Hindawi
Complexity
Volume 2018, Article ID 4359036, 17 pages
https://doi.org/10.1155/2018/4359036

WILEY | Hindawi

*Research Article*

# Smooth Path Planning for Robot Docking in Unknown Environment with Obstacles

**Peng Cui**,[1,2] **Weisheng Yan**,[1,2] **Rongxin Cui**,[1,2] **and Jiahui Yu**[3]

[1]*Research & Development Institute of Northwestern Polytechnical University in Shenzhen, Shenzhen, Guangdong 518057, China*
[2]*School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an, Shaanxi 710072, China*
[3]*School of Automation and Electrical Engineering, Shenyang Ligong University, Shenyang, Liaoning 110159, China*

Correspondence should be addressed to Weisheng Yan; wsyan@nwpu.edu.cn

Received 8 July 2018; Accepted 26 September 2018; Published 8 November 2018

Guest Editor: Zhaojie Ju



This paper presents an integrated approach to plan smooth path for robots docking in unknown environments with obstacles. To determine the smooth collision-free path in obstacle environment, a tree structure with heuristic expanding strategy is designed as the foundation of path planning in this approach. The tree employs 3D Dubins curves as its branches and foundation for path feasibility evaluation. For the efficiency of the tree expanding in obstacle environment, intermediate nodes and collision-free branches are determined inspired by the elastic band theory. A feasible path is chosen as the shortest series of branches that connects to the docking station after the sufficient expansion of the tree. Simulation results are presented to show the validity and feasibility of the proposed approach.

## 1. Introduction

Nowadays autonomous robots are widely used in danger and complex tasks such as aerial surveillance, ocean exploration, and automated transportation [1–5]. As the persistent tasks of robots are increasing, their performances are constrained by the battery capacity and the driver efficiency [6–9]. To address this problem, docking robots to the deployed stations and recharging them attract more and more attention of the researchers in practical applications for their convenience and efficiency [10, 11]. However, if the surrounding environment of the docking station is unknown or partially unknown beforehand, especially for nonholonomic robots such as autonomous underwater vehicles (AUVs) and unmanned ground vehicles, the safety of the robot in docking process may be endangered by the existing obstacles [12–16]. Furthermore, short-sighted collision avoidance maneuvers may also lead to the failure of docking if the docking pose adjustment is not fully considered [17–19]. In this case, the study of smooth path planning technique for robot docking in unknown environment with obstacle is meaningful and practical.

Several efforts have been made by researchers to address the path planning problem for robots docking already [13, 15, 20–27]. Among them, one of the most popular approaches is the artificial potential field (APF) method. The basic concept of the APF method assumes that the robot is attracted by the goal and repulsed by the barriers. The attractive effect and repulsive effect are indicated by the attractive potential field and repulsive potential field, respectively. With properly designed potential functions, the robot will be navigated to the goal point and avoid collision in real time. However, the local minimal points of the total potential field in the navigation process may cause the local minima problem of the APF method which will prevent the robot from approaching the destination [24]. In [22], the APF approach is adopted to generate the desired trajectory for an AUV homing and docking. Through modeling the environment with designed potential fields, the minimum field vector is used to navigate the robot to dock in environment with known stationary obstacles. In [23], the collision-free path for AUV docking to a submarine is obtained via the APF approach while considering the kinematic constraints of the vehicle. Local minima problem is also considered in this

paper and addressed with particularly designed potential functions and parameters.

Another popular approach applied in robot docking is to integrate feasible curves with optimization strategies for path generation [13, 25, 26, 28]. In this approach, the resultant path consists of several path segments which are connected to the waypoints. Firstly, the waypoints are determined and optimized by the global path planners which are usually based on certain searching algorithm or optimization technique. Then, path segments are generated in the form of space curves such as Dubins curves and B-splines so that the motion characters of the robots can be easily addressed [15]. Thus, the quality and efficiency of the integrated approach mainly depend on the feasibility of the global path planner. In [25], the determination of the feasible path for AUV docking is considered in an environment with stationary obstacles. Resultant path is obtained as a series of 2D lines and circles determined by the shortest path faster algorithm. In [27], the path planning problem for multiple underwater gliders rendezvous is addressed while considering the minimal energy consumption target. Modified Dubins curves are presented and combined with the genetic algorithm (GA) to determine the resultant paths. In [28], an integrated motion planning method is proposed for the robots with nonlinear constraints. The rapidly exploring random tree star (RRT$*$) algorithm is adopted as the searching strategy in the clustered environment. The RRT$*$ approach is a famous sampling-based path planning approach, which determines the collision-free path via sampling and searching of the workspace. As the increase of sample nodes in the workspace, it can provide optimal solution for the path planning problem. A neural network is designed to generate the feasible curves for the RRT$*$ algorithm to obtain the near-optimal trajectory.

Although various approaches for robot docking have been proposed, the study of smooth path planning technique for robot docking in unknown environment is still meaningful. To the best of the authors' knowledge, the problem of balancing the feasibility and efficiency of collision-free path planning for docking in unknown environment, especially in the environment with moving obstacles, has not been well addressed yet. For example, the path planning result of the APF approach is sensitive to the parameters of the designed potential fields. Designing the feasible potential fields and choosing proper parameters in the application in unknown environment require the skill and dexterity. Besides, most of the integrated approaches prescribed above adopt the biological intelligence or geometrical model searching method to determine the waypoints in the workspace of robots. The feasibilities of the resultant paths rely on the ample prior information about the environment for robots docking, which is hard to satisfy in practical applications.

To address this problem, this paper presents a smooth path planning approach for robot docking in unknown environment with obstacles. Our approach is motivated by the notion of integrating space curves with heuristic searching algorithms. Considering the practical applications, 3D Dubins curves are adopted as the basic path segments in our approach because of their smoothness and simplicity. Their characters are also utilized to design the collision prediction and path feasibility evaluation functions, which provide the foundation of our path planning approach. Then we designed a tree structure with the Dubins branches, which is named the Dubins tree, to explore the environment for feasible path determination and help avoid collision with obstacles. The feasible path for docking is obtained based on the sufficiently expanded Dubins tree. Our approach is executed in a typical planning and replanning mode and only needs the local information, which improves the adaptability of our approach in dynamic environment. To enhance the efficiency of the tree expansion, a heuristic searching strategy is proposed to keep the feasibility of the resultant path based on the elastic band theory.

The main contribution of this paper can be summarized as follows. The first contribution is that we presented the multisegment path calculation and feasibility evaluation method, which are derived from the detailed analysis of the characters of 3D Dubins curves in obstacle environment and thus applicable for nonholonomic robots path planning. The second contribution is that we proposed an efficient path replanning strategy for the integrated approaches, where both the necessity of collision avoidance and the feasibility of resultant path are sufficiently considered to optimize the space searching process. This effort can reduce the workload of path planning in crowded environment, which makes our approach promising to apply in time-sensitive tasks. The third contribution of this paper is that our approach can be used to address the smooth path planning problem for the robot with kinematic constraints in unknown environment with both stationary obstacles and moving obstacles.

The rest of this paper is organized as follows. The smooth path planning problem for robot docking is formulated in Section 2. In Section 3, the analysis and utilization of the Dubins curves in the path planning tasks are presented in detail. The conception and procedures, as well as the algorithm, of our path planning approach are presented in Section 4. Simulations and discussions of the proposed approach are given in Section 5. The conclusion and future work are provided in Section 6.

## 2. Problem Statement

In this paper, we formulate the path planning task for robot docking as a shortest trajectory planning problem with prescribed terminal conditions. To be practical, a 3D mobile robot with kinematic constraints and constant cruising speed is considered as the mathematic model for path planning. In this case, the constrained mobility of the robot is assumed to be indicated by the minimal turning radius which is given by $R_{\min}$. The configuration of the robot $\mathscr{C}_r$ is defined as

$$\mathscr{C}_r = \left\{P_r, \overrightarrow{V}_r\right\}, \tag{1}$$

where $P_r$ is the position of the robot and $\overrightarrow{V}_r$ is the velocity of the robot. Since the cruising speed of the robot is constant, we have $\|\overrightarrow{V}_r\| = v$ where $v$ is a positive constant. The limited detection ability of the robot is also considered in this paper, which is common in the practical applications outdoors, and written as the maximal detection range $R_s$.

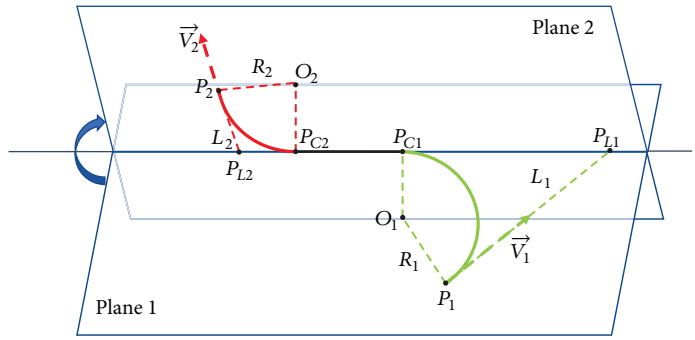



Figure 1: A typical CSC curve starts from $P_1$ with $\vec{V}_1$ and ends at $P_2$ with $\vec{V}_2$.

Table 1: Mathematical function definition.

| Function | Expression |
|---|---|
| $\mathbf{s}(v)$ | calculate the sign of the variable $v$ |
| $\Theta(\vec{a}, \vec{b})$ | calculate the angle between $\vec{a}$ and $\vec{b}$ |
| $\mathbf{n}(\vec{S})$ | normalizes $\vec{S}$ |

The docking station considered in this approach is assumed to be static and its configuration is assumed to be known by the robot beforehand. Therefore, the configuration of the docking station is expressed as $\mathscr{C}_d$ and

$$\mathscr{C}_d = \left\{P_d, \vec{V}_d\right\}, \tag{2}$$

where $P_d$ is its position and $\vec{V}_d$ denotes its designed entrance direction.

Irregular obstacles are considered in this paper and they are assumed to have been decomposed into several spherical obstacles already. This decomposition technique can be found in [29]. Hence, for an obstacle $i$, its center is written as $O_i$, its radius is written as $R_i$, and its velocity is written as $\vec{V}_i$. The obstacle area $\Gamma(O_i)$ of obstacle $i$ can be described as

$$\Gamma\left(O_i\right) = \left\{P\left(x, y, z\right) \mid \left\|P - O_i\right\| \le R_i\right\}. \tag{3}$$

For a feasible path $\mathbf{L}$ for robot docking, for any obstacle $i$, there are

$$\mathbf{L} \notin \Gamma\left(O_i\left(t\right)\right) \tag{4}$$

and

$$\mathscr{C}_r\left(T_f\right) = \mathscr{C}_d, \tag{5}$$

where $T_f$ is the elapsed time during which the robot reaches the docking station.

To simplify the expression, several functions are defined as Table 1 presents.

## 3. Design of Dubins Tree

In this section, we present the notation and techniques about the design of Dubins tree in our path planning approach. The collision prediction in our approach is achieved based on the characters of 3D Dubins curves as well. The main results of this section provide the foundations for the proposed path planning approach.

