



LJMU Research Online

Liu, X, Hu, Z, Yang, Z and Zhang, W

Graph Search-Based Path Planning for Automatic Ship Berthing

<http://researchonline.ljmu.ac.uk/id/eprint/25994/>

Article

Citation (please note it is advisable to refer to the publisher's version if you intend to cite from this work)

Liu, X, Hu, Z, Yang, Z and Zhang, W (2024) Graph Search-Based Path Planning for Automatic Ship Berthing. Journal of Marine Science and Engineering, 12 (6).

LJMU has developed **LJMU Research Online** for users to access the research output of the University more effectively. Copyright © and Moral Rights for the papers on this site are retained by the individual authors and/or other copyright owners. Users may download and/or print one copy of any article(s) in LJMU Research Online to facilitate their private study or for non-commercial research. You may not engage in further distribution of the material or use it for any profit-making activities or any commercial gain.


The version presented here may differ from the published version or from the version of the record. Please see the repository URL above for details on accessing the published version and note that access may require a subscription.

For more information please contact researchonline@ljmu.ac.uk

<http://researchonline.ljmu.ac.uk/>

Article

Graph Search-Based Path Planning for Automatic Ship Berthing

Xiaocheng Liu, Zhihuan Hu *, Ziheng Yang and Weidong Zhang * 

Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, China; liuxc@sjtu.edu.cn (X.L.); sjtu_yzh@sjtu.edu.cn (Z.Y.)

* Correspondence: scar1et@sjtu.edu.cn (Z.H.); wdzhang@sjtu.edu.cn (W.Z.)

Abstract: Ship berthing is one of the most challenging operations for crews, involving optimal trajectory generation and intricate harbor maneuvering at low speed. In this paper, we present a practical path-planning method that generates smooth trajectories for an underactuated surface vehicle (USV) traveling in a confined harbor environment. Our approach introduces a Generalized Voronoi Diagram (GVD)-based path planner to handle the unberthing phase. The hybrid A* search-based path finding method is used for the transportation phase. A simple planner based on a Bézier curve is proposed for the berthing phase. To track the target path, an adaptive pure pursuit method and proportional-derivative (PD) controller is used. The performance of the given method is tested numerically and experimentally on a catamaran with a pair of non-steerable thrusters. The results demonstrate that the proposed algorithm can achieve a successful berthing operation through static obstacle handling and smooth trajectory generation.

Keywords: path planning; underactuated surface vehicle; automatic berthing; hybrid A* search; Bézier curve

1. Introduction

Bringing about an automatic berthing system for USVs is a complicated task. Such an operation involves moving from open waters or a harbor area toward the berthing position, which requires advanced maneuvering skills. Especially when the ship hits the quay, high-precision movements are required for motion control. Therefore, it is necessary to conduct research into the automatic berthing operation.

The full-scale experiment on the automatic berthing operation of a fully-actuated surface vehicle was demonstrated in the 1990s [1]. Since then, various studies have been performed on the simple berthing operation without surrounding obstacles. The motion control of the berthing operation is highly nonlinear, as various types of maneuvers are required, including reversal, turning, stopping and accelerating motion. Due to the non-linearity of the ship's motion, many researchers solved the berthing problem using the optimal control, which was modeled as a minimum-time optimization problem. The discrete augmented Lagrangian approach [2] was used to generate the optimal trajectory while considering the actuator capacity and nonlinearity of the ship model. A covariance matrix adaptation evolution strategy (CMA-ES) [3] solver was used to give the berthing solutions. A multiple shooting algorithm [4] has been introduced to give the sub-optimal solution for the minimum-time optimization problem. An artificial neural network (ANN) [5–8] has been proposed to learn how to navigate from the starting position to the berthing quay. The ANN was trained using the simulated data under different wind conditions. The effectiveness of the ANN-based path planner has been verified in the automatic berthing simulation under wind disturbance. A hybrid method [9] has been proposed to predict the ship motion, based on the ANN and dynamic model. The trained data include the relative wind speed and direction, measured position, altitude and velocity, etc. However, it is essential to consider the harbor geometry and surrounding obstacles in the path-planning method for the automatic berthing operation.



