Carlo sampling. Therefore, it is worth believing that the trajectory planning method based on Monte Carlo sampling is feasible and reasonable and provides a solution for trajectory planning with fault tolerance for UAVs.

5.4. Comparison from Different Methods. The computation time is a crucial factor that decides whether the method
proposed in this paper can be used in practice. Besides, the computation time based on Monte Carlo sampling may sometimes be unacceptable, so it is necessary to evaluate the time cost to obtain an optimized trajectory. To study the computation time, we give the computer configurations for simulation as follows.

Operating system: Windows 10 with 64bit
CPU: AMD Ryzen 7 4800H with Radeon Graphics
RAM: 16GB with 3200MHz
MATLAB version: MATLAB R2018a

From the results with the failure probability on certain correction points, it can be seen that using the definite rule cannot obtain a feasible trajectory. Hence, the computation time will be analyzed only for the results without the failure probability.

Table 6 shows under the different weights the computation time, which is the average time by repeatedly performing 20 times. By observing the correction points searched under different weights shown in Table 2, it can be seen from Table 6 that the time computation mainly depends on their number. That is, the more correction points searched are, the longer the computation time is. Table 6 also reveals that the computation time during each iteration is about 0.031s (i.e., 0.404/13 where “13” is the number of points searched). However, regardless of which weight is selected, computation time is short enough, which indicates that the method based on the definite rule has high efficiency and will promise to realize trajectory planning in real time under the hardware.

The research in this paper comes from the mathematical modeling competition of Chinese graduate students (including doctoral students) [43]. There are two excellent documents selected in this competition [44]. Their serial
numbers are F19102510081 and F19103360018, respectively. As described in the two documents, the adaptive and improved Dijkstra algorithm and ant colony algorithm are used in the first document, and the greedy algorithm and tabu search algorithm are used in the second document.

It can be seen after referring to the two documents that the numbers (i.e., 12) of correction points in the optimal trajectory are the same. The computation time is 42.23 s in the first document, 78.87 s in the second document, and however only 0.404 s in this paper. Their computation time is 104 and 195 times that of this paper, respectively. It is noted that the total lengths of the optimal trajectory are very close, and they are 109.46, 109.32, and 111.779 km, respectively, but our computational efficiency is much higher than the two documents.

Although our computers may differ, the configuration is generally little different. Moreover, we just use the laptop whose configurations have been given in this paper. Therefore, this work provides an efficient trajectory planning method, which is expected to realize real-time trajectory planning considering the error correction. This efficient method also lays a foundation for the realization of trajectory planning with error correction with failure tolerance based on Monte Carlo sampling.

6. Conclusions

To achieve trajectory planning of UAVs with fault tolerance, this paper respectively proposes two trajectory planning methods based on the definite rule and Monte Carlo sampling. By establishing five models, designing corresponding algorithms, and developing computational programs based on MATLAB, the results of trajectory planning of UAVs are obtained for a practical problem. According to the analyses and discussions on the results, the conclusions are drawn as follows.

(1) The results obtained by trajectory planning methods based on the definite rule and Monte Carlo sampling are feasible and reasonable, which achieve the planning objectives that the total trajectory length is as short as possible and that the correction points UAV passes are as few as possible.

(2) The trajectory planning method based on the definite rule proposed in this paper has high efficiency and will promise to realize trajectory planning in real time, suitable for those problems without the failure probability of correction points. However, the trajectory planning method of UAVs with fault tolerance (allowing error corrections fails) based on Monte Carlo sampling is more suitable for off-line trajectory planning due to its expensive time cost.

(3) This work provides not only an effective solution for the trajectory planning of UAVs considering error correction without the failure probability but also a feasible solution for the trajectory planning of UAVs which considers the error correction, failure probability on certain correction point after being corrected, and arriving at the destination successfully.

Nomenclature

δ: Vertical or horizontal error increased by the distance of each 1 m flying
α1: Maximum vertical error allowed during the vertical correction
α2: Maximum horizontal error allowed during the vertical correction
β1: Maximum vertical error allowed during the horizontal correction
β2: Maximum horizontal error allowed during the horizontal correction
σ: Probability of a UAV successfully corrected when it passes a correction point with the failure probability
ζ: Maximum remainder error after the flight error is unsuccessfully corrected by correction points with the failure probability
τ: Failure label of a correction point, either “0 or 1” ith iteration.

|C| = Straight line length from the point C i to X

C iC i+1: Trajectory length from the point C i to C i+1

δ1: Vertical error of the starting point at iteration i
δ2: Vertical error of the starting point at iteration i+1

ζ: Maximum remainder error after the flight error is unsuccessfully corrected by correction points with the failure probability
τ: Failure label of a correction point, either “0 or 1” ith iteration.

|C| = Straight line length from the point C i to X

C iC i+1: Trajectory length from the point C i to C i+1

δi: Vertical error of the starting point at iteration i
δi+1: Vertical error of the starting point at iteration i+1

ζi: Maximum remainder error after the flight error is unsuccessfully corrected by correction points with the failure probability
τ: Failure label of a correction point, either “0 or 1” ith iteration.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.
Acknowledgments

This research was supported by the Guangxi Science and Technology Major Project (Grant Nos. GUIKE AA18242036 and GUIKE AA18242037).

References


