Hindawi
Journal of Sensors
Volume 2020, Article ID 8824202, 9 pages
https://doi.org/10.1155/2020/8824202

Research Article

Feedback-Dubins-RRT Recovery Path Planning of UUV in an Underwater Obstacle Environment

Bing Hao,[1] **Zheping Yan**,[2] **Xuefeng Dai**,[1] **and Qi Yuan**[3]

[1]*College of Computer and Control Engineering, Qiqihar University, Qiqihar, Heilongjiang Province, China*
[2]*College of Automation, Harbin Engineering University, Harbin, Heilongjiang Province, China*
[3]*College of Telecommunication and Electronic Engineering, Qiqihar University, Heilongjiang Province, China*

Correspondence should be addressed to Bing Hao; haobing_learning@163.com

Received 29 March 2020; Revised 24 June 2020; Accepted 1 July 2020; Published 1 August 2020

Academic Editor: Bin Gao

In this paper, a UUV (Unmanned Underwater Vehicle) recovery path planning method of a known starting vector and end vector is studied. The local structure diagram is designed depending on the distance and orientation information about the obstacles. According to the local structure diagram, a Rapidly exploring Random Tree (RRT) method with feedback is used to generate a 3D Dubins path that approaches the target area gradually, and the environmental characteristic of UUV reaching a specific target area is discussed. The simulation results demonstrate that this method can effectively reduce the calculation time and the amount of data storage required for planning. Meanwhile, the smooth spatial path generated can be used further to improve the feasibility of the practical application of UUV.

1. Introduction

With the development and advancement of science and technology, more and more underwater special tasks require UUV to complete. After more than ten or twenty hours of work, the recovery of UUV can be carried out on an underwater platform or a surface ship. At this time, both underwater and surface recovery will face the problem of three-dimensional autonomous motion planning for UUV [1–2]. The content of this paper is to assume that UUV and the recovery platform are not in the same plane. UUV can independently plan the path from the initial point to the target point, and it needs to make sure that the attitude angle at the target point is consistent with the recovery platform.

If the environment where UUV recovery is carried out is complicated with dense obstacles, many various obstacle avoidance methods can be used. However, most obstacle avoidance methods require local replanning, which will generate several subpaths. Repeated iterations of subpaths will increase the time spent in planning recovery paths, and the movement time of UUV and the data storage capacity will also be increased.

In [3], path planning is divided into two types: global path planning and local path planning. A graph search method is one of the global path planning methods. One of the graph search methods is Rapidly exploring Dense Tree (RDT) [4]. By supposing that there is a dense sequence of sample points in the C space, the new configuration points of the shortest distance from the current configuration source point are selected iteratively and connected to form a tree structure. An RDT algorithm is called Rapidly exploring Random Trees (RRT) when using a random method to select new configuration points [5]. Frazzoli et al. proposed an algorithm for constructing a two-dimensional random incremental road map [6]. It is assumed that there is a non-collision guidance loop to guide the vehicle from any state (including configuration point and speed) to any desired configuration point. In [7], Griffiths et al. proposed a RRT algorithm for constructing traversable paths by modeling

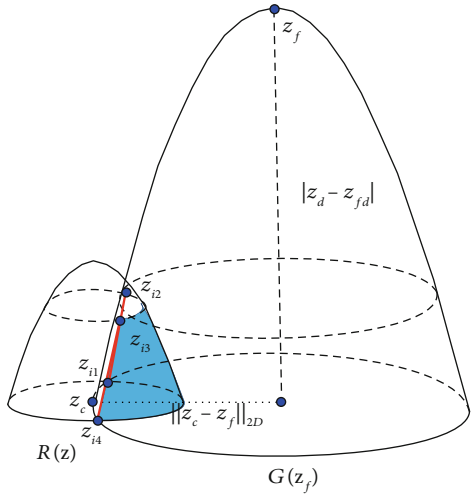

FIGURE 1: The geometry of relative positions of the local-level frame map centered at the current vector and the target vector.

prior environmental data. Examine the branches of the tree as it grows to ensure that the vehicle's turning radius and climb rate constraints are met. The previous research results have fully confirmed that the RRT algorithm [8–10] has a very important value both in theory and in application. However, many previous studies cannot find the most solution to this algorithm theory based on the given cost function [11, 12].

Local path planning algorithms can be divided into heuristic-based path planning algorithms and trajectory generation algorithms [13, 14]. Trajectory generation technology is considered to be a part of the local path planning process, which mainly realizes the construction of a traversal path between two or more path points. The Dubins curve is a type of trajectory generation technology [15]. In order to move the control object from the initial position to the target position, the optimal plane path is constructed in the form of the Dubins curve, and the heading at the initial position and the target position is fixed. The path is a combination of a curve that satisfies the maximum curvature constraint of the control object and a straight line tangent to it.

2. Feedback-Dubins-RRT Recovery Path Planning of UUV

In this paper, the RRT algorithm of global planning and the Dubins curve generation technology of local planning are combined, and the two technologies are extended to a three-dimensional space, taking into account the constraints of UUV, and the autonomous path planning method is given, so as to complete the precise docking of UUV and the recovery platform moves in a three-dimensional environment. The previous path planning method relies on the current sensor data, which can be called the forward path planning method here. The accuracy of forward path planning depends on the accuracy of sensor data and the UUV state model, but these are often difficult to guarantee in long-distance navigation [16].

This paper will imitate the idea of a feedback control loop in control theory, plan the path and feedback the current location information, at the same time compensate the feedback location information, and carry out the next planning according to the feedback position information, which can improve the success rate and accuracy of path planning to a certain extent. The total length of the whole Dubins path can be obtained by the planning algorithm. Since the speed of UUV is seen as a constant, the time at which the UUV reaches the original target point can be calculated.