Citation: Liu, X.; Hu, Z.; Yang, Z.; Zhang, W. Graph Search-Based Path Planning for Automatic Ship Berthing. *J. Mar. Sci. Eng.* **2024**, *12*, 933. <https://doi.org/10.3390/jmse12060933>

Academic Editor: Sergei Chernyi

Received: 30 April 2024

Revised: 26 May 2024

Accepted: 28 May 2024

Published: 2 June 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

To consider the harbor layout and surrounding obstacles, the optimization-based path planners are often preferable, as the vehicle constraints, path smoothness and dynamic environments can be explicitly formulated as an optimization problem. The berthing problem was formulated as a nonlinear programming one [10], where the spatial constraints were enforced through a convex approximation of the collision-free reachable space. The simulation demonstrated that their proposed method could generate a collision-free path toward the target pose. The multi-objective optimization [11] with the trade-off between energy consumption, berthing accuracy and collision avoidance was presented for the path planning of the berthing and unberthing operation. The effects of wind disturbance and user-specified waypoints were also considered in the optimization problem. To generate more efficient trajectories, the pseudo-Huber cost was introduced in the cost function of online optimization [12]. Their method utilizes the real-time mapping of the harbor and obstacles using LIDAR and ultrasonic sensors. For these optimization-based approaches, the optimization is non-convex and time-consuming; this means the feasible and efficient trajectory generation cannot be guaranteed [13].

To develop fast and online algorithms for the berthing trajectory, the graph search methods [14], including Dijkstra, the A* and D* algorithm, etc., can be useful in finding a feasible path from the starting position to the berthing pose. For the search-based planner, the state space of a vehicle is represented as a discrete grid space, where the smoothness of the best path cannot be guaranteed. The hybrid A* algorithm [15] was proposed to generate a drivable path for a self-driving car in the unstructured environment. By replacing the discrete state spaces with the precomputed motion primitives, the hybrid A* method can take into account the physical limits of a vehicle. However, the best paths produced by hybrid A* are not smooth and often sub-optimal. Further improvement of the resulting A* path is required. The path smoothing method based on a Voronoi potential field [16] was used to improve the quality of the resulting path. In [17], the authors utilized the hybrid A* path as a warm start to the nonlinear optimization, which produced the smooth and energy-optimized trajectory for the berthing operation.

A practical path-planning algorithm for the berthing operation must generate a smooth, feasible and obstacle-free path in real-time, while maintaining the accuracy required for the berthing control. In this paper, we present a two-stage path planner for the berthing operation of a small USV. At the first stage, we extend the hybrid-state A* search with analytic expansions based on the Dubins curve [18]. It considers the kinematic constraints of the USV by constructing motion primitives. The improvement of the resulting A* path is achieved by a path smoother. If the hybrid A* search fails to find a solution, it will produce a temporary path based on the Generalized Voronoi Diagram (GVD) of the surrounding obstacles. During the second stage, it is assumed that there is no dynamic obstacle between the final berthing state and the goal state of the hybrid A* search. A cubic Bézier curve is used to link these two states, which can guide the vehicle into the final berthing pose with sufficient precision. The pure pursuit method and PD controller are implemented to track the target path, which is the concatenation of two stages (see Figure 1). In the experiments on a small USV equipped with a pair of non-steerable thrusters, the proposed method can generate a collision-free and smooth path and achieves successful berthing.