*3.1. 3D Dubins Curves.* Dubins curve is proved to be the optimal trajectory which connects two configurations with bounded curvature in 2D space [30, 31]. Dubins curves are simply composed of simple segments (straight line segments (S) and circular arcs (C)) and smooth enough for robots to follow. These curves are also attractive in the studies of 3D path planning for nonholonomic robots because of their simpleness and smoothness, even if their optimality is not ensured in this situation [32–35]. However, as the dimension increases compared to 2D environment, the calculation consumption of the Dubins curves increases as well. It is disadvantageous especially when plenty of 3D Dubins curves are required, i.e., in the applications of searching in the environment with numerous obstacles. Rather than using modified Dubins curves, i.e., in [32–34], we attempt to overcome this disadvantage through determining 3D Dubins curves based on their geometrical characters [36].

A typical form of 3D Dubins curves is named the CSC curve, which is presented in Figure 1. The CSC curve consists of two circular segments, $C_1$ (green circular arc) and $C_2$ (red circular arc), and one straight line segment $\overline{P_{C1}P_{C2}}$ (black straight line). Its initial configuration is written as $\mathscr{C}_1 = [P_1, \vec{V}_1]$ (green arrow) and its final configuration is written as $\mathscr{C}_2 = [P_2, \vec{V}_2]$ (red arrow). The terminal configurations $\mathscr{C}_1$ and $\mathscr{C}_2$ are on two different planes which are named plane 1 and plane 2 as Figure 1 shows.

To outline the geometric characters of the CSC curve, three auxiliary straight lines are adopted as shown in Figure 1, which are $L_1$, $L_2$, and $L_i$. $L_1$ and $L_2$ are the colinear lines with $\mathscr{C}_1$ and $\mathscr{C}_2$, respectively. $L_i$ is the intersection line of plane 1 and plane 2. In this case, it can be seen from Figure 1 that plane 1 can be determined by $L_1$ and $L_i$. Likewise, plane 2 can be determined by $L_i$ and $L_2$. Furthermore, since the circular segments $C_1$ and $C_1$ are on plane 1 and plane 2 correspondingly and intersect with $L_i$ at $P_{C1}$ and $P_{C2}$ separately, once $L_1$, $L_2$, and $L_i$ are determined, all the segments of the CSC curve are determined as well. On this basis, the determination approach of the CSC curve is presented as follows.

Assuming that the intersection points of $L_i$ with $L_1$ and $L_2$ are written as $P_{L1}$ and $P_{L2}$, we have

$$\begin{aligned} P_{L1} &= P_1 + k_3 \vec{V}_1, \\ P_{L2} &= P_2 + k_4 \vec{V}_2, \end{aligned} \tag{6}$$

where $k_3$ and $k_4$ are nonzero constants. Because $P_{C1}$ and $P_{C2}$ are on $L_i$, they can be obtained as

$$\begin{aligned} P_{C1} &= P_{L1} + k_5 \vec{V}_i, \\ P_{C2} &= P_{L2} + k_6 \vec{V}_i, \end{aligned} \tag{7}$$

where $k_5$ and $k_6$ are constants. $\vec{V}_i$ is the auxiliary vector along with the curves propagation direction.

According to the spatial-temporal relations of the CSC curve, we have

$$\begin{aligned} \vec{V}_i &= \mathbf{s}(k_3)\,\mathbf{n}\left(\overrightarrow{O_1P_1} \times \overrightarrow{O_1P_{C1}} \times \overrightarrow{O_1P_{C1}}\right) \\ &= -\mathbf{s}(k_4)\,\mathbf{n}\left(\overrightarrow{O_2P_{C2}} \times \overrightarrow{O_2P_2} \times \overrightarrow{O_2P_{C2}}\right). \end{aligned} \tag{8}$$

Meanwhile, $P_{C1}$ and $P_{C2}$ are also the tangent points of $C_1$ and $C_2$ with the S segment; then

$$\begin{aligned} \overrightarrow{P_{C1}O_1} \times \vec{V}_i &= 0, \\ \overrightarrow{P_1O_1} \times \vec{V}_1 &= 0, \\ \overrightarrow{P_{C2}O_2} \times \vec{V}_i &= 0, \\ \overrightarrow{P_2O_2} \times \vec{V}_2 &= 0, \end{aligned} \tag{9}$$

and

$$\begin{aligned} \overline{P_{C1}O_1} &= \overline{P_1O_1} = R_1, \\ \overline{P_{C2}O_2} &= \overline{P_2O_2} = R_2. \end{aligned} \tag{10}$$

It is derived from (9) and (10) that $O_1$ and $O_2$ are on the angle bisectors of $\angle P_{C1}P_{L1}P_1$ and $\angle P_{C2}P_{L2}P_2$, which are described as

$$\begin{aligned} O_1 &= P_{L1} + k_1 \mathbf{s}(k_3)\left(-\vec{V}_1 + \vec{V}_i\right), \\ O_2 &= P_{L2} + k_2 \mathbf{s}(k_4)\left(-\vec{V}_2 + \vec{V}_i\right). \end{aligned} \tag{11}$$

Nonlinear equations to solve $k_1$ to $k_6$ can be derived from (9) and (10). The parameters of the CSC curve can be determined exactly by solving these nonlinear equations. The length of the Dubins curve can be calculated by adding up its segments, which can be written as $\|\widehat{P_1P_{C1}}\| + \|P_{C1}P_{C2}\| + \|\widehat{P_{C2}P_2}\|$. Similarly, the other Dubins curves can be determined via the equations described above with slight modifications.

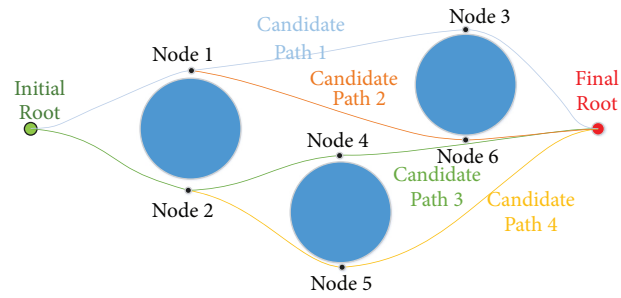



Figure 2: Outline of the Dubins tree. In this figure, the Dubins tree has 2 roots, 6 nodes, and 10 branches.

Table 2: Dubins tree symbol definitions.

| Name | Definition | Expression |
|---|---|---|
| **node** | the intermediate configuration | $\mathscr{C}_i$ |
| **branches** | the Dubins curve that connects $\mathscr{C}_i$ and $\mathscr{C}_j$ | $\mathbf{A}(i, j)$ |
| **roots** | the initial and terminal nodes | $\mathscr{C}_0, \mathscr{C}_f$ |

*3.2. Dubins Tree.* Based on the determination method of 3D Dubins curves, we present the **Dubins tree** as the foundation of path planning for robot docking in obstacle environment. The notion of the Dubins tree is illustrated in Figure 2. A Dubins tree consists of three elements which are the nodes, the branches, and the roots, as the example illustrated in Table 2.

Hence, the path planning task for robot docking based on the Dubins tree is equal to determine a set of branches that connect each other from $\mathscr{C}_0$ to $\mathscr{C}_f$ which are collision-free and shortest. The collision-free set of branches that connects $\mathscr{C}_0$ and $\mathscr{C}_f$ is defined as the **candidate path** $\mathscr{L}_k(0, f)$, which can be written as

$$\mathscr{L}_k(0, f) = \{\mathbf{A}(0, 1), \ldots, \mathbf{A}(i, j), \ldots, \mathbf{A}(h, f)\}, \tag{12}$$

where all the branches are collision-free. In this case, there are

$$P_r \in \mathbf{A}(i, j) \tag{13}$$

and

$$P_r \notin \{\Gamma(O_m(t)) \mid m = 1, \ldots, N_o\} \tag{14}$$

for $t \in [T_i, T_j]$, where $N_o$ is the number of obstacles. For instance, in Figure 2, the candidate path $\mathscr{L}_1 = \{\mathbf{A}(0, 1), \mathbf{A}(1, 3), \mathbf{A}(3, f)\}$ (light blue line) and it is a collision-free candidate path if only stationary obstacles exist in the environment.

*3.3. Path Planning Based on Dubins Tree.* The Dubins tree $\mathbf{G}(0, f)$ consists of several candidate paths, which can be described as

$$\mathbf{G}(0, f) = \sum_{k=1}^{N_{\mathscr{L}}} \mathscr{L}_k(0, f), \tag{15}$$

where $N_{\mathscr{L}}$ is the number of the candidate paths.

The scheme of the path planning approach based on Dubins tree is presented as follows. Once the planned path **L** is infeasible, a Dubins tree $\mathbf{G}(0, f)$ is planted and expanded from the initial root to the final root while avoiding obstacles. Sufficient nodes and branches are generated to spread the branches of the Dubins tree through the collision-free space, namely, the tree expansion. They are generated under the guidance of an optimized tree expanding strategy, considering the constraints and requirements of the task. Then, candidate paths are formed by these nodes and branches that connect to the docking station. After the tree expanding process is completed, the shortest candidate path in $\mathbf{G}(0, f)$ is selected as the resultant path **L**.

## 4. Smooth Path Planning for Docking in Unknown Environment

In this section, we present the design and implementation of the smooth path planning approach for docking in unknown environment.

*4.1. Collision Prediction.* Collision prediction is a key issue for robot path planning in the practical applications especially when the prior information is not sufficiently provided, i.e., in the unknown environment [37–40]. Since the parameters of the branches in our approach can be obtained via the techniques provided in Section 3, inspired by this notion, this issue is addressed based on the characters of Dubins curves [36].

Because the candidate paths only consist of C segments and S segments, based on the feasibility evaluation of these segments, the collision prediction process can be implemented efficiently. The descriptions of the planned trajectories are presented as follows. The planned trajectory of the S segment can be described as

$$P_r(t) = P_1 + t\overrightarrow{V}_1, \tag{16}$$

where $P_1$ represents the start point of $S$ segment and $\overrightarrow{V}_1$ represents the planned velocity of the robot at $P_1$.

There are three types of the C segments with prescribed radii $R$ which are the minor C curve, the semicircular C curve, and the major C curve. Hence, the last two types are divided into several minor subsegments for calculation efficiency as Figure 3 shows. For the minor C segment case $\widehat{P_1P_2^m}$, $P_r(t)$ can be expressed as

$$Pr(t) = O + R\left(\sin\frac{vt}{R}\overrightarrow{V}_x + \cos\frac{vt}{R}\overrightarrow{V}_y\right), \tag{17}$$

where $t \in [0, R\Theta(\overrightarrow{OP_1}, \overrightarrow{OP_2^m})/v]$, $\overrightarrow{V}_x = \mathbf{n}(\overrightarrow{OP_1})$, and $\overrightarrow{V}_2 = \mathbf{n}(\overrightarrow{V}_1)$.