2.1. Spatial Obstacle Model. Based on the sensors carried by UUV, the range and shape size of the obstacles can be recognized. In this paper, the obstacle that may occur in a recovery environment is described as a spatial ellipsoid model. Let the ith space obstacle model be R_{obi} and the coordinate of the center point of the obstacle model be $R_{\text{obi}}(R_{\text{obin}}, R_{\text{obi}e}, R_{\text{obi}d})$. The radius and height of the obstacle model are a_{obi}, b_{obi}, and c_{obi}, respectively. In most cases, the data of the obstacle shape is detected by carrying sensors. However, after long-term underwater navigation, its navigation error has accumulated to a value that cannot be ignored. In order to avoid obstacles accurately, the obstacle model size is expanded according to navigation errors. B By assuming that the error coefficient of navigation is ξ, the obstacle ellipsoid model is defined as

$$R_{\text{obi}}(\mathbf{z}) \triangleq \left\{ \mathbf{z} \in Z : \sqrt{\frac{(z_n - z_{\text{obin}})^2}{(\xi a_{\text{obi}})^2} + \frac{(z_e - z_{\text{obi}e})^2}{(\xi b_{\text{obi}})^2}} \le 1, \right.$$
$$\left. |z_d - z_{\text{obi}d}| \le \frac{1}{\xi c_{obi}} \sqrt{\frac{{z_{\text{obin}}}^2}{(\xi a_{\text{obi}})^2} + \frac{{z_{\text{obi}e}}^2}{(\xi b_{\text{obi}})^2}} \right\}. \tag{1}$$

Let $\varphi_{\max}$ represent the maximum rotation angle of UUV, $\gamma_{\max}$ represent the maximum navigation angle of UUV, and V represent the speed of UUV. The minimum turning radius of UUV can be calculated as

$$r_{\min} = \frac{V^2 \cos \gamma_{\max}}{g \tan \varphi_{\max}}. \tag{2}$$

2.2. Feedback-Dubins-RRT Path Planning

2.2.1. Spatial Model of a Current Vector and Target Vector. Figure 1 shows the local structure of the current vector model and the target vector model of UUV in a recovery path.

Definitions $R(z_c(t))$ and $R(z_f)$, respectively, represent the ellipsoidal regions with the current vector point $\mathbf{z}_c(t) = [z_{cn}, z_{ce}, z_{cd}, z_{c\phi}]$ and the target vector point $\mathbf{z}_f = [z_{fn}, z_{fe}, z_{fd}, z_{f\varphi}]$ as the center point of the model. The four elements of the vector represent the north-east earth coordinate and the heading angle of UUV at this point, respectively.

$$R(\mathbf{z}_c(t)) \triangleq \left\{ \mathbf{z} \in Z : \sqrt{\frac{(z_n - z_{cn})^2}{a^2} + \frac{(z_e - z_{ce})^2}{b^2}} \leq 1, |z_d - z_{cd}| \leq \frac{1}{c}\sqrt{\frac{z_{cn}^2}{a^2} + \frac{z_{ce}^2}{b^2}} \right\},$$
$$R(\mathbf{z}_f) \triangleq \left\{ \mathbf{z} \in Z_{\text{free}} : \sqrt{\frac{(z_n - z_{fn})^2}{a^2} + \frac{(z_e - z_{fe})^2}{b^2}} \leq 1, |z_d - z_{fd}| \leq \frac{1}{c}\sqrt{\frac{z_{fn}^2}{a^2} + \frac{z_{fe}^2}{b^2}} \right\}. \tag{3}$$

Here, $\mathbf{Z}$ represents the set of all space vectors. $\mathbf{Z}_{\text{obs}}$ represents the set of position vectors where UUV is located which inevitably intersect with obstacles. Let $\mathbf{Z}_{\text{free}} = \mathbf{Z} \setminus \mathbf{Z}_{\text{obs}}$.

Then, the goal of the planning is tried to find a path so that all vectors on the path are in the set and satisfy the kinematic constraints of UUV. a, b, and c are the parameters of the ellipsoid model, and its value can be designed according to the navigation environment.

Construct another spatial ellipsoid based on the current vector and the target vector $G(\mathbf{z}_f(t))$, the geometric positional relationship is shown in Figure 1, and the expressed area is calculated as the following equation.

$$G(\mathbf{z}_f(t)) \triangleq \left\{ \mathbf{z} \in \mathbf{Z} : \sqrt{\frac{(z_n - z_{fn})^2}{A^2} + \frac{(z_e - z_{fe})^2}{B^2}} = 1, |z_d - z_{fd}| \leq \frac{1}{C}\sqrt{\frac{z_{fn}^2}{A^2} + \frac{z_{fe}^2}{B^2}} \right\}, \tag{4}$$

where A, B, and C are the parameters of the ellipsoid, $\max(A, B) = \|z_c(t) - z_f\|_{2D}$.

$\partial R(\mathbf{z}_c(t))$ represents the boundary of the current vector region model $R(\mathbf{z}_c(t))$.

$$\partial R(\mathbf{z}_c(t)) \triangleq \left\{ \mathbf{z} \in \mathbf{Z} : \sqrt{\frac{(z_n - z_{cn})^2}{a^2} + \frac{(z_e - z_{ce})^2}{b^2}} = 1, |z_d - z_{cd}| \leq \frac{1}{c}\sqrt{\frac{z_{cn}^2}{a^2} + \frac{z_{ce}^2}{b^2}} \right\}. \tag{5}$$

$z_{i1}z_{i2}$ and $z_{i3}z_{i4}$ are at the intersection of ellipsoid $G(\mathbf{z}_f)$ and $R(z_c(t))$.