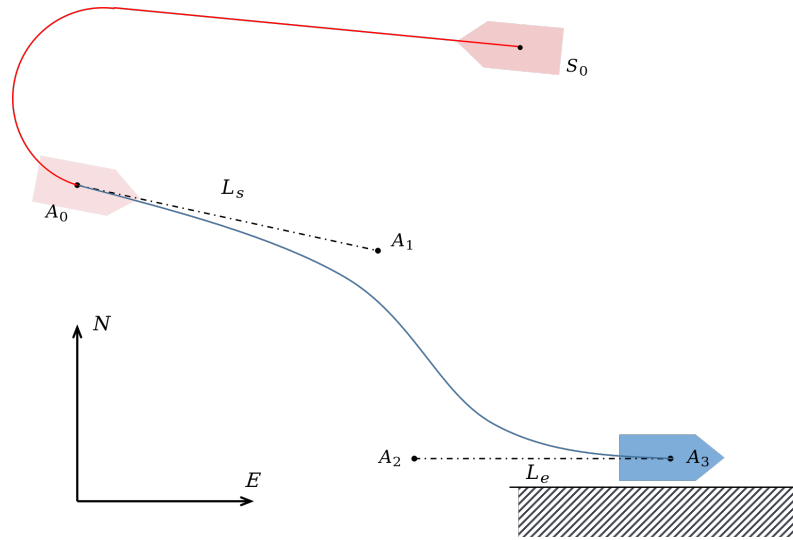


Figure 1. Concatenation of two curves for berthing operation (red line indicates the resulting hybrid A* path; blue line means the Bézier curve).

2. Theoretical Background

2.1. Modelling and Identification

The experimental USV (see Figure 2) is equipped with a pair of propellers, whose parameters are listed in Table 1. We develop and implement a 3-Degree of Freedom (DoF) equation of motion for USV.

$$\dot{\eta} = T(\eta) \cdot \mathbf{v}, \quad T(\eta) = \begin{bmatrix} \cos \varphi & -\sin \varphi & 0 \\ \sin \varphi & \cos \varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (1)$$

$$(M_{RB} + M_A) \cdot \dot{\mathbf{v}} + (C_{RB}(\mathbf{v}) + C_A(\mathbf{v}) + D(\mathbf{v})) \cdot \mathbf{v} = \boldsymbol{\tau}_c \quad (2)$$

Here, vector $\eta = [N, E, \varphi]^T$ contains position (N, E) in the north and east coordinates and is heading φ from true north. $\mathbf{v} = [u, v, r]^T$ denotes velocity vector in the body-fixed coordinate; u, v, r are surge, sway velocity and rate of turning, respectively. $T(\eta)$ is the coordinate transformation matrix. According to the literature [19],

$$M_{RB} = \begin{bmatrix} m & 0 & 0 \\ 0 & m & mx_g \\ 0 & mx_g & I_z \end{bmatrix}, M_A = \begin{bmatrix} -X_u & 0 & 0 \\ 0 & -Y_v & -Y_r \\ 0 & -N_v & -N_r \end{bmatrix} \quad (3)$$

where m denotes the mass, x_g center of gravity (CoG) and I_z denotes moment of inertia about z-axis. The Coriolis $C_{RB}(\mathbf{v})$, centrifugal $C_A(\mathbf{v})$ and damping matrix $D(\mathbf{v})$ are given by

$$C_{RB}(\mathbf{v}) + C_A(\mathbf{v}) = \begin{bmatrix} 0 & -mr & -mx_g r + Y_v v + \gamma \\ mr & 0 & -X_u u \\ mx_g r - Y_v v - \gamma & X_u u & 0 \end{bmatrix}, \gamma = \frac{Y_r + N_v}{2} \quad (4)$$

$$D(\mathbf{v}) = - \begin{bmatrix} X_u + X_{|u|u}|u| & 0 & 0 \\ 0 & Y_v + Y_{|v|v}|v| & 0 \\ 0 & 0 & N_r + N_{|r|r}|r| \end{bmatrix} \quad (5)$$

The linear drag and quadratic drag coefficient are considered in damping matrix. Figure 3a shows the propellers on PORT and STBD side. By calculating the thrust τ_{port} and τ_{stbd} of two propellers, the control input τ_c is given by

$$\tau_c = [\tau_{port} + \tau_{stbd}, 0, (\tau_{port} - \tau_{stbd})l] \tag{6}$$