For the semicircular C segment case $\widehat{P_1P_2^s}$, it is divided into two minor C subsegments which are $\widehat{P_1P_2^m}$ and $\widehat{P_2^mP_2^s}$. The division point $P_2^m$ can be chosen as

$$P_2^m = O + R\mathbf{n}\left(\overrightarrow{V}_1\right). \tag{18}$$

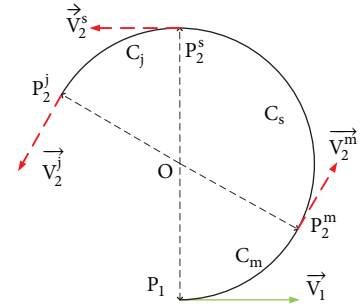



Figure 3: Divisions of the C segments. The minor C segment case is $\widehat{P_1P_2^m}$, the semicircular C segment is $\widehat{P_1P_2^s}$, and the major C segment is $\widehat{P_1P_2^j}$.

Thus, the trajectories of these minor C subsegments $\widehat{P_1P_2^m}$ and $\widehat{P_2^mP_2^s}$ can be expressed according to (17) correspondingly.

For the major C segment case $\widehat{P_1P_2^j}$, it is divided into three minor subsegments which are $\widehat{P_1P_2^m}$, $\widehat{P_2^mP_2^s}$, and $\widehat{P_2^sP_2^j}$. In this case, the division points $P_2^m$ and $P_2^s$ are chosen as

$$\begin{aligned} P_2^m &= O + R\mathbf{n}\left(\overrightarrow{P_2^jO}\right), \\ P_2^s &= O + R\mathbf{n}\left(\overrightarrow{P_1O}\right). \end{aligned} \tag{19}$$

$\widehat{P_1P_2^m}$, $\widehat{P_2^mP_2^s}$, and $\widehat{P_2^sP_2^j}$ can be expressed according to (17)-(19) correspondingly.

In this case, the description of certain trajectory can be derived from the combination of these basic subsegments easily. For instance, the planned trajectory of the CSC curve, as Figure 1 shows, is written as

$$P_r(t) = \begin{cases} C_1(t), & t \in [0, T_1) \\ S(t - T_1), & t \in [T_1, T_2) \\ C_2(t - T_2), & t \in [T_2, T_3) \end{cases} \tag{20}$$

where $C_1(t)$, $S(t - T_1)$, and $C_2(t - T_2)$ represent the trajectories of $C_1$, $S$, and $C_2$, respectively. Meanwhile, $T_1 = (R/v)\Theta(\overrightarrow{O_1P_1}, \overrightarrow{O_1P_{C1}})$, $T_2 = T_1 + \|\overrightarrow{P_{C1}P_{C2}}\|/v$, and $T_3 = T_2 + (R/v)\Theta(\overrightarrow{O_2P_{C2}}, \overrightarrow{O_2P_2})$.

Since the candidate paths consist of several Dubins curves, their planned trajectories can be determined based on the combination of these curves in spatial-temporal order, as well as their feasibilities. In this paper, the movements of the obstacles are assumed to be predictable; the candidate path $\mathscr{L}_k$ is collision-free if (13) and (14) hold for all its branches.

If $\mathscr{L}_k$ is not collision-free, we define the obstacle $i$ that the robot would collide firstly as the **maximal collision obstacle**, which is written as $Oc$. The position of the robot when the distance between $P_r(t)$ and $Oc$ will be minimal is defined as the **maximal collision point** $Pc$. The moment maximal collision would take place is defined as the **maximal collision**

**time** $Tc$. In this case, if there are multiple obstacles in the area, $Tc$ and $Pc$ can be obtained as

$$\begin{aligned} Tc &= \min\{Tc_i\}, \\ Pc &= P_r(Tc), \end{aligned} \tag{21}$$

where $i = 1, 2, \ldots, N_o$. Additionally, the **maximal collision pose** $\overrightarrow{Vc}$ is defined as

$$\overrightarrow{Vc} = \overrightarrow{V_r}(Tc). \tag{22}$$

*4.2. Heuristic Tree Expanding Strategy.* To improve the performance of the proposed approach, the elastic band theory is integrated into the tree expansion strategy which is named the Heuristic Tree Expanding (HTE) strategy [41, 42]. The HTE strategy assumes that there exist interactive forces between each neighbor node connected with one branch as if they were connected by the potential spring. Hence, there are potential spring forces along the branches. In addition, the obstacles are assumed to be able to provide sufficient support forces for these nodes when the nodes contact their surfaces. Hence, according to the elastic band theory, although various nodes are generated in the attempt of expanding, only these nodes whose total forces can be balanced remain [43]. This strategy ensures that not only the collision avoidance but also the optimality of expanding branches is considered in the determination of nodes.

The design of the HTE strategy is presented as follows. For the infeasible path $\mathbf{L}$, its maximal collision obstacle $Oc$, point $Pc$, time $Tc$, and pose $\overrightarrow{Vc}$ can be acquired as Section 4.1 presents. According to the state of $Oc$, two situations are considered in the determination of nodes for Dubins tree expansion. If $Oc$ is a static obstacle, this situation is called the static collision situation. If $Oc$ is a moving obstacle, this situation is called the moving collision situation. The implementation of the HTE strategy is slightly different in the two situations, which are provided as follows.

*4.2.1. Static Collision Situation.* In the static collision situation, the obstacle avoidance is not urgent because the configuration of the obstacle environment is stable. Hence, the optimality of the resultant path is considered firstly in the tree expanding process. In this case, the nodes for tree expanding are generated on the surfaces of all the static obstacles, which provides the potential for the resultant branches to form collision-free and short candidate paths.

For a known static obstacle $i$, two nodes are determined which are $\mathscr{C}_{ia} = [O_i + R_i\overrightarrow{N}, \overrightarrow{Vc}]$ and $\mathscr{C}_{ib} = [O_i - R_i\overrightarrow{N}, \overrightarrow{Vc}]$, where

$$\overrightarrow{N} = \begin{cases} \mathbf{n}\left(\overrightarrow{OcPc} \times \overrightarrow{Vc} \times \overrightarrow{Vc}\right), & \text{if } \overrightarrow{OcPc} \times \overrightarrow{Vc} \neq \mathbf{0}, \\ \mathbf{n}\left(\overrightarrow{S}\right), & \text{if } \overrightarrow{OcPc} \times \overrightarrow{Vc} = \mathbf{0}, \end{cases} \tag{23}$$

and $\overrightarrow{S}$ is a nonzero random 3D vector.

Furthermore, three angles $\angle P_r P_{ik} O_i$, $\angle O_i P_{ik} P_d$, and $\angle P_r P_{ik} P_d$, $k \in \{a, b\}$, are measured to evaluate the feasibilities of these nodes. They are utilized to simulate the potential spring forces of $\mathscr{C}_{ik}$ and then the feasibility of this node is determined based on the following inequalities.

$$\begin{aligned} \angle P_r P_{ik} O_i &> \frac{\pi}{2}, \\ \angle O_i P_{ik} P_d &> \frac{\pi}{2}, \end{aligned} \tag{24}$$

and

$$\begin{aligned} \angle P_r P_{ik} O_i &> \angle P_r P_{ik} P_d, \\ \angle O_i P_{ik} P_d &> \angle P_r P_{ik} P_d. \end{aligned} \tag{25}$$

$\mathscr{C}_{ik}$ is infeasible and excluded from the Dubins tree if one or more of these inequalities hold. It is because, for $\mathscr{C}_{ik}$, its total force cannot be unbalanced in this situation if the branches are generated to connect it and tightened up. Otherwise, $\mathscr{C}_{ik}$ is accepted and added to the Dubins tree.

An example of the node selection in static collision situation is illustrated in Figure 4. In this example, the initial path (black solid line) is infeasible and four nodes, $\mathscr{C}_{1a}$, $\mathscr{C}_{1b}$ and $\mathscr{C}_{ia}$, $\mathscr{C}_{ib}$, are determined according to the HTE strategy. The potential spring forces of the nodes are along the dash lines towards $\mathscr{C}_0$ and $\mathscr{C}_f$. Besides, the total potential spring forces of these nodes are indicated by the red arrows. In this case, $\mathscr{C}_{ia}$ is evaluated to be infeasible according to the HTE strategy and excluded from the tree expanding. The other three nodes are evaluated to be feasible for branches generation so that $\mathbf{A}(0, 1a)$ (black solid curve) is accepted as one of the generated branches rather than $\mathbf{A}(0, 2a)$.

*4.2.2. Moving Collision Situation.* In the moving collision situation, the collision avoidance is considered previously for the sake of safety due to the configuration changes of the obstacle environment. Thus, feasible node is solely selected on the maximal collision obstacle in this case to avoid the immediate danger. For a moving maximal collision obstacle $i$, the node $\mathscr{C}_i$ is determined as $\mathscr{C}_i = [Oc + Rc\overrightarrow{N}, \overrightarrow{Vc}]$, where $\overrightarrow{N}$ is a certain radial direction of the obstacle $i$. The velocity of $i$ is also considered in the determination of $\overrightarrow{N}$ and $\mathscr{C}_i$ for the efficiency of the obstacle-avoiding maneuver, which is inspired by the implement of the velocity obstacle method in [44]. $\overrightarrow{N}$ is obtained as

$$\overrightarrow{N} = \begin{cases} -\mathbf{g}\left(\overrightarrow{Vc} \times \overrightarrow{V_{Oc}} \times \overrightarrow{Vc}\right), & \text{if } \overrightarrow{V_{Oc}} \times \overrightarrow{Vc} \neq \mathbf{0} \\ \mathbf{g}\left(\overrightarrow{S}\right), & \text{if } \overrightarrow{V_{Oc}} \times \overrightarrow{Vc} = \mathbf{0}, \end{cases} \tag{26}$$

where $\overrightarrow{V_{Oc}}$ is the velocity of $Oc$.

An implementation example of the HTE strategy in moving obstacle situation is presented in Figure 5, where the original path (black solid line) is infeasible and $\mathscr{C}_1$ is the determined node for branch generation.

After the determination of nodes for tree expanding, new candidate paths are formed via these generated nodes to the docking station, which is expressed as

$$\mathscr{L}_{new}(0, f) = \mathscr{L}_{new}(0, i) + \mathbf{A}(i, f), \tag{27}$$

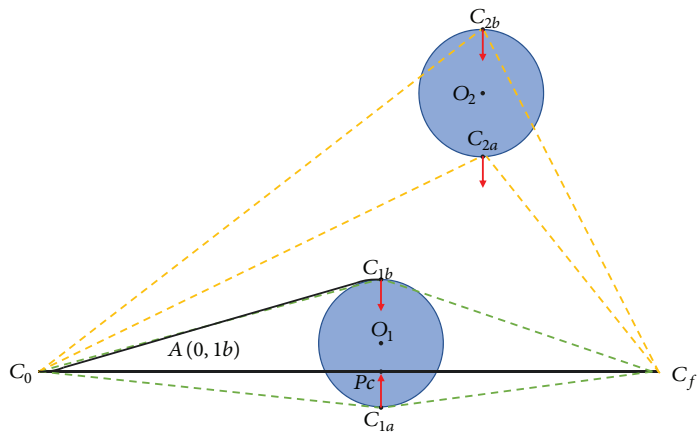



FIGURE 4: The implementation example of the HTE strategy in static collision situation.