$C(\mathbf{z})$ represents the vector on the boundary $\partial R(\mathbf{z}_c(t))$, which is also located in $G(\mathbf{z}_f)$. Next, the problem of recovery path planning can be converted to find a collision-free path on $\partial R(\mathbf{z}_c(t))$, so that the distance between UUV and ellipsoid $G(\mathbf{z}_f)$ gradually reduces until UUV can reach the target vector $\mathbf{z}_f$.

Definition 1. If $\sqrt{((z_{cn} - z_{fn})^2/a^2) + ((z_{ce} - z_{fe})^2/b^2)} > 1$ and $|z_{cd} - z_{fd}| > 1/c\sqrt{(z_{fn}{}^2/a^2) + (z_{fe}{}^2/b^2)}$, then $\mathbf{z} \notin R(\mathbf{z}_f)$.

Definition 2. If $\sqrt{((z'_n - z_{fn})^2/a^2) + ((z'_e - z_{fe})^2/b^2)} \leq \sqrt{((z_{cn} - z_{fn})^2/a^2) + ((z_{ce} - z_{fe})^2/b^2)}$ and $|z'_d - z_{fd}| \leq |z_d - z_{fd}|$, then $\mathbf{z}' \in C(\mathbf{z})$.

Definition 3. Two integers ρ_r and ρ_d satisfy

$$\rho_r < \min(a, b), \quad \rho_d < \frac{1}{c}\sqrt{\frac{z_{fn}^2}{a^2} + \frac{z_{fe}^2}{b^2}}. \tag{6}$$

$O(\mathbf{z})$ is a subset of $C(\mathbf{z})$.

$$O(\mathbf{z}) \triangleq \left\{ \sqrt{(z_{cn} - z_{fn})^2 + (z_{ce} - z_{fe})^2} - \sqrt{(z'_n - z_{fn})^2 + (z'_e - z_{fe})^2} \geq \rho_r, \mathbf{z}' \in C(\mathbf{z}) : \left(|z_{cd} - z_{fd}| - |z'_d - z_{fd}| \geq \rho_d\right) \right\}. \tag{7}$$

The distance of the vector to the target vector is less than the distance from the current point $\mathbf{z}_c(t)$ to the target vector.

2.2.2. The Basic Idea to Generate a Recovery Path. This section describes the Feedback-Dubins-RRT algorithm which produces a collision-free path from the current vector $\mathbf{z}_c(t)$ to the target model region in $O(\mathbf{z})$, such that the distance between the current vector $\mathbf{z}_c(t)$ and the target vector z_f is progressively shortened.

Step 1. Initialize the set of vectors $\{\mathbf{z}\}$. Let the current vector $\mathbf{z}_c(t)$ be the only vector in the tree structure $\{\mathbf{z}\}$.

Step 2. Progressive growth of random trees:

Step 2.1. Random node vectors $\mathbf{z}_{\text{rand}}$ are generated in set $R(\mathbf{z})\text{O}(\mathbf{z})$ and set $O(\mathbf{z})$ with a fixed probability P and $1 - P$, respectively.

Step 2.2. Find the node vector $\mathbf{z}_{\text{near}}$ closest to the randomly generated vector $\mathbf{z}_{\text{rand}}$ in the structure of the existing tree.

Step 2.3. Connect the node vector $\mathbf{z}_{\text{near}}$ and the random vector $\mathbf{z}_{\text{rand}}$ with the Dubins curve.

Step 2.4. Check whether the generated Dubins curve is collision-free and meets the kinematic constraints of UUV.

Step 2.5. If the result of Step 2.4 is collision-free, the random vector $\mathbf{z}_{\text{rand}}$ is added to the tree structure to become the node vector of the tree and the Dubins curve path is used as the branch of the tree.

Step 3. Iteration of complete growth of random trees:

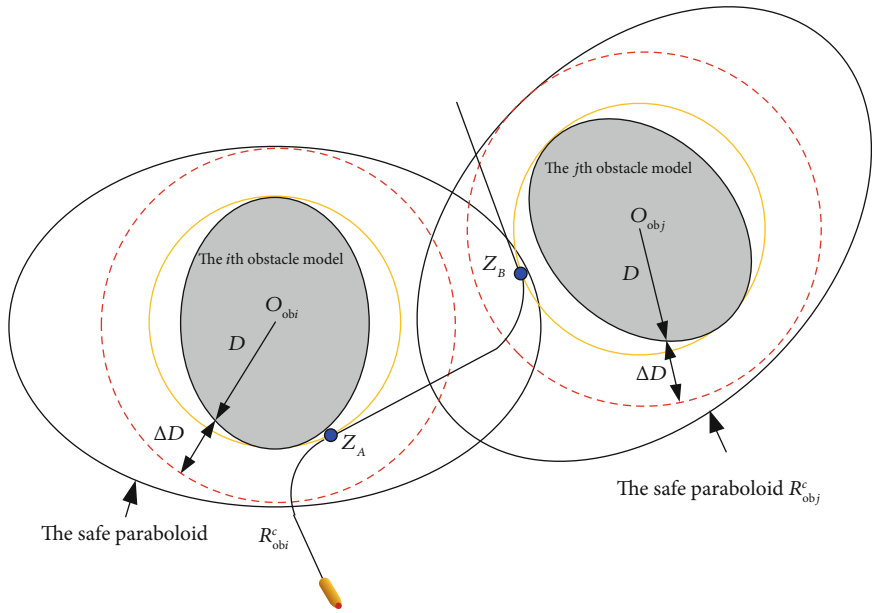

Figure 2: The distance between obstacles meets the traversable condition.