For each propeller, the thrust force is given by

$$\tau_e = c_1 \rho d^4 |n|n - c_2 \rho d^3 u_a |n| \tag{7}$$

where ρ, d, n denote the water density, propeller diameter and the rotational speed, respectively. The inflow velocity u_a of each propeller is calculated independently:

$$u_{a-port} = u + lr \quad u_{a-stbd} = u - lr \tag{8}$$

The four-quadrant model [20] is used for coefficients (c_1, c_2). As shown in Figure 3b, $c_1 = a_2, c_2 = 0$ if $n < 0, u_a \geq 0$, and so on. With $\rho = 1000 \text{ kg/m}^3, d = 0.24 \text{ m}, l = 0.68 \text{ m}$, the identification of the modelling parameters (see Table 2) was performed using the zigzag and spiral maneuver test as well as a bollard pull test. The more detailed description of such a model identification test can be found in this paper [21].



Figure 2. Experimental USV.

Table 1. Vehicle specifications.

Items	Unit	Value
Length	m	3.1
Width	m	1.8
Weight	kg	244.0
CoG	m	0.68

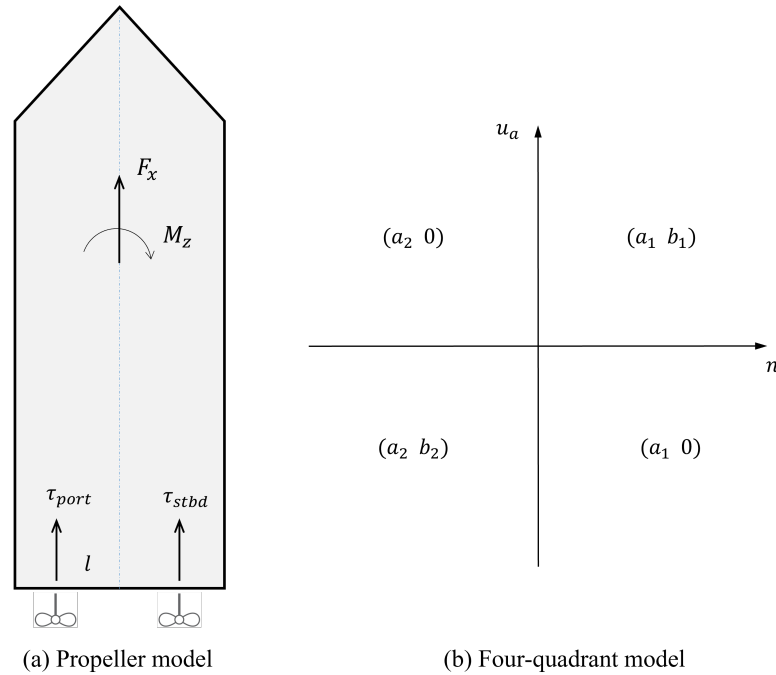


Figure 3. Propulsion model ($a_1 = 0.0618, a_2 = 0.0271, b_1 = b_2 = 0.136$).

Table 2. The modeling parameters of USV.

Items	Unit	Value
I_z	kg	192.0
$Y_{\dot{\nu}}$	kg	-72.1
$Y_{\dot{r}}$	kgm	-179.2
$N_{\dot{\nu}}$	kgm	-132.8
$N_{\dot{r}}$	kgm ²	-828.8
X_{uu}	Ns/m	-8.6
Y_{vv}	Ns/m	-232.3
N_r	Nms/rad	-171.2
$X_{ u u}$	Ns ² /m ²	-48.5
$Y_{ v v}$	Ns ² /m ²	-81.2
$N_{ r r}$	N	-163.1

2.2. Generalized Voronoi Diagram (GVD)

If the edges of each obstacle are decomposed into many discrete points, we can generate a Voronoi diagram on a 2-dimensional plane around these points using Fortune’s algorithm [22]. When the eliminations on those Voronoi edges intersect one of the obstacles, the remaining edges form a good approximation of the Generalized Voronoi Diagram (GVD) [23]. The GVD provides the safest path around the environment, such as obstacles or areas that the vehicle cannot traverse. Figure 4a,b shows an example of the Voronoi diagram and GVD.