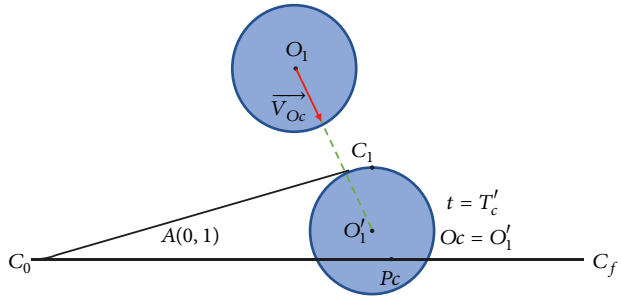



FIGURE 5: The implementation example of the HTE strategy in moving obstacle situation.

where

$$\mathscr{L}_{new}(0, i) = \mathscr{L}_{old}(0, m) + \mathbf{A}(m, i). \tag{28}$$

Besides, $\mathscr{L}_{new}$ represents the new candidate path and $\mathscr{L}_{old}$ represents the old infeasible path. These new candidate paths will be evaluated according to the HTE strategy in the further tree expansion.

Additionally, to improve the performance of the proposed path planning approach in complex unknown environment, a searching acceleration technique is designed based on the Dubins tree structure and applied in the tree expanding process. Considering that the candidate paths can only grow longer in the expanding process, Proposition 1 is derived for the comparison between two candidate paths $\mathscr{L}_i(0, x)$ and $\mathscr{L}_j(0, y)$ with a common node $\mathscr{C}_0$.

**Proposition 1.** *If* $\|\mathscr{L}_i(0, x)\| \geq \|\mathscr{L}_j(0, y)\|$ *and* $\mathscr{L}_k(0, y) = \mathscr{L}_i(0, x) + \mathbf{A}(x, y)$, *then* $\|\mathscr{L}_k(0, y)\| > \|\mathscr{L}_j(0, y)\|$.

Proposition 1 evaluates the potential of the branches to form the feasible path in future and abandons the undesirable nodes. It will obviously reduce the computation burden and enhance the quality of generated branches in the obstacle environment.

Consider that, in the environment crowded with obstacles, there will be superabundant nodes generated for processing which will take excessive computation time in the implementation of path planning.

To enhance the efficiency of our path planning approach, both the time constraint in practical applications and the sufficiency of the environment exploration are considered in the HTE strategy.

The **spreading degree** of a node $\mathscr{C}_i$ is defined as

$$\mathbf{K}(\mathscr{C}_i) = -\|P_i - Oc\|, \tag{29}$$

where $P_i$ is the position of the node $\mathscr{C}_i$ and $i$ is the index of this node. $\mathbf{K}$ indicates the compactness of the generated nodes and the larger $\mathbf{K}$ is, the more compact the node is.

To reduce the computation consumption in tree expansion, the notion **maximal spreading ratio** $S_{\max}$ is employed in the HTE strategy to control the number of the generated nodes for efficiency. It indicates the maximal number of the generated nodes $\{\mathscr{C}_i\}$ in one tree expanding process, where

$$\mathscr{C}_i = \{\mathscr{C}_k \mid \mathbf{K}(\mathscr{C}_k) \geq \mathbf{K}_{max} \text{ and } k = 1, 2, \ldots, S_{\max}\}, \tag{30}$$

and $\mathbf{K}_{max}$ is the maximal spreading degree.

*4.3. Smooth Path Planning Algorithm.* Based on the Dubins tree, the smooth path planning approach for robot docking is implemented under the guidance of the HTE strategy, as presented below. For the simplification of expression, several useful variables are defined firstly. The unchecked path set is defined as $\mathbf{H}_u$ and the candidate path set is defined as $\mathbf{H}_f$. Hence, the smooth path planning approach is implemented in four procedures which are illustrated as follows.

*4.3.1. Activation.* The proposed approach is activated when the configuration of the obstacle environment changes, which is described as

$$\{\mathcal{O}'\} \neq \{\mathcal{O}\}, \tag{31}$$

where $\{\mathcal{O}'\}$ represents the present configuration and $\{\mathcal{O}\}$ represents the known configuration. This change is probably

**Input:** $\mathscr{C}_r, \mathscr{C}_d, R_{min}, \overrightarrow{V}_r, R_s$
**Output:** **L**
(1) $\mathscr{C}_0 \longleftarrow \mathscr{C}_r$
(2) **L** $\longleftarrow$ **T**$(\mathscr{C}_0, \mathscr{C}_d, R_{min}, R_{min})$
(3) **while** $\mathscr{C}_r \neq \mathscr{C}_d$ **do**
(4) **if** $\{\mathscr{O}'\} \neq \{\mathscr{O}\}$ **then**
(5) $\{\mathscr{O}\} \longleftarrow \{\mathscr{O}\} \cup \{\mathscr{O}'\}$
(6) Push **L** into $\mathbf{H}_u$, set $\|\mathbf{L}\| = +\infty$, $Nd = 0$
(7) **while** $\mathbf{H}_u \neq \emptyset$ **do**
(8) Pop $\mathscr{L}_k$ from $\mathbf{H}_u$
(9) **if** $\mathbf{f}(\mathscr{L}_k) = 0$ **then**
(10) $[Oc, Pc, \overrightarrow{Vc}] \longleftarrow$ **CP**$(\mathscr{L}_k, \{\mathscr{O}\})$
(11) $\{\mathscr{C}_i, \mathbf{A}(s,i)\} \longleftarrow$ **TE**$([Oc, Pc, \overrightarrow{Vc}], \mathscr{O}, S_{\max})$
(12) $\{\mathscr{L}_i\} \longleftarrow \{\mathscr{L}_k + \mathbf{A}(\mathscr{C}_i, \mathscr{C}_d)\}$
(13) Push $\{\mathscr{L}_i\}$ into $\mathbf{H}_u$
(14) **else**
(15) Push $\{\mathscr{L}_k\}$ into $\mathbf{H}_f$
(16) **if** $\|\mathbf{L}\| < \|\mathscr{L}_k\|$ **then**
(17) **L** $\longleftarrow \mathscr{L}_k$
(18) **end if**
(19) **end if**
(20) **if** $\mathbf{H}_f \neq \emptyset \wedge \{\mathbf{H}_u = \emptyset \vee$ **TB**$(\mathbf{G}(0, f)) > b_{max}\}$ **then**
(21) **break**
(22) **end if**
(23) **end while**
(24) **else**
(25) follow **L**
(26) **end if**
(27) **end while**

Algorithm 1: The smooth path planning approach.

caused by the detection of new obstacles in path following or the configuration change of certain obstacle. In this case, the planned path **L** is added into $\mathbf{H}_u$ for further evaluation.

*4.3.2. Path Evaluation.* The feasibility of each planned path in $\mathbf{H}_u$ is evaluated firstly according to the equations in Section 4.1. If the planned path, i.e., $\mathscr{L}_k$, is feasible, it is added into $\mathbf{H}_f$ for the selection. Otherwise, it will be handled in the path generation procedure for path replanning. Then $\mathscr{L}_k$ is popped out from $\mathbf{H}_u$.

*4.3.3. Path Generation.* For the infeasible $\mathscr{L}_k$, the HTE strategy is adopted to generate the nodes and branches for the expansion of $\mathbf{G}(0, f)$. Firstly, a level of nodes and branches are determined and connected for collision avoidance. Then branches that connect the nodes to the docking station are generated, as (27) represents. They provide the new planned paths for further evaluation which are saved into $\mathbf{H}_u$. The path evaluation and path generation procedures will be repeated iteratively till the termination condition is satisfied.

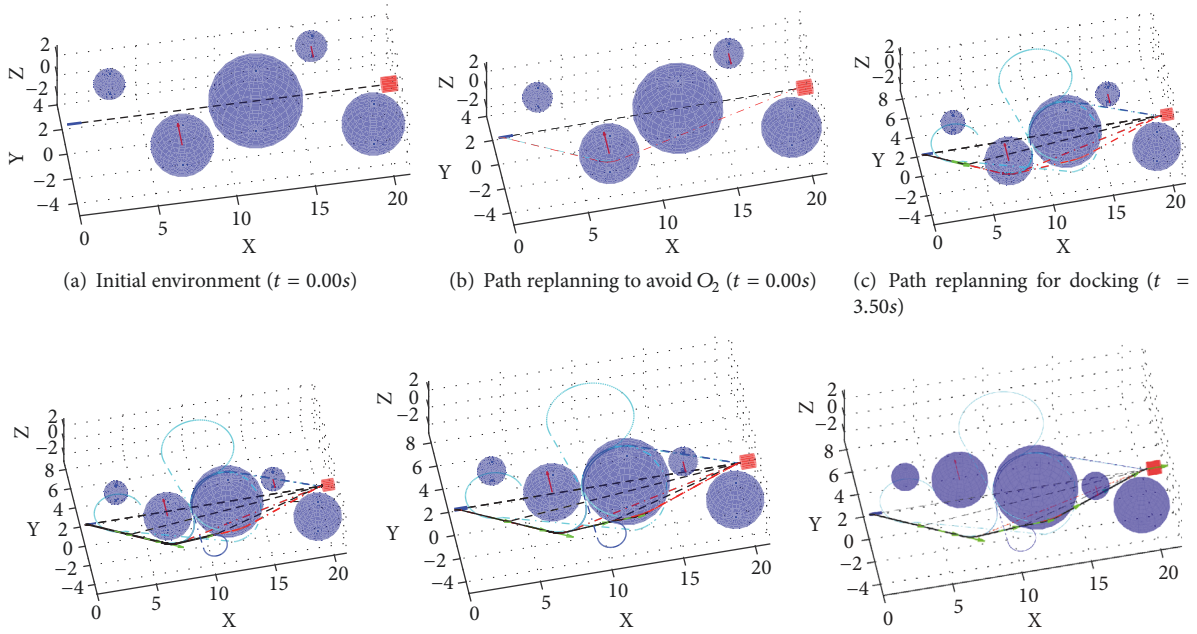
*4.3.4. Termination.* If $\mathbf{H}_f \neq \emptyset$, two termination situations are considered, namely, the mature situation and the immature situation. The mature situation represents the fact that all the possible candidate paths are obtained, which is indicated by $\mathbf{H}_u = \emptyset$. In this situation, the Dubins tree has been sufficiently expanded and its shortest candidate path in $\mathbf{H}_f$ is chosen as the resultant path. In the immature situation, plenty of candidate paths have been determined; however, $\mathbf{H}_u \neq \emptyset$. In this case, the optimality and computation consumption should be balanced to ensure the efficiency. The termination condition in the immature situation is described as

$$N_b \geq N_o \left( S_{\max}^{N_d} + \frac{b_j}{S_{\max}} + N_o \right),$$
$$N_{\mathscr{L}} \geq \frac{N_o}{j} + 0.5, \tag{32}$$

where $N_b$ is the number of whole generated branches; $N_{\mathscr{L}}$ is the number of the candidate paths in $\mathbf{H}_f$. Meanwhile, $b_j$ is the number of the new generated branches at the $j$th path generation execution, $N_o$ is the number of detected obstacles, and $N_d$ is the depth of $\mathbf{G}(0, f)$.