Table 1: Center point coordinates and parameters of obstacles.

Obstacle number	Coordinate (m)	a_{obi} (m)	b_{obi} (m)	c_{obi} (m)
1	(100, 200, 435)	30	40	13
2	(440, 300, 425)	50	30	15
3	(440, 740, 300)	40	30	40
4	(900, 600, 430)	40	30	14
5	(550, 550, 325)	32	40	25
6	(750, 150, 450)	20	20	10
7	(900, 900, 325)	60	50	35

Step 3.1. Check if there is a vector in $O(\mathbf{z})$ that has been connected to the structure of the tree.

Step 3.2. If the condition in Step 3.1 is met, the path from the current node vector to this vector is extracted. If the condition in Step 3.1 is not satisfied, a node vector is randomly generated and connected to the current node vector and the Dubins path between them is extracted.

Step 4. Determine whether the growth of random trees ends: the algorithm ends when the vector in $O(\mathbf{z})$ becomes the structure of the tree or the preset number of cycles is reached.

2.2.3. Existence Condition of a Collision-Free Path. For the environment with dense obstacles, it is needed to give the existence conditions of collision paths. Let $(R_{ob1}, R_{ob2}, \cdots, R_{obn})$, $n \in N$ represent the set of all obstacles. R_{obi}, R_{obj}, $i \neq j$, $i, j \in N$ are two arbitrarily adjacent obstacles. The shortest distance from the ith obstacle to the jth obstacle is defined as

$$d_{ij} = \min_{p_i \in \partial R_{obi}, p_j \in \partial R_{obj}} \sqrt{\left(p_{in} - p_{jn}\right)^2 + \left(p_{ie} - p_{je}\right)^2}. \quad (8)$$

Here, $p_i = (p_{in}, p_{ie}, p_{id})$ and $p_j = (p_{jn}, p_{je}, p_{jd})$ represent any point on the boundary of the ith obstacle and the jth obstacle.

According to the aforementioned method, a threatening circle is set on the outside of each obstacle. If there is a collision-free path through a dense obstacle, the distance between two obstacles should be greater than the difference between the radius of the threatening circle and the minimum turning radius. In other words, if there is a collision-free path through obstacles in a dense-obstacle environment, no point on the threaten circle can be covered by other obstacle areas.

Definition 4. A dense-obstacle environment is traversable if and only if the distance between the ith obstacle and the jth obstacle satisfies

$$d_{ij} > \max\left\{\left[\min\left(a^c_{obi}, b^c_{obi}\right) - \max\left(a_{obi}, b_{obi}\right)\right], \left[\min\left(a^c_{obj}, b^c_{obj}\right) - \max\left(a_{obj}, b_{obj}\right)\right]\right\}. \quad (9)$$

a^c_{obi}, b^c_{obi}, a^c_{obj}, b^c_{obj} are the parameters of threatening zones R^c_{obi} and R^c_{obj}, respectively. $\min\left(a^c_{obi}, b^c_{obi}\right)$, $\min\left(a^c_{obj}, b^c_{obj}\right)$ are the radius of the threatening circle of the ith obstacle and the jth obstacle, respectively. a_{obi}, b_{obi}, a_{obj}, b_{obj} are the parameters of the ith obstacle and the jth obstacle, respectively.

Theorem 5. *It is assumed that obstacle-intensive environments are traversable. The current node vector is* $\mathbf{z}_c(t) = [z_{cn}(t), z_{ce}(t), z_{cd}(t), z_{c\phi}(t)]$. *The minimum turning radius of UUV is* $r_{\min}$. $R_{obi} \in (R_{ob1}, R_{ob2}, \cdots, R_{obn})$ *represents any obstacle model, and its radius and height are* a_{obi}, b_{obi}, *and* c_{obi}, *respectively. Ellipsoid* R^c_{obi} *is a model expanded according to the obstacle model* R_{obi}, *and its parameters are* a^c_{obi}, b^c_{obi}, *and* c^c_{obi} *and*

Table 2: Distance between different obstacles.

Spacing (m)	Obstacle 1	Obstacle 2	Obstacle 3	Obstacle 4	Obstacle 5	Obstacle 6	Obstacle 7
Obstacle 1	*	308.06	559.25	Not adjacent	521.21	Not adjacent	Not adjacent
Obstacle 2	308.06	*	Not adjacent	450.08	241.05 Not passable	315.49	Not adjacent
Obstacle 3	559.25	Not adjacent	*	Not adjacent	171.26 Not passable	Not adjacent	427.20
Obstacle 4	Not adjacent	450.08	Not adjacent	*	314.36	441.61	264.71
Obstacle 5	521.21	241.05 Not passable	171.26 Not passable	314.36	*	402.52	430.63
Obstacle 6	Not adjacent	315.49	Not adjacent	441.61	402.52	*	Not adjacent
Obstacle 7	Not adjacent	Not adjacent	427.20	264.71	430.63	Not adjacent	*

satisfied $\min(a^c_{obi}, b^c_{obi}) = \sqrt{[\max(a_{obi}, b_{obi}) + r_{\min}]^2 - r^2_{\min}}$ *and* $c^c_{obi} = c_{obi}$. *If the current node vector* $\mathbf{z}_c(t)$ *is not included in ellipsoid* R^c_{obi}, *then a collision-free path can be found by the Feedback-Dubins-RRT algorithm.*