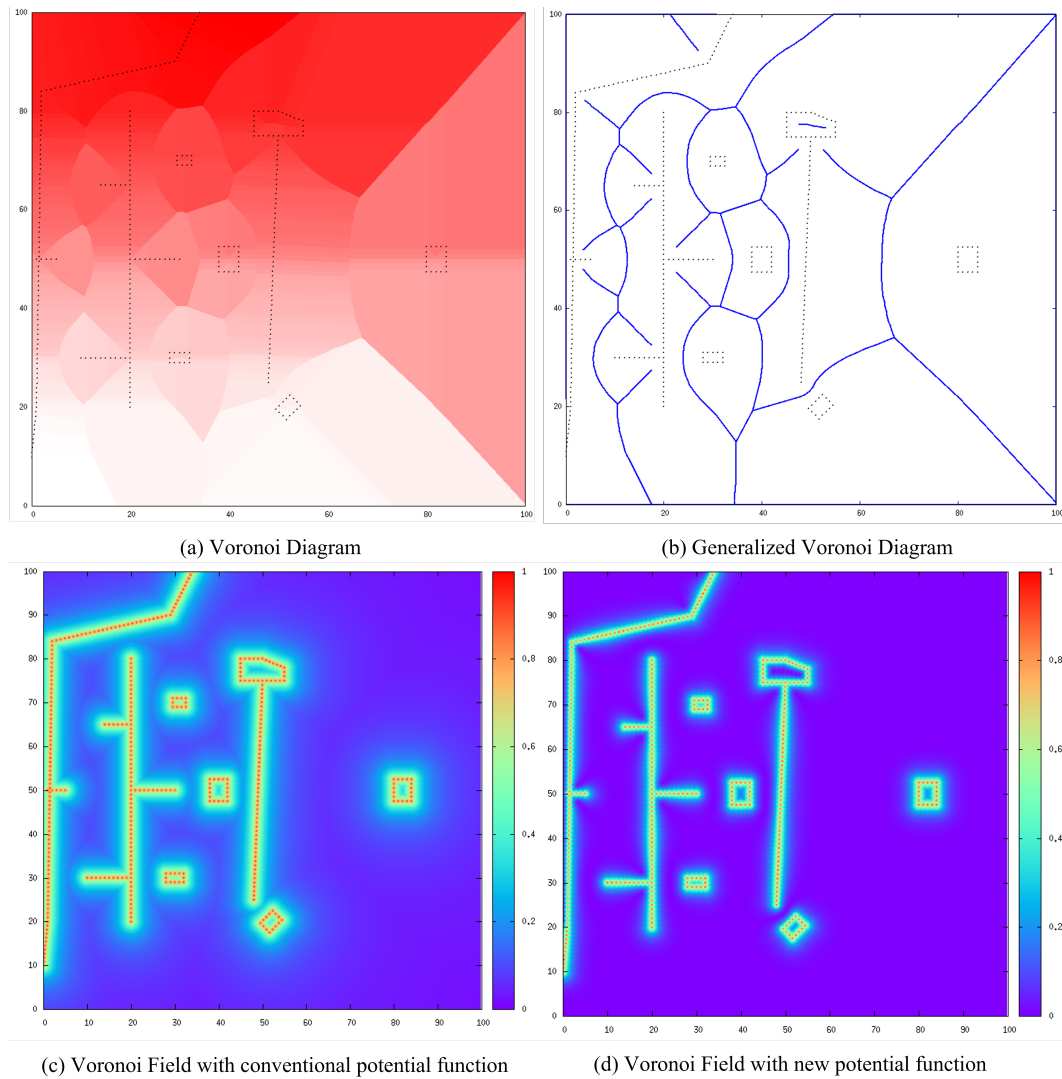


Figure 4. GVD and Voronoi potential field.