If one or more equations in (32) hold, which indicates either plenty of branches or candidate paths are generated, the path planning approach terminates and the shortest candidate path is chosen as the resultant path.

The algorithm of the prescribed smooth path planning approach is presented as Algorithm 1, where the definitions of the functions are given in Table 3.



(a) Initial environment ($t = 0.00s$)
(b) Path replanning to avoid $O_2$ ($t = 0.00s$)
(c) Path replanning for docking ($t = 3.50s$)
(d) Path replanning for docking ($t = 11.00s$)
(e) Path replanning for docking ($t = 13.70s$)
(f) Docking successfully ($t = 21.38s$)

Figure 6: Simulation results of the smooth path planning approach in unknown environment with both static and moving obstacles.

Table 3: Function definition in Algorithm 1.

| Function | Expression |
|---|---|
| $\mathbf{T}(\mathscr{C}_i, \mathscr{C}_j, R_i, R_j)$ | determine $\mathbf{A}(i, j)$ with $R_i$ and $R_j$ |
| $\mathbf{f}(\mathbf{L})$ | true if $\mathbf{L}$ is infeasible and vise versa |
| $\mathbf{CP}(\mathbf{L}, \{\mathscr{O}\})$ | collision prediction procedure of $\mathbf{L}$ |
| $\mathbf{TE}([Oc, Pc, \overrightarrow{Vc}], \mathscr{O}, S_{\max})$ | tree expanding procedure |
| $\mathbf{TB}(\mathbf{G}(0, f))$ | returns generated branches' number |

*4.4. Case Study.* A case study is provided in this section to illustrate the workflow of the prescribed smooth path planning algorithm. Parameters of the robot in this case are set as

$$\begin{aligned} v &= 1m/s, \\ R_s &= 6m, \\ R_{min} &= 1m. \end{aligned} \tag{33}$$

The configurations of the obstacles in this case are set as

$$\begin{aligned} P_r(0) &= (0,0,0)^{\mathrm{T}}, \\ \overrightarrow{V}_r(0) &= (1,0,0)^{\mathrm{T}}, \\ P_d &= (20,0,0)^{\mathrm{T}}, \\ \overrightarrow{V}_d &= (1,0,0)^{\mathrm{T}}, \\ O_1 &= (3,2,1)^{\mathrm{T}}, \\ \overrightarrow{V}_1 &= (0,0,0)^{\mathrm{T}}, \\ R_1 &= 1m, \\ O_2 &= (7,-2,-1)^{\mathrm{T}}, \\ \overrightarrow{V}_2 &= (0,0,0.2)^{\mathrm{T}}, \\ R_2 &= 2m, \\ O_3 &= (12,0,0)^{\mathrm{T}}, \\ \overrightarrow{V}_3 &= (0,0,0)^{\mathrm{T}}, \\ R_3 &= 3m, \\ O_4 &= (16,3,1)^{\mathrm{T}}, \\ \overrightarrow{V}_4 &= (0,-0.2,0)^{\mathrm{T}}, \\ R_4 &= 1m, \\ O_5 &= (19,-3,0)^{\mathrm{T}}, \\ \overrightarrow{V}_5 &= (0,0,0)^{\mathrm{T}}, \\ R_5 &= 2m. \end{aligned} \tag{34}$$

The implementation of Algorithm 1 in an unknown environment with both static and moving obstacles is illustrated in Figure 6, where the simulation step length is set as $0.5s$ and the parameter of Algorithm 1 is set as $S_{max} = 2$. In these figures, the planned paths, the candidate paths, and generated branches are indicated by the black dash lines, the red dash

**Input:** Root $\mathcal{N}_0$, Goal $\mathcal{N}_g$, $r$, Iterations $I_n$, $\epsilon$
**output:** **L**
(1) **for** $i = 1$ to $I_n$ **do**
(2) Generate random sample nodes $\mathcal{N}_r$ within $\epsilon$ to $\mathcal{N}_i$
(3) Determine the nearest existent $\mathcal{N}_j$ to $\mathcal{N}_i$ within $r$
(4) **if** the branch $\mathcal{N}_j\mathcal{N}_i$ is collision-free **then**
(5) Determine $\mathcal{N}_k$ with lowest cost from $\mathcal{N}_0$ to $\mathcal{N}_i$
(6) Set $\mathcal{N}_k$ as the parent node of $\mathcal{N}_i$
(7) **end if**
(8) **end for**
(9) the shortest path $\mathcal{N}_0\mathcal{N}_g \longrightarrow$ **L**

Algorithm 2: The RRT∗ path planning approach.

lines, and cyan dash lines, respectively. The docking station is presented by the surface of a red cylinder. Meanwhile, the motion directions of the moving obstacles are indicated by the red arrows. Black solid lines and the green arrows are used to present the path and the vector of the robot separately.

The initial configuration of the workspace is illustrated in Figure 6(a), where the initial path is not collision-free for the robot docking. Hence, the smooth path planning algorithm is utilized to generate the maneuver of avoiding the moving obstacle $O_2$, which is presented in Figure 6(b). As Figure 6(c) shows, new static obstacle ($O_3$) is detected in the collision avoidance maneuvers and the planned path is blocked by $O_3$. In this case, new path for docking is determined by the proposed path planning algorithm and followed by the robot. The static obstacle $O_5$ and moving obstacle $O_4$ are detected in the path following process chronologically, as Figures 6(d) and 6(e) illustrate, respectively. Because they are evaluated to be collision-free by the proposed path planning algorithm, the planned path navigates the robot to dock successfully. In the path planning process, 24 Dubins curves are generated to determine the feasible path for docking, whose length is $21.38m$.

## 5. Simulations and Discussion

To evaluate the validity and feasibility of the proposed path planning approach, several practical scenarios are presented and discussed in this section. Meanwhile, the performance of the prescribed algorithm is also discussed via the comparisons with the other three path planning approaches, which are the APF approach, the rapidly exploring random tree star (RRT∗) approach, and the DAPF approach we proposed in our previous work [36].

The parameters of the robot are the same as (33) in Section 4.4. In addition, the initial configurations of the robot and the docking station in these simulation scenarios are set as

$$P_r(0) = [0,0,0]^{\mathrm{T}},$$
$$\vec{V}_r(0) = [1,0,0]^{\mathrm{T}},$$
$$P_d = [20,0,0]^{\mathrm{T}},$$
$$\vec{V}_d = [1,0,0]^{\mathrm{T}}, \tag{35}$$

and the step length in the simulations is set as $0.5s$.

*(1) APF Approach.* The APF approach is well known for its efficiency in path planning for collision avoidance. In this paper, the potential functions of APF approach are designed as

$$U_{rep}\left(\overrightarrow{P_rO_i}\right) = \frac{k_p}{2}\left(\frac{1}{\left\|\overrightarrow{P_rO_i}\right\| - R_i} - \frac{1}{\left\|\overrightarrow{P_rP_d}\right\|}\right)^2,$$
$$U_{att}\left(\overrightarrow{P_rO_i}\right) = \frac{k_a}{2}\left(\left\|\overrightarrow{P_r - P_d}\right\|\right)^2, \tag{36}$$

where $U_{rep}$ and $U_{att}$ are the repulsive and attractive potential functions and $\|O_i - P_r\| \leq R_s$. Besides, $k_a$ is the parameter to adjust the strength of the attractive potential field and $k_r$ is the parameter to adjust the strength of the repulsive potential field. The parameters of the potential functions in the following simulation scenarios are set as $k_a = 1, k_p = 5$.

*(2) RRT∗ Approach.* The proposed path planning algorithm is also compared to RRT∗ approach. The algorithm of the RRT∗ approach in obstacle environment is presented as Algorithm 2.

The parameters of this algorithm in the following simulation scenarios are set as $r = 6, \epsilon = 0.5, I_n = 2000$.

*(3) DAPF Approach.* The DAPF approach employs a reactive path planner to determine the maneuvers for robot docking in steps, which is designed based on potential fields and Dubins curves in the previous work of the authors. Its implementation is illustrated in Algorithm 3.

The parameters of the DAPF are selected as $k_s = 1, k_v = 1, \alpha = 0.1$.

*5.1. Scenario 1: Environment with Concave Obstacle.* Concave obstacle may cause the local minimum problem in the implements of the path planning approaches based on potential fields, which will prevent the robot from approaching the goal. In scenario 1, 9 static obstacles are employed to form