Proof. As shown in Figure 2, the obstacle circle is generated with the radius of $D = \max(a_{\text{obi}}, b_{\text{obi}})$ and a threatening circle is set to the radius of $D + \Delta D$, where $D + \Delta D = \min(a^c_{\text{obi}}, b^c_{\text{obi}})$. This radius is the minimum distance between UUV and obstacles when UUV navigates with the maximum rotation angle and maximum navigation angle. Z' is the tangent point between the circle generated with the minimum turning radius and the obstacle circle. By projecting their geometric positions in the x-y plane, the geometric relationship is expressed as follows:

$$\begin{aligned}&(D + \Delta D + r_{\min}\sin\alpha_i)^2 + (r_{\min}\cos\alpha_i)^2 = (D + r_{\min})^2,\\&(D + r_{\min}\sin\alpha_i)^2 + \Delta D^2 + 2\Delta D(D + r_{\min}\sin\alpha_i)\\&\quad + (r_{\min}\cos\alpha_i)^2 = D^2 + r^2_{\min} + 2Dr_{\min}.\end{aligned} \tag{10}$$

Merge and get the following form:

$$\begin{aligned}&D^2 + 2\Delta D(D + r_{\min}\sin\alpha_i) + 2Dr_{\min}\sin\alpha_i - 2Dr_{\min} = 0,\\&\Delta = 4(D + r_{\min}\sin\alpha_i)^2 - 4\times 2Dr_{\min}(\sin\alpha_i - 1)\\&\quad = 4\left[D^2 + (r_{\min}\sin\alpha_i)^2 + 2Dr_{\min}\sin\alpha_i - 2Dr_{\min}(\sin\alpha_i - 1)\right]\\&\quad = 4\left[D^2 + (r_{\min}\sin\alpha_i)^2 + 2Dr_{\min}\right],\\&\Delta D = \frac{-2(D + r_{\min}\sin\alpha_i) \pm \sqrt{4\left[D^2 + (r_{\min}\sin\alpha_i)^2 + 2Dr_{\min}\right]}}{2}\\&\quad = -(D + r_{\min}\sin\alpha_i) \pm \sqrt{D^2 + (r_{\min}\sin\alpha_i)^2 + 2Dr_{\min}}.\end{aligned} \tag{11}$$

ΔD is positive, so there is

$$\Delta D = -(D + r_{\min}\sin\alpha_i) + \sqrt{D^2 + (r_{\min}\sin\alpha_i)^2 + 2Dr_{\min}}. \tag{12}$$

When $\alpha_i = 0$,

$$\begin{aligned}\Delta D_{\max} &= -D + \sqrt{(D + r_{\min})^2 - r^2_{\min}},\\D + \Delta D_{\max} &= \sqrt{(D + r_{\min})^2 - r^2_{\min}}.\end{aligned} \tag{13}$$

Therefore, if the current vector is not contained by the ellipsoid R^c_{obi}, UUV is navigating with the direction angle $\pi/2$ or $-\pi/2$ and the collision-free path can be obtained by the planning algorithm.

If the current vector $\mathbf{z}_c(t)$ is not included in the ellipsoid R^c_{obi}, then there is an arbitrary node vector $\mathbf{Z}_A$ that approaches the boundary of the obstacle R_{obi} with the direction angle $\pi/2$ or $-\pi/2$ and makes the path of the current node vector $\mathbf{z}_c(t)$ to the vector $\mathbf{Z}_A$ collision-free.

In addition, the obstacle environment is assumed to be traversable, indicating that $\mathbf{Z}_A$ must be outside the ellipsoid R^c_{obj}. In other words, the next node vector $\mathbf{Z}_B$ can also be found to approach the boundary of the obstacle R_{obj} with direction angle $\pi/2$ or $-\pi/2$ and make $\mathbf{Z}_A$ to $\mathbf{Z}_B$ a collision-free path. This process is repeated multiple times until a collision-free path from the current vector to the target vector is obtained.

2.3. Feedback-Dubins-RRT Path Pruning. Although it is efficient and feasible to use the proposed algorithm for path planning, the results of the algorithm are still not optimal due to the random growth of spatial trees. In order to reduce unnecessary navigation and rotation of UUV, some unnecessary path node vectors need to be deleted. In this paper, a simple and effective method is used to remove the unnecessary path node vectors so that the recovery path can be quickly generated. The generated Feedback-Dubins-RRT vector set is trimmed as follows:

Step 1. All vectors from the start vector to the target vector generated by the foregoing method are placed into the set $(\mathbf{z}_1, \cdots, \mathbf{z}_N)$, where $\mathbf{z}_1$ represents the starting vector and $\mathbf{z}_N$ represents the target vector.

Step 2. Let the set of node vectors of the recovery path be an empty set. Then, let $j = N$ and add vector $\mathbf{z}_j$ to the trimmed set of recovery path vectors.