2.3. Path Planning at the Unberthing Phase

The unberthing phase involves the movement from the confined space to freer waters. We use the GVD for navigation during the unberthing phase. In order to generate the feasible path for vehicle with kinematic constraints, the motion primitives of the under-actuated vehicle are introduced, as shown in Figure 5. These motion primitives can be computed based on the maximum curvature and movement length. We exclude the motion primitives with high risk of collision and compute the Voronoi potential value of each movement based on GVD. The trajectory with the minimum potential value will be used to navigate the vehicle in the unberthing phase.

2.4. Hybrid A* Search Method

To ensure the kinematic feasibility and collision avoidance of the planned path, the four-dimensional search space is applied in hybrid A* search algorithm. Search with collision checking is also considered.

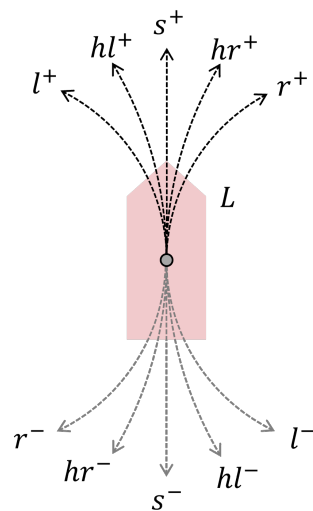


Figure 5. Motion primitives (L denotes the movement length).

2.4.1. Costmap and Single-Source Shortest Path (SSSP) Map

The costmap is used to represent the planning space around vehicle. It represents a map of environment as an evenly spaced field, often called a grid cell. Each grid cell has a value in the range $[0, 255]$ that indicates the likelihood the cell contains an obstacle. Firstly, we decompose the vehicle shape into a set of circles with the same radius R , called the inflation radius. Then, the shape of obstacles is inflated by computing the smallest integer number not less than R/d . Figure 6 shows an example of obstacle inflation, where the grid cell in the center is specified by 255 and the cells completely enclosing the circle are specified by 204. After the costmap is generated, the collision check will be performed by checking whether the center points of overlapping circles lie on the inflated area. For instance, the vehicle pose in Figure 6 is considered in collision.

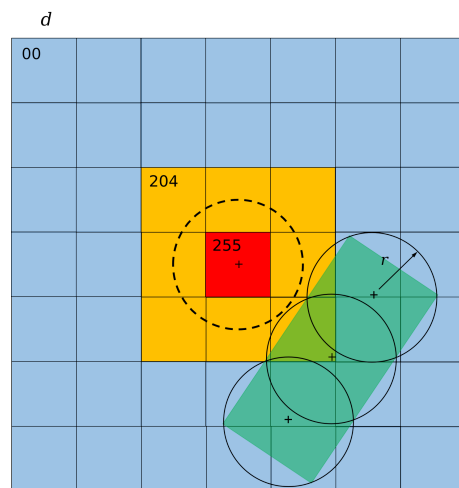


Figure 6. 2D costmap.

The costmap inflates the size of obstacles based on the inflation radius. Decreasing the inflation radius will lead to more precise collision checking. Increasing the inflation radius makes it possible for vehicle to maintain a safe distance from obstacles. However, a large inflation radius will overinflate the obstacles, which is undesirable for path planning in narrow waters. Figure 7a shows an example of costmap with a large inflation radius, where the narrow channel becomes unnavigable. In this paper, we present a novel costmap using adaptive inflation radius, where the inflation radius of each grid cell (x, y) is determined based on the distance from grid cell to its nearest edge of the GVD $d_V(x, y)$. Figure 7c

shows the costmap with the adaptive inflation radius, where the inflated areas are reduced around the narrow channel.