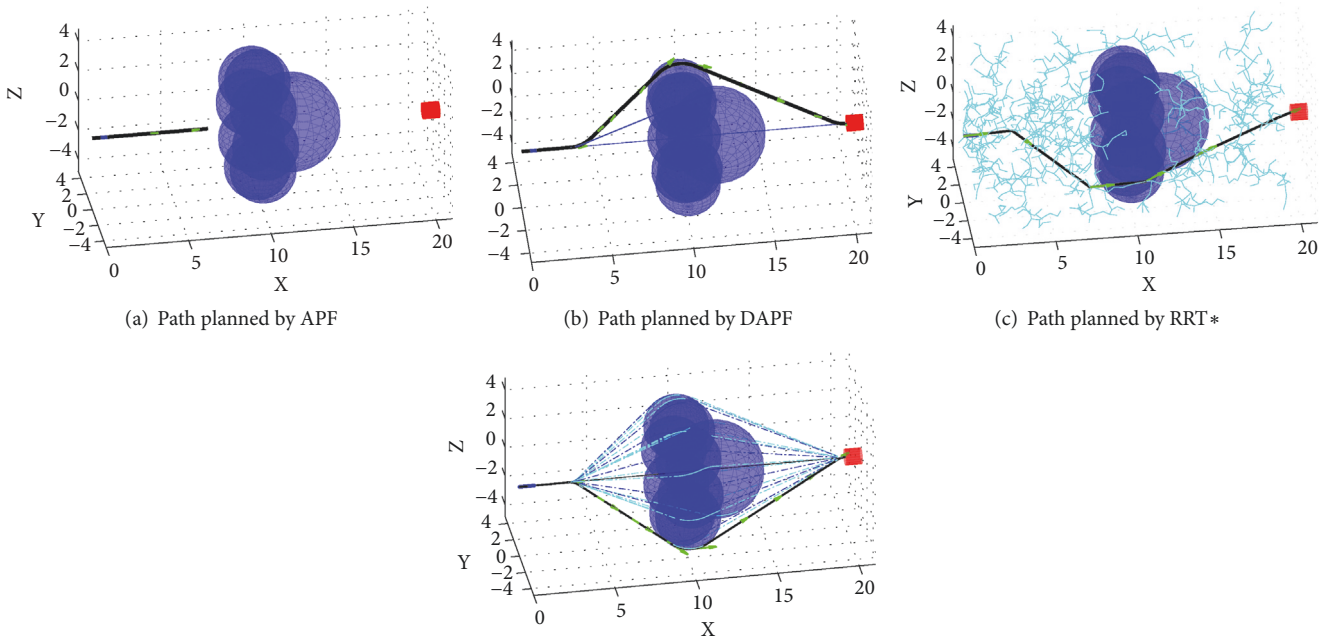
**Input:** $\mathscr{C}_r, \mathscr{C}_d, R_{min}, k_s, k_v, \alpha$
**Output:** **L**
(1) **L** $\longleftarrow$ **T**$(\mathscr{C}_r, \mathscr{C}_d, R_{min}, R_{min})$
(2) $\mathscr{C}' \longleftarrow [O, \overrightarrow{Vo}, R']$
(3) **while** $\mathscr{C}_r \neq \mathscr{C}_d$ **do**
(4) **if** $\{\mathscr{O}'\} \not\subset \{\mathscr{O}\}$ **then**
(5) $R_1 \longleftarrow R_2, R_2 \longleftarrow R'_g$
(6) **while f(L)** $= 0$ **do**
(7) $[T_g, O_g, R'_g, \mathscr{C}_m] \longleftarrow \min\{Tc_i \mid Dc_i < R'_i\}$
(8) $L_{pass-by} \longleftarrow$ **T**$(\mathscr{C}_r, \mathscr{C}_m, R_1, R_2)$
(9) **if f**$(L_{pass-by}) = 0$ **then**
(10) **L** $\longleftarrow L_{pass-by}, R_1 \longleftarrow R_2, R_2 \longleftarrow R_t$
(11) **continue**
(12) **end if**
(13) $L_{follow-up} \longleftarrow$ **T**$(\mathscr{C}_m, \mathscr{C}_d, R_1, R_2)$
(14) **if f**$(L_{follow-up}) = 0$ **then**
(15) Replan $L_{follow-up}$
(16) **else**
(17) **L** $\longleftarrow L_{pass-by} + L_{follow-up}$
(18) **end if**
(19) **end while**
(20) **end if**
(21) $\{\mathscr{O}\} \longleftarrow \{\mathscr{O}\} \cup \{\mathscr{O}'\}$
(22) follow **L**
(23) **end while**

Algorithm 3: The DAPF path planning approach.

a concave barrier, which may cause the local minimum problem and block the original path for docking. The configurations of the obstacles are set as

$$\begin{aligned}
O_1 &= (10, -2, -2)^{\mathrm{T}}, \\
O_2 &= (10, -2, 2)^{\mathrm{T}}, \\
O_3 &= (10, 2, 2)^{\mathrm{T}}, \\
O_4 &= (10, 2, -2)^{\mathrm{T}}, \\
O_5 &= \left(10, -2\sqrt{2}, 0\right)^{\mathrm{T}}, \\
O_6 &= \left(10, 0, 2\sqrt{2}\right)^{\mathrm{T}}, \\
O_7 &= \left(10, 0, -2\sqrt{2}\right)^{\mathrm{T}}, \\
O_8 &= \left(10, 2\sqrt{2}, 0\right)^{\mathrm{T}}, \\
O_9 &= (12, 0, 0)^{\mathrm{T}}, \\
R_1 &= 2m, \\
R_2 &= 2m, \\
R_3 &= 2m, \\
R_4 &= 2m, \\
R_5 &= 3m.
\end{aligned} \tag{37}$$

Resultant paths of the prescribed path planning approaches in scenario 1 are presented in Figure 7. As Figure 7(a) shows, the APF approach is unable to overcome the local minimum problem and trapped in the obstacle area. The resultant path of the DAPF approach is presented in Figure 7(b). The length of this path is $22.60m$; meanwhile, it only takes 5 branches in scenario 1 to obtain the resultant path. From this figure we can also see that the planned path is free from the affection of the concave barrier and reaches the docking station smoothly. The path planned by the RRT$*$ approach is presented in Figure 7(c), which shows that this approach avoids the concave barrier successfully and the path reaches the docking station as well. However, in this approach, it totally takes 2000 sample nodes and 1478 branches to determine the resultant path, whose length is $32.07m$. Therefore, it is inefficient in real-time applications especially when the calculation of branches is time-consuming. The resultant path of the smooth path planning approach is presented in Figure 7(c). From this figure we can see that the smooth path planning approach is capable of handling the local minimum problem and navigating the robot to dock with smooth path. It only takes 54 Dubins branches to determine the resultant path in this scenario, whose length is $23.15m$. Meanwhile, in the path planning process of our proposed approach, the feasible path for docking is selected from 8 evenly spread candidate paths in the obstacle area, which attempts to avoid the local optimal problem of the path planning result.




(a) Path planned by APF

(b) Path planned by DAPF

(c) Path planned by RRT∗

(d) Path planned by the smooth path planning approach

FIGURE 7: Simulation results of the path planning approaches in scenario 1.

*5.2. Scenario 2: Environment with Numerous Obstacles.* The efficiency of path planning approaches in complex environment is critical to the applications of robots in practice. In scenario 2, we present a discussion about the prescribed approaches in the environment crowded with numerous obstacles. To verify the performance of our smooth path planning approach, comparisons are implemented in two cases with slight difference.

*5.2.1. Case 1: Environment Crowded with Numerous Obstacles.* The configurations of these obstacles in case 1 are set as

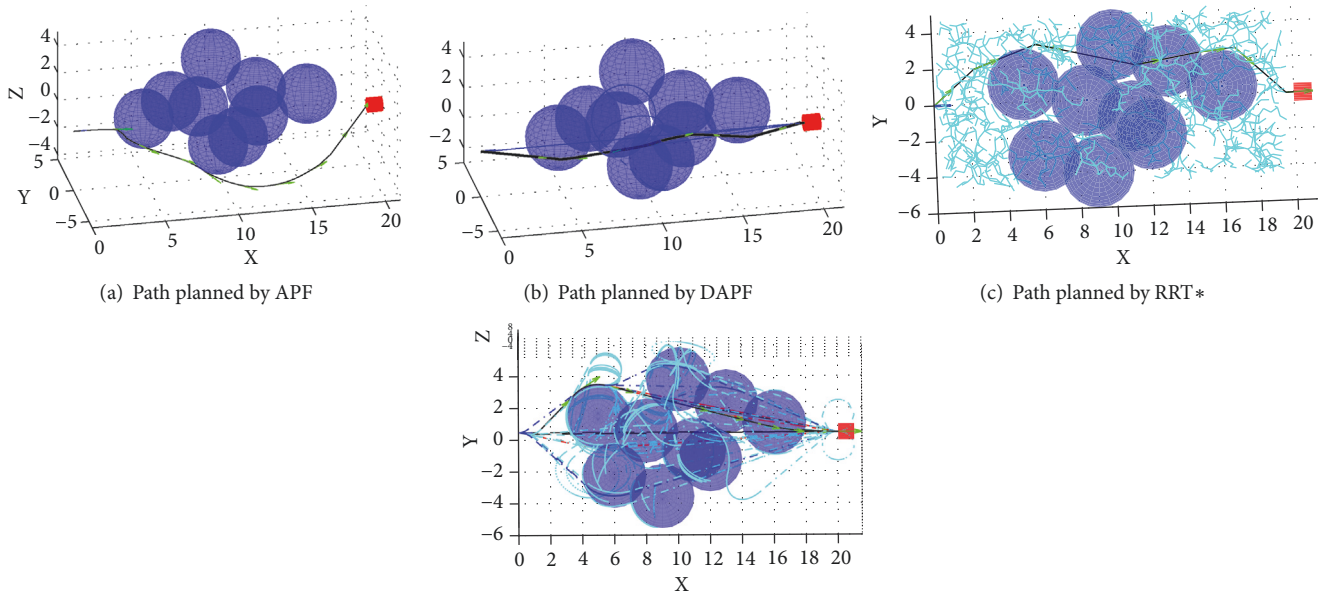
$$\begin{aligned} O_1 &= (5, 1, -1)^{\mathrm{T}}, \\ O_2 &= (6, -3, 3)^{\mathrm{T}}, \\ O_3 &= (8.5, 0, 1)^{\mathrm{T}}, \\ O_4 &= (9, -4, 0)^{\mathrm{T}}, \\ O_5 &= (10, 3, 3)^{\mathrm{T}}, \\ O_6 &= (11, -1, -1)^{\mathrm{T}}, \\ O_7 &= (12, -2, 2)^{\mathrm{T}}, \\ O_8 &= (13, 2, -1)^{\mathrm{T}}, \\ O_9 &= (16, 0.5, 1)^{\mathrm{T}}, \\ R_1 &= 2m, R_2 = 2m, \cdots R_9 = 2m. \end{aligned} \tag{38}$$

The path planning results of the approaches prescribed above in scenario 2 are presented correspondingly in Figure 8.

Figure 8(a) shows that the traditional APF can navigate the robot to approach the docking station with collision-free path in the crowded environment. The length of the resultant path is 23.50$m$; however, the feasibility of the robot's final pose is not ensured. Figure 8(b) presents the resultant path planned by the DAPF approach. It generates 11 branches in this approach to determine the maneuvers of the robot in the crowded environment, whose length is 21.60$m$. The path planned by the RRT∗ is presented in Figure 8(c), which proves that the RRT∗ is capable of planning the collision-free path as well through generating plenty of sample nodes and branches in the workspace. 1604 branches are evaluated and utilized to determine the resultant path whose length is 31.43$m$. Nevertheless, the calculation of this approach is time-consuming and the resultant path as well as the final pose is infeasible for the robot docking. The path planning result of our proposed approach in this paper is shown in Figure 8(d), which proves that this approach is capable of navigating the docking process of the robot smoothly in crowded environment with desired pose. It only takes 229 branches, which is less than the RRT∗, to determine the planned path whose length is 23.46.

*5.2.2. Case 2: Short-Sighted Problem.* The short-sighted problem indicates that the local maneuver is mainly focused on rather than the global feasibility in path planning. This problem would lead to the result that no feasible subsequent maneuver can be determined because of the previous low-quality movement.

This problem is considered in our comparison in case 2. To prove the performance of the smooth path planning approach, the environment of case 2 is modified based




(a) Path planned by APF (b) Path planned by DAPF (c) Path planned by RRT∗

(d) Path planned by the proposed approach

Figure 8: Simulation results of the path planning approaches in case 1, scenario 2.

on case 1, where $O_3$ is set as $(8, 0, 1)^T$ in this case. The path planning results of the DAPF and our smooth path planning approach are compared in this case, which are presented in Figure 9 separately. Figures 9(a), 9(b), and 9(c) present the performance of the DAPF approach in this case. These figures show that the path planning process of the DAPF is trapped in the crowded environment. Because the local environment is not sufficiently searched in this path planning approach, the robot is trapped in the obstacle area because of its constrained turning ability. Compared with the DAPF approach, more branches are generated in the work area of the robot in the proposed approach, which provides more possibility to determine the feasible path for docking. Thus, as Figure 9(d) shows, the smooth path planning approach is still capable of determining the feasible path for robot docking in the crowded environment. The length of the resultant path is $23.66m$, which is only slightly longer than in scenario 2. Meanwhile, our proposed path planning approach is easy to implement because only 130 branches are utilized in its path planning process in this case.