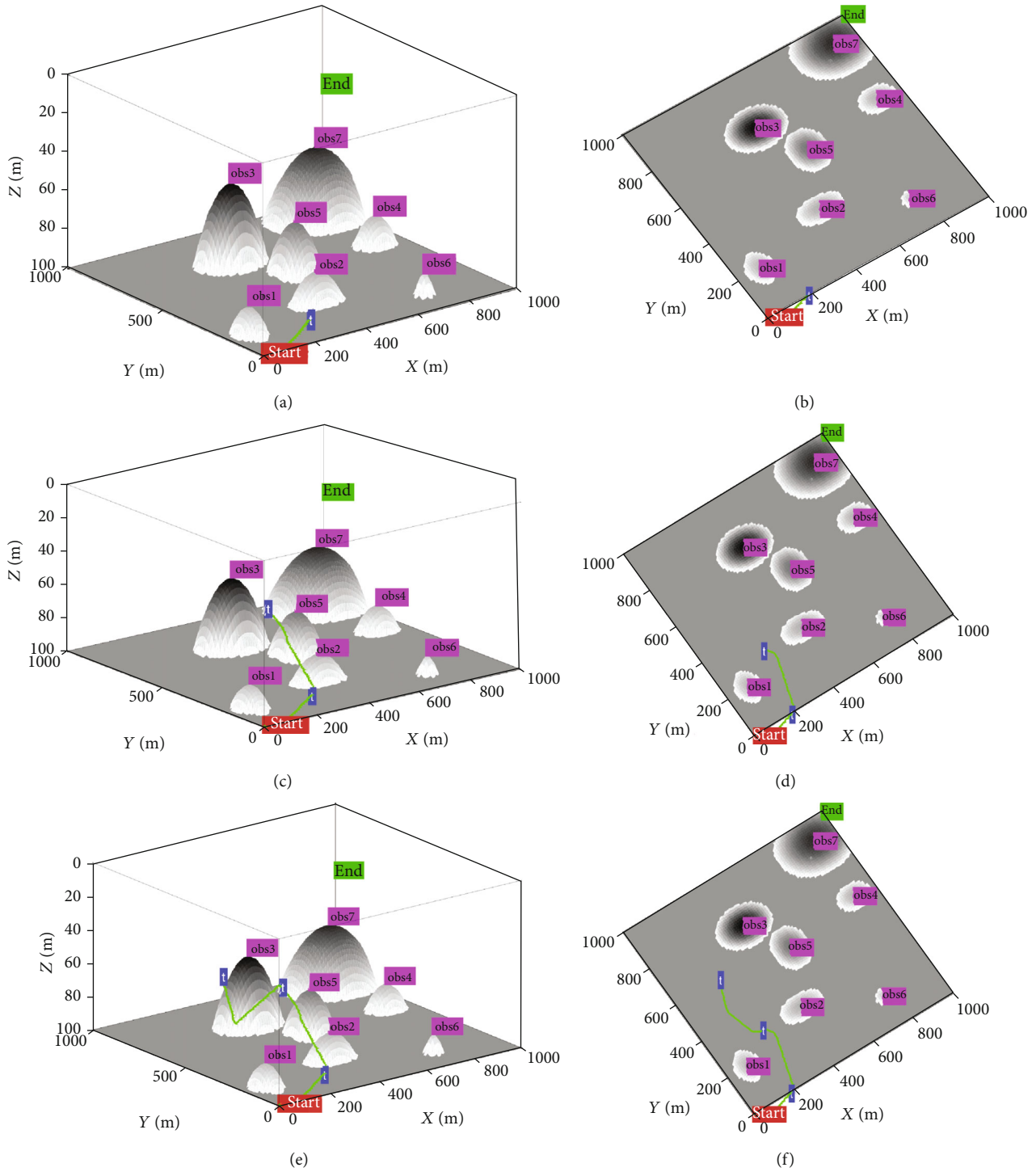

Figure 3: Continued.