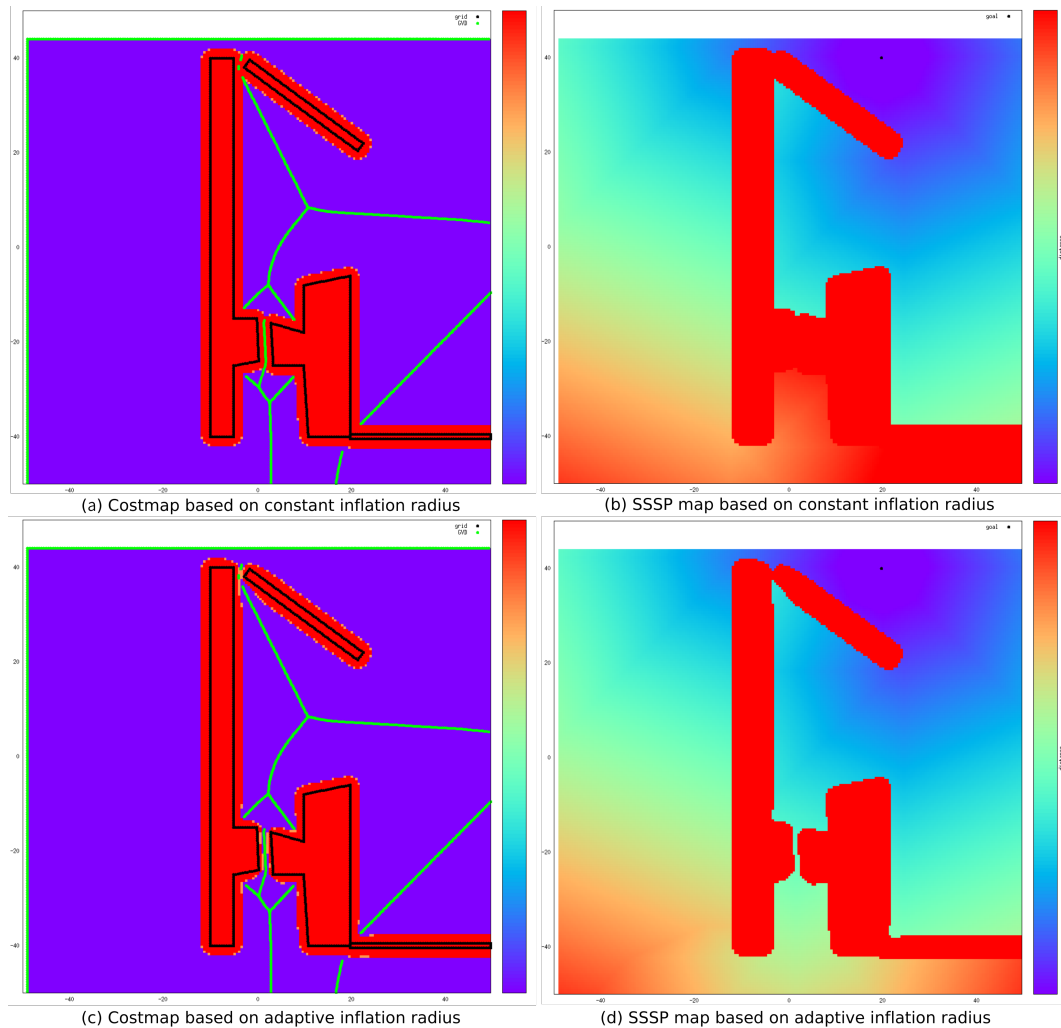


Figure 7. Comparison of costmap and SSSP map using constant and adaptive inflation radius. (In costmap, black dots indicate the shape of obstacles, green dots represent the edge of GVD, and cost value of each cell is described by a color bar. In SSSP map, black dot represents the goal, and the distances of the shortest path from all cells to goal are described by a color bar.)

The 2D map of Single-Source Shortest Path (SSSP) consists of the distances of the shortest paths from the goal and all other vertices in the graph. It uses the costmap to compute the shortest distances by performing Dijkstra’s algorithm, ignoring the non-holonomic nature of vehicle. Figure 7b,d illustrate the maps of SSSP. The SSSP map will never overestimate the actual cost to reach the goal, meaning that it can be used as an admissible heuristic function for hybrid A* search.