*5.3. Scenario 3: Simulation Results with Physical Engine.* Another simulation is implemented in the virtual robot experimentation platform V-REP with physical simulation engine to prove the validity of the proposed smooth path planning approach.

The obstacles in this simulation are set as

$$P_d = (10, 0, 0)^{\mathrm{T}},$$

$$\vec{V}_d = (1, 0, 0)^{\mathrm{T}},$$

$$O_1 = (3, -3, 0)^{\mathrm{T}},$$

$$O_2 = (3, 4.5, 0)^{\mathrm{T}},$$

$$O_3 = (3.5, 0.5, 0)^{\mathrm{T}},$$

$$O_4 = (6, -1.5, 0)^{\mathrm{T}},$$

$$O_5 = (6.5, 4, 0)^{\mathrm{T}},$$

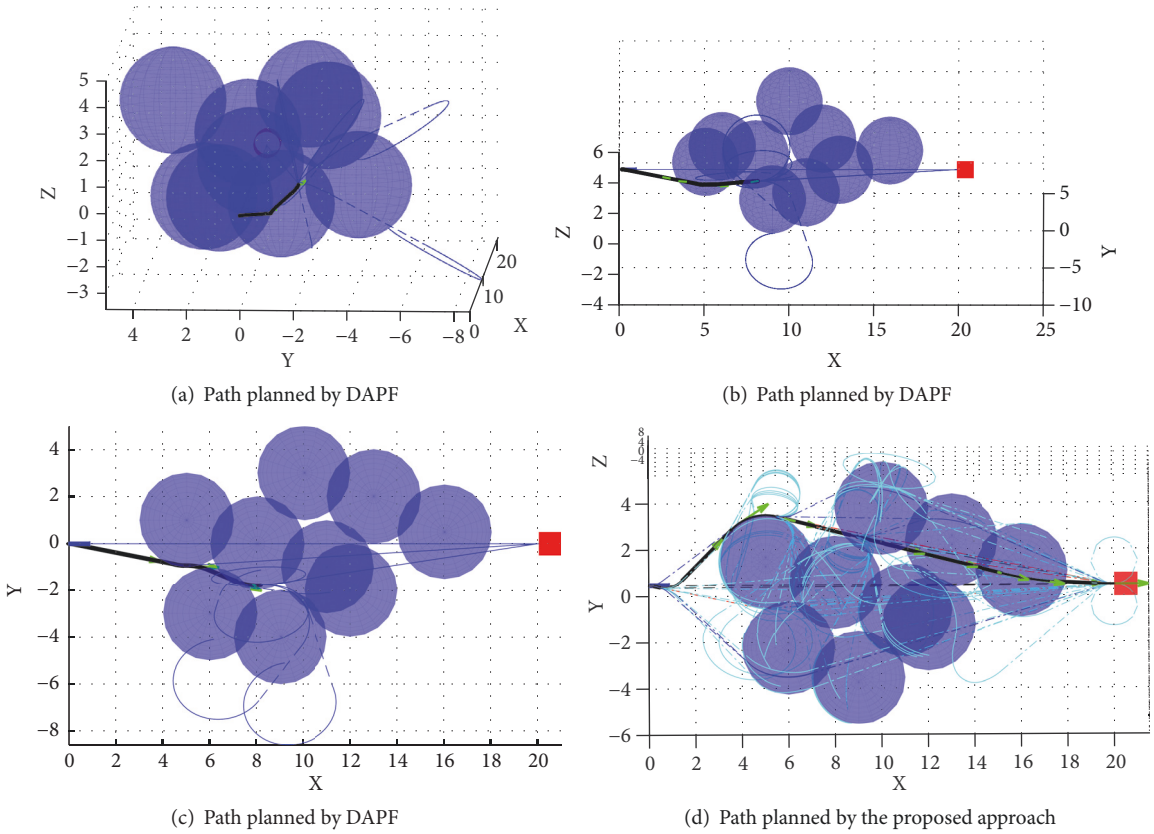
$$O_6 = (7.5, 1, 0)^{\mathrm{T}},$$

$$O_7 = (8, -4, 0)^{\mathrm{T}},$$

$$R'_1 = R'_2 = \cdots = R'_7 = 0.5m. \tag{39}$$

Because the practical constraints, such as physical size of the robot and the fitness of the ground, are considered in the path planning task, the radii of the obstacles are set as $R_i = 1m$ to remain $0.5m$ as the safety margin for the robot. Matlab simulation result of the smooth path planning approach in scenario 3 is presented in Figure 10(a). 129 branches are generated in the planning and replanning process to determine the feasible path for docking. Numeric simulation result proves that the proposed approach can determine the feasible path for 2D robot docking in obstacle environment.

Moreover, the performance of the proposed path planning approach is evaluated with physical simulation engine. We build the simulation scenario based on the walls and nature models provided in the V-REP PRO EDU software. The resultant path is replicated according to the Matlab simulation result and followed by the robot named Line Tracer in the V-REP. The Line Tracer robot only employs three sensors to trace the planned path, which are placed in its front-left, front-middle, and front-right sides. Additionally, it




(a) Path planned by DAPF

(b) Path planned by DAPF

(c) Path planned by DAPF

(d) Path planned by the proposed approach

Figure 9: Simulation results of the DAPF and the smooth path planning approach in case 2, scenario 2.

adopts an open-loop controller to control its mobile behavior, which means that the performance of path following mainly depends on the quality of path planning. Corresponding simulation result in V-REP is presented in Figures 10(b) and 10(c). Firstly, as Figure 10(b) shows, only a part of the obstacles can be detected, and a smooth path is generated to navigate the robot which is marked by cyan in this figure. As the robot moves on, the initial path is evaluated to be infeasible and the subsequent path for docking is replanned immediately. The path replanning result is presented in Figure 10(c), where the resultant path is marked by green and the maneuvers of the robot are marked by yellow. From this figure we can see that the Line Tracer robot follows the planned path stably and reaches the docking station eventually. The trajectory of the Line Tracer robot in these figures proves that our smooth path planning approach can navigate the mobile robot to dock and its resultant path is feasible for robots to follow.

## 6. Conclusion

To solve the path planning problem for robot docking in unknown obstacle environment, a smooth path planning approach has been proposed in this paper. Considering the pose constraints, Dubins curves have been adopted as the basic path segments and combined with a tree structure for path determination. Based on the elastic band theory, a heuristic strategy has been designed and has been employed to determine the nodes and branches for obstacle avoidance. The shortest collision-free route is chosen as the resultant path for docking after the sufficient expansion. The validity and feasibility of the smooth path planning approach are proved by simulation results.

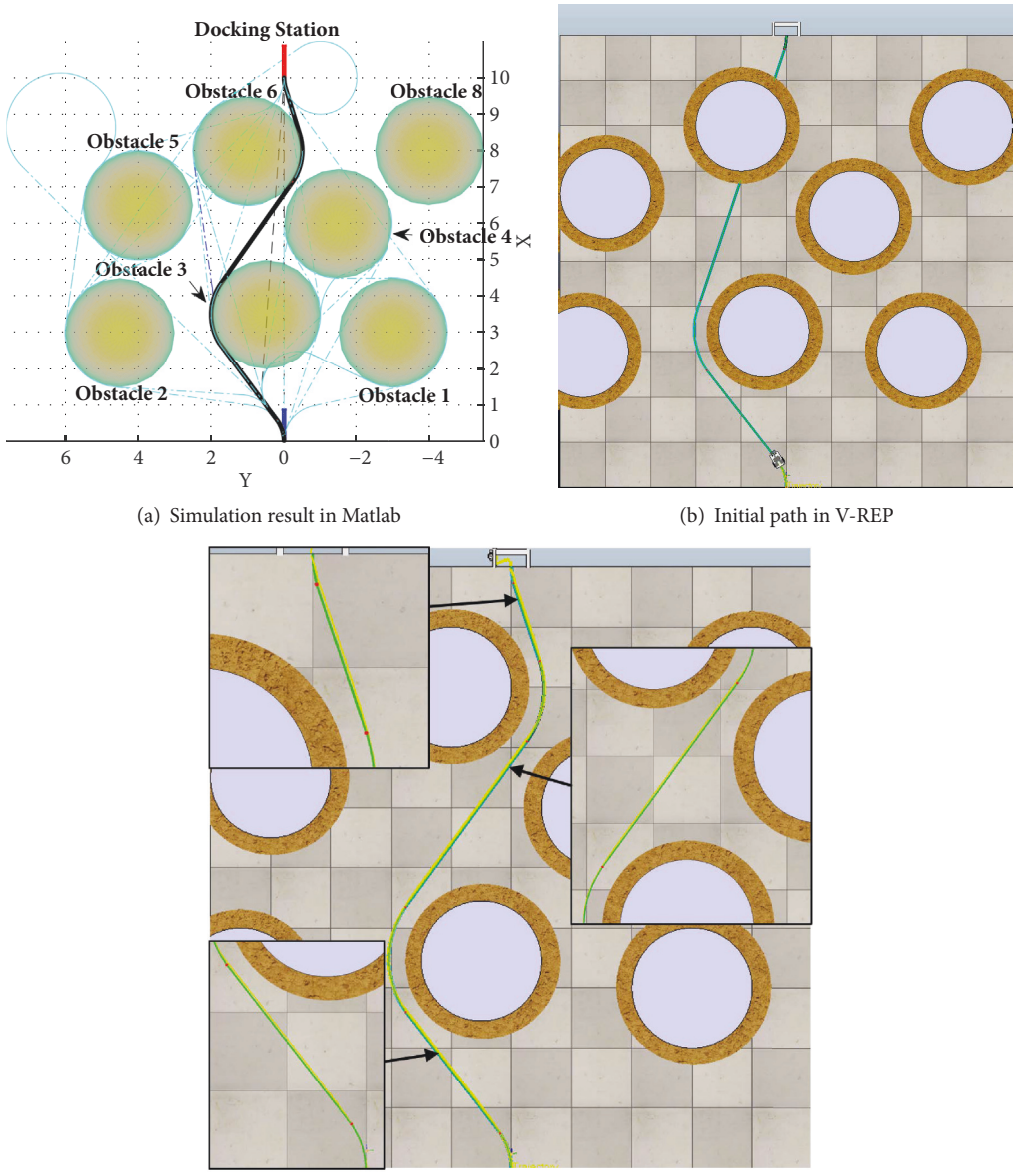
Our approach can handle the obstacle avoidance problem and planning path for the robots with pose constraints. In addition, this approach is efficient in determining the high-quality path for robot to follow. Future areas of researches can be concluded as follows: (i) the parallel computing techniques can be adopted to improve the efficiency of the smooth path planning approach in complex environment with numerous obstacles. (ii) The moving docking station should be considered, which is practical in some special applications of robots. (iii) Winds, water currents, and the changes of the ground materials may affect the feasibility of the resultant path, which should be considered in the further studies.