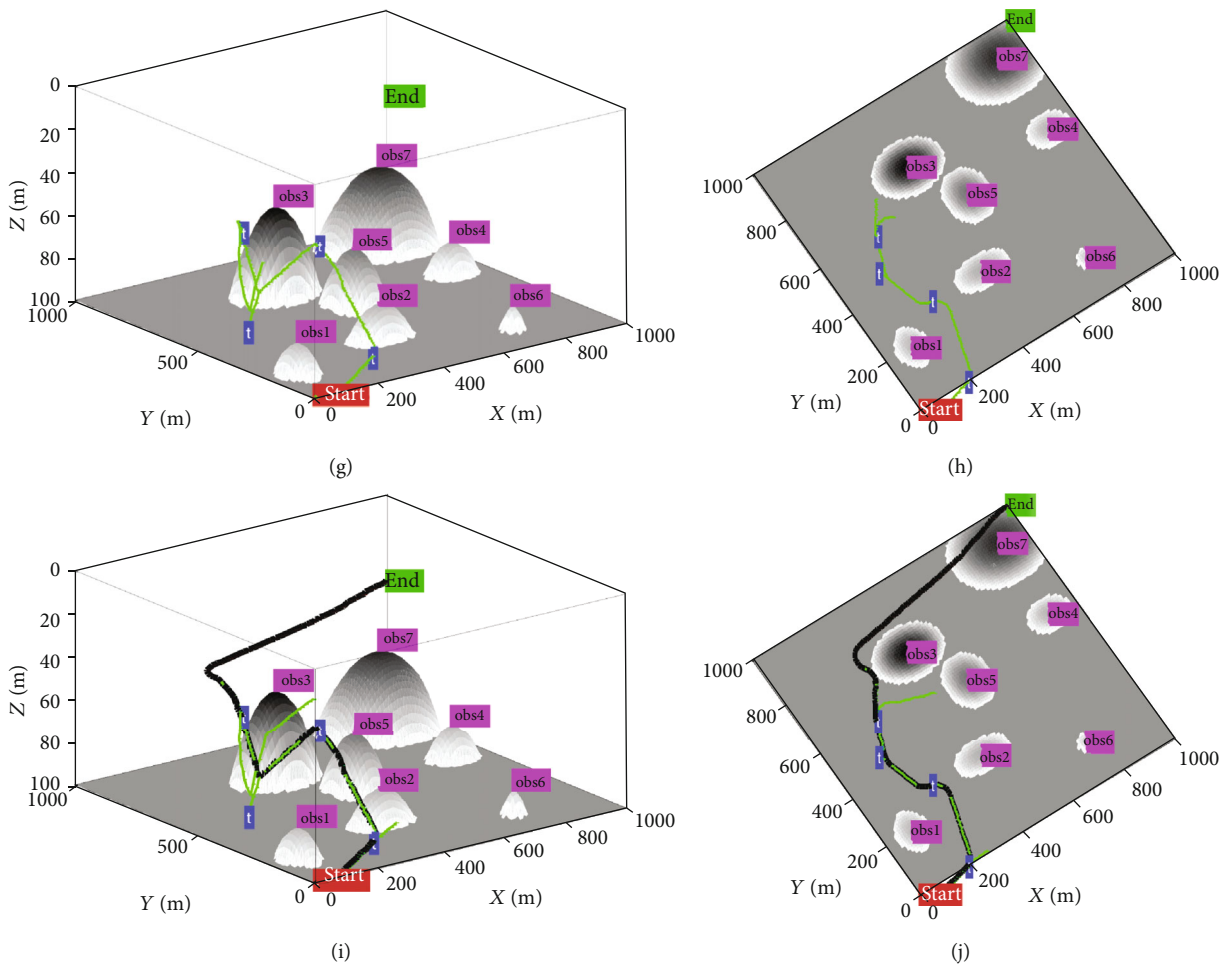

FIGURE 3: The recovery path generated by the Feedback-Dubins-RRT algorithm: (a) three-dimensional graph of the second node generated, (b) top view of the second node generated, (c) three-dimensional graph of the fourth node generated, (d) top view of the fourth node generated, (e) three-dimensional graph of the sixth node generated, (f) top view of the sixth node generated, (g) three-dimensional graph of the eighth node generated, (h) top view of the eighth node generated, (i) three-dimensional graph of all the nodes generated, and (j) top view of all the nodes generated.

Step 3. Check whether there is a collision-free Dubins path between the node vector $\mathbf{z}_i$ and other node vectors in the order of the foot labels, $i \in [1, \cdots, j-1]$.

Step 4. When a node vector is found and the Dubins path between it and other node vectors is collision-free, let $j = i$ and add vector $\mathbf{z}_i$ to the set of trimmed path points. Repeat this process until a complete collision-free path is generated from the starting vector to the target vector.

By trimming the path node vector generated by the Feedback-Dubins-RRT algorithm, some unnecessary path node vectors are deleted, so that the original path can be optimized.

3. Simulation

In this section, a set of cases will be given to verify the effectiveness of the proposed recovery path planning algorithm.

In the three-dimensional geodetic coordinate system, the range of length, width, and height of a space is 1000 m, 1000 m, and 100 m. The initial point position is set at (0, 0, 98), and the target point position is set at (1000, 1000, 40). In order to prove the generality of the algorithm, the heading angle and pitch angle of UUV at the initial position and the target position are randomly generated. Suppose that there are seven intensive obstacles in the environment. The coordinates of the center point and its parameters of the obstacles are shown in Table 1. The distance between adjacent obstacles is shown in Table 2, in which the second and fifth obstacles, as well as the third and fifth obstacles, do not satisfy the conditions adopted in Theorem 5. The program has been running many times, and the results of the two versions are given in Figures 3 and 4. The radius of the Dubins path depends on the scope of the area. $r_{\min} = 0.05 * dx$, where dx is the maximum value of the length, width, and height of the area.

In Figure 3, the growth of the random tree and the process of obtaining the final path can be seen in detail at dynamically intercepting the output of the second, fourth, sixth, and eighth path nodes generated.

Figure 4 shows the result of the rerun of the same initial conditions as Figure 3. It can be seen from Figure 4 that the

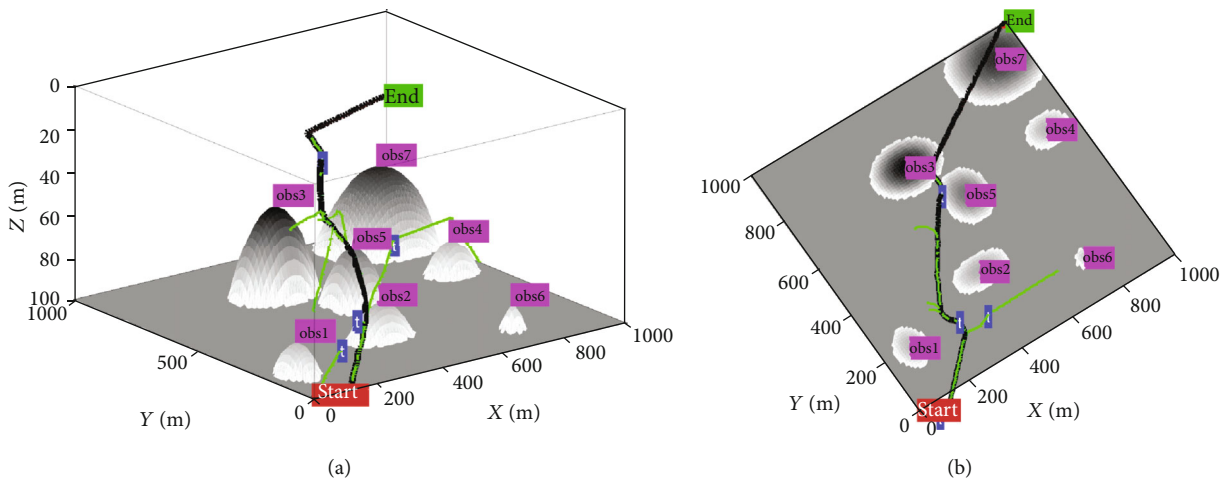

Figure 4: The second recovery path generated by the Feedback-Dubins-RRT algorithm: (a) three-dimensional graph of all the nodes generated and (b) top view of all the nodes generated.

path passes through two obstacles ob1 and ob2 which are determined to be inaccessible in Theorem 5, and the path does not intersect the obstacle. The path satisfies the constraints and reaches the target point. The length of the path is 1933 m. Thus, it can be seen that Theorem 5 gives a sufficient condition to ensure traversability rather than a necessary condition.

4. Conclusion

In this paper, the planning method of a spatial path is given for the known recovery starting point vector and target point vector in a 3D environment. For the obstacle-intensive environment, a method which constructs a local structure map between the obstacle and the target vector based on the distance and orientation information is proposed. The three-dimensional Dubins planning path is generated by Feedback-Dubins-RRT according to the local structure map, and the obstacle can be avoided infinitely and reach the target area. The simulation results show that the proposed algorithm can solve the problem of path planning and obstacle avoidance in a three-dimensional environment.

The Feedback-Dubins-RRT recovery path planning algorithm still needs to be improved in the following two aspects. (1) In a three-dimensional environment, when the environment is unknown or there are dynamic obstacles, the path planning algorithm in this paper needs to be improved to add a dynamic obstacle avoidance strategy to generate a real-time planned path. (2) The path generated by Feedback-Dubins-RRT can only be asymptotically optimal, and the algorithm idea needs to be improved to generate the optimal path under the current initial conditions.

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 is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is partially supported by the Basic Scientific Research Business Cost Scientific Research Project of Heilongjiang Provincial University (135209236 and 135309453) and the Joint Guiding Project of Natural Science Foundation of Heilongjiang Province under Grant LH2019F038.

References

[1] M. D. Feezor, F. Yates Sorrell, P. R. Blankinship, and J. G. Bellingham, "Autonomous underwater vehicle homing/docking via electromagnetic guidance," *IEEE Journal of Oceanic Engineering*, vol. 26, no. 4, pp. 515–521, 2001.

[2] J.-Y. Park, B.-H. Jun, P.-M. Lee, Y.-K. Lim, and J.-H. Oh, "Modified linear terminal guidance for docking and a time-varying ocean current observer," in *2011 IEEE Symposium on Underwater Technology and Workshop on Scientific Use of Submarine Cables and Related Technologies*, pp. 1–6, Tokyo, Japan, April 2011.

[3] D. Ferguson and A. Stentz, "Using interpolation to improve path planning: the field D∗ algorithm," *Journal of Field Robotics*, vol. 23, no. 2, pp. 79–101, 2006.

[4] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[5] S. G. Faal and C. D. Onal, "Regionally growing random trees: a synergistic motion planning and control algorithm for dynamic systems," in *Proceedings of 2016 IEEE Internationa Conference on Automation Science and Engineering*, pp. 141–147, Fort Worth, USA, 2016.

[6] E. Frazzoli, M. A. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 1, pp. 116–129, 2002.

[7] S. Griffiths, J. Saunders, A. Curtis, B. Barber, T. McLain, and R. Beard, "Maximizing miniature aerial vehicles," *IEEE Robotics and Automation Magazine*, vol. 13, no. 3, pp. 34–43, 2006.

[8] W. Aguilar and S. Morales, "3D environment mapping using the Kinect V2 and path planning based on RRT algorithms," *Electronics*, vol. 5, no. 4, p. 70, 2016.

[9] D. Connell and H. M. La, "Dynamic path planning and replanning for mobile robots using RRT," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 1429–1434, Banff, AB, Canada, October 2017.

[10] Y. Wang, P. Pandit, A. Kandhari, Z. Liu, and K. A. Daltorio, "Rapidly exploring random tree algorithm-based path planning for worm-like robot," *Biomimetics*, vol. 5, no. 2, p. 26, 2020.

[11] A. Pandey, V. S. Panwar, M. E. Hasan, and D. R. Parhi, "V-REP-based navigation of automated wheeled robot between obstacles using PSO-tuned feedforward neural network," *Journal of Computational Design and Engineering*, vol. 7, 2020.

[12] A. Pandey, A. K. Kashyap, D. R. Parhi, and B. K. Patle, "Autonomous mobile robot navigation between static and dynamic obstacles using multiple ANFIS architecture," *World Journal of Engineering*, vol. 16, no. 2, pp. 275–286, 2019.

[13] L. Singh and J. Fuller, "Trajectory generation for a UAV in urban terrain, using nonlinear MPC," in *Proceedings of the 2001 American Control Conference. (Cat. No.01CH37148)*, pp. 2301–2308, Arlington, VA, June 2001.

[14] D. H. Shim, Hoam Chung, and S. S. Sastry, "Conflict-free navigation in unknown urban environments," *IEEE Robotics and Automation Magazine*, vol. 13, no. 3, pp. 27–33, 2006.

[15] L. E. Dubins, "On curves of minimal length with a constraint on average curvature with prescribed initial and terminal positions and tangents," *American Journal of Mathematics*, vol. 79, no. 3, pp. 497–516, 1957.

[16] H. Yu and R. Beard, "A vision-based collision avoidance technique for micro air vehicles using local-level frame mapping and path planning," *Autonomous Robots*, vol. 34, no. 1-2, pp. 93–109, 2013.