## 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 they have no conflicts of interest.




(a) Simulation result in Matlab

(b) Initial path in V-REP

(c) Feasible path in V-REP

Figure 10: The path planning result of the proposed approach in scenario 3.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China, under grant nos. 61733014, 51579210, 61472325, and 61633002, and the Science, Technology and Innovation Commission of Shenzhen Municipality under grant no. JCYJ20170817145216803.

## Supplementary Materials

5 figures (.png) are provided as the supplementary materials for our manuscript. These figures are the screenshots captured while running the simulation of scenario 3 in the V-REP software. Figure 1 to Figure 4 are captured when the Line Tracer robot is following the planned path towards the docking station, which shows that the planned path is collision-free and smooth to follow. Figure 5 is the enlarged screenshot when the Line Tracer robot docks in the station, which reveals that the pose of the robot for docking is feasible. *(Supplementary Materials)*

## References

[1] Y. Shi, C. Shen, H. Fang, and H. Li, "Advanced control in marine mechatronic systems: A survey," *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 3, pp. 1121–1131, 2017.

[2] C. Yoo, R. Fitch, and S. Sukkarieh, "Online task planning and control for aerial robots with fuel constraints in winds," in *Springer Tracts in Advanced Robotics*, vol. 107, pp. 711–727, Springer, Cham, 2015.

[3] J. S. Jaffe, P. J. S. Franks, P. L. D. Roberts et al., "A swarm of autonomous miniature underwater robot drifters for exploring submesoscale ocean dynamics," *Nature Communications*, vol. 8, 2017.

[4] J. Villagra, V. Milanés, J. Pérez, and J. Godoy, "Smooth path and speed planning for an automated public transport vehicle," *Robotics and Autonomous Systems*, vol. 60, no. 2, pp. 252–265, 2012.

[5] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1162–1171, 2017.

[6] M. Kovac, "Learning from nature how to land aerial robots," *Science*, vol. 352, no. 6288, pp. 895-896, 2016.

[7] T. Setter and M. Egerstedt, "Energy-Constrained Coordination of Multi-Robot Teams," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 4, pp. 1257–1263, 2017.

[8] H. Xiao, R. Cui, and D. Xu, "A Sampling-Based Bayesian Approach for Cooperative Multiagent Online Search With Resource Constraints," *IEEE Transactions on Cybernetics*, vol. 48, no. 6, pp. 1773–1785, 2018.

[9] R. Cui, Y. Li, and W. Yan, "Mutual information-based multi-AUV path planning for scalar field sampling using multidimensional RRT*," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 7, pp. 993–1004, 2016.

[10] G. Song, H. Wang, J. Zhang, and T. Meng, "Automatic docking system for recharging home surveillance robots," *IEEE Transactions on Consumer Electronics*, vol. 57, no. 2, pp. 428–435, 2011.

[11] N. Mathew, S. L. Smith, and S. L. Waslander, "Multirobot rendezvous planning for recharging in persistent tasks," *IEEE Transactions on Robotics*, vol. 31, no. 1, pp. 128–142, 2015.

[12] T. Oral and F. Polat, "Mod* lite: An incremental path planning algorithm taking care of multiple objectives," *IEEE Transactions on Cybernetics*, vol. 46, no. 1, pp. 245–257, 2016.

[13] H. Dong, Q. Hu, and M. R. Akella, "Safety control for spacecraft autonomous rendezvous and docking under motion constraints," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 7, pp. 1680–1692, 2017.

[14] C. Luo, S. X. Yang, X. Li, and M. Q.-H. Meng, "Neural-Dynamics-Driven Complete Area Coverage Navigation Through Cooperation of Multiple Mobile Robots," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 1, pp. 750–760, 2017.

[15] Y. Chen and G. W. Dorn II, "PINK1-phosphorylated mitofusin 2 is a parkin receptor for culling damaged mitochondria," *Science*, vol. 340, no. 6131, pp. 471–475, 2013.

[16] A. Macwan, J. Vilela, G. Nejat, and B. Benhabib, "A Multirobot Path-Planning Strategy for Autonomous Wilderness Search and Rescue," *IEEE Transactions on Cybernetics*, vol. 45, no. 9, pp. 1784–1797, 2015.

[17] H.-X. Wei, H.-Y. Li, Y. Guan, and Y.-D. Li, "A dynamics based two-stage path model for the docking navigation of a self-Assembly modular robot (Sambot)," *Robotica*, vol. 34, no. 7, pp. 1517–1528, 2016.

[18] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, vol. 47, no. 10, pp. 3148–3159, 2017.

[19] J. Park, B. Jun, P. Lee, Y. Lim, and J. Oh, "Docking problem and guidance laws considering drift for an underactuated AUV," in *Proceedings of the OCEANS 2011 - SPAIN*, pp. 1–7, Santander, Spain, June 2011.

[20] S. Jain and B. Argall, "Automated perception of safe docking locations with alignment information for assistive wheelchairs," in *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2014*, pp. 4997–5002, USA, September 2014.

[21] A. Yazdani, K. Sammut, A. Lammas, and Y. Tang, "Real-time quasi-optimal trajectory planning for autonomous underwater docking," in *Proceedings of the 2015 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS)*, pp. 15–20, Langkawi, Malaysia, October 2015.

[22] P. A. Wilson, "Autonomous homing and docking tasks for an underwater vehicle," in *Proceedings of the 8th International IFAC Conference on Manoeuvring and Control of Marine Craft, MCMC 2009*, pp. 304–309, Brazil, September 2009.

[23] P. B. Sujit, A. J. Healey, and J. B. Sousa, "AUV docking on a moving submarine using a K-R navigation function," in *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems: Celebrating 50 Years of Robotics, IROS'11*, pp. 3154–3159, USA, September 2011.

[24] M. Guerra, D. Efimov, G. Zheng, and W. Perruquetti, "Avoiding local minima in the potential field method using input-to-state stability," *Control Engineering Practice*, vol. 55, pp. 174–184, 2016.

[25] C. Liu, S. Fan, B. Li, S. Chen, Y. Xu, and W. Xu, "Path planning for autonomous underwater vehicle docking in stationary obstacle environment," in *Proceedings of the OCEANS 2016 - Shanghai*, China, April 2016.

[26] Y. Li, Y. Jiang, G. Zhang, Y. Li, and P. Chen, "AUV recovery path planning method considering geometrical constraints," *Jiqiren/Robot*, vol. 37, no. 4, pp. 478–485, 2015.

[27] J. Cao, J. Cao, Z. Zeng, B. Yao, and L. Lian, "Toward Optimal Rendezvous of Multiple Underwater Gliders: 3D Path Planning with Combined Sawtooth and Spiral Motion," *Journal of Intelligent & Robotic Systems*, vol. 85, no. 1, pp. 189–206, 2017.

[28] Y. Li, R. Cui, Z. Li, and D. Xu, "Neural Network Approximation-based Near-optimal Motion Planning with Kinodynamic Constraints Using RRT," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 11, pp. 8718–8729, 2018.

[29] M. Behandish and H. T. Ilies, "Analytic methods for geometric modeling via spherical decomposition," *Computer-Aided Design*, vol. 70, pp. 100–115, 2016.

[30] A. M. Shkel and V. Lumelsky, "Classification of the Dubins set," *Robotics and Autonomous Systems*, vol. 34, no. 4, pp. 179–202, 2001.

[31] Y. Meyer, P. Isaiah, and T. Shima, "On Dubins paths to intercept a moving target," *Automatica*, vol. 53, pp. 256–263, 2015.

[32] G. Ambrosino, M. Ariola, U. Ciniglio, F. Corraro, E. De Lellis, and A. Pironti, "Path generation and tracking in 3-D for UAVs," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 4, pp. 980–988, 2009.

[33] V. Duindam, J. Xu, R. Alterovitz, S. Sastry, and K. Goldberg, "Three-dimensional motion planning algorithms for steerable needles using inverse kinematics," *International Journal of Robotics Research*, vol. 29, no. 7, pp. 789–800, 2010.

[34] Y. Lin and S. Saripalli, "Path planning using 3D dubins curve for unmanned aerial vehicles," in *Proceedings of the 2014 International Conference on Unmanned Aircraft Systems, ICUAS 2014*, pp. 296–304, Orlando, FL, USA, May 2014.

[35] H. J. Sussmann, "Shortest 3-dimensional paths with a prescribed curvature bound," in *Proceedings of the 1995 34th IEEE Conference on Decision and Control. Part 1 (of 4)*, pp. 3306–3312, December 1995.

[36] P. Cui, W. Yan, and Y. Wang, "Reactive path planning approach for docking robots in unknown environment," *Journal of Advanced Transportation*, vol. 2017, 2017.

[37] R. Vatcha and J. Xiao, "Detection of robustly collision-free trajectories in unpredictable environments in real-time," *Autonomous Robots*, vol. 37, no. 1, pp. 81–96, 2014.

[38] Y. Lu, Z. Xi, and J.-M. Lien, "Online collision prediction among 2D polygonal and articulated obstacles," *International Journal of Robotics Research*, vol. 35, no. 5, pp. 476–500, 2015.

[39] T. Sawabe, M. Kanbara, N. Ukita et al., "Comfortable autonomous navigation based on collision prediction in blind occluded regions," in *Proceedings of the IEEE International Conference on Vehicular Electronics and Safety, ICVES 2015*, pp. 75–80, Japan, November 2015.

[40] R. Cui, B. Gao, and J. Guo, "Pareto-optimal coordination of multiple robots with safety guarantees," *Autonomous Robots*, vol. 32, no. 3, pp. 189–205, 2012.

[41] M. Keller, F. Hoffmann, T. Bertram, C. Hass, and A. Seewald, "Planning of optimal collision avoidance trajectories with timed elastic bands," in *Proceedings of the 19th IFAC World Congress on International Federation of Automatic Control, IFAC 2014*, pp. 9822–9827, South Africa, August 2014.

[42] T. Gu, J. Atwood, C. Dong, J. M. Dolan, and J.-W. Lee, "Tunable and stable real-time trajectory planning for urban autonomous driving," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2015*, pp. 250–256, Germany, October 2015.

[43] J. Hilgert, K. Hirsch, T. Bertram, and M. Hiller, "Emergency path planning for autonomous vehicles using elastic band theory," in *Proceedings of the 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM 2003*, pp. 1390–1395, Japan, July 2003.

[44] Y. I. Jenie, E.-J. van Kampen, C. C. de Visser, J. Ellerbroek, and J. M. Hoekstra, "Three-dimensional velocity obstacle method for UAV deconflicting maneuvers," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference 2015, MGNC 2015 - Held at the AIAA SciTech Forum 2015*, USA, January 2015.