2.4.2. Dubins and Reeds–Shepp Curves

Dubins and Reeds-shepp curves (see Figure 8) are commonly used for non-holonomic vehicles, which are analytical methods to generate the optimal path in the two-dimensional plane [24]. With the curvature constraint on path, the Dubins path finds the shortest curve for forward-only vehicles that connects the initial pose and goal pose of vehicle. If the vehicle can also travel in reverse, then the optimal path follows the Reeds–Shepp curve [25].

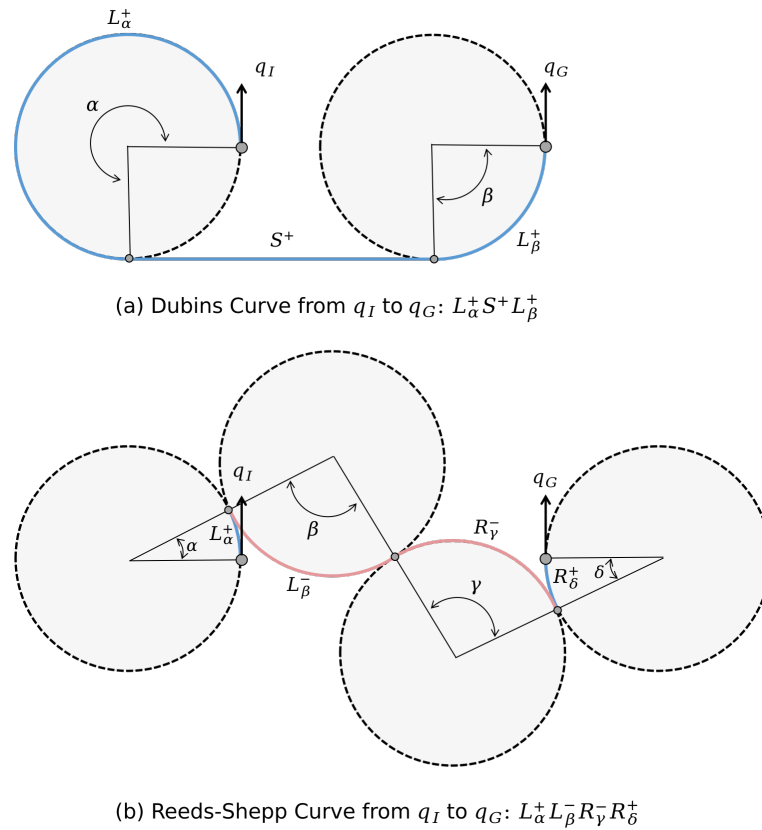


Figure 8. Dubins curve and Reeds–Shepp curve.

2.4.3. Search Space Representation

The A* search extends its paths based on the cost of path $g(n)$ and an admissible heuristic $h(n)$ that estimates the cost to get to the goal state.

$$f(n) = g(n) + h(n) \tag{9}$$

In the hybrid A* search, the path extension is achieved by a set of pre-computed motion primitives (see Figure 5). It means that the smallest entities of path extension are defined by arcs, which associates a continuous state with search space. The arc length L should be guaranteed to leave the current grid cell.

$$L > \sqrt{2} \cdot d \tag{10}$$

The four-dimensional search space (x, y, φ, d) , including the location, orientation and direction of motion, are used to give an estimate of the cost of path. It applies higher penalties for traveling in the reverse direction and switching the direction of motion. The heuristic function in hybrid A* has been discussed above. During the node extension, the Dubins or Reeds–Shepp curves are computed from the current node to the goal. If these analytic curves are collision-free against the obstacles, the hybrid A* search will terminate.

2.5. Bézier Curve

Due to the resolution of costmap and movement length in hybrid A* search, the search-based planner will never reach the exact goal pose of vehicle. To address this precision issue, we augment the search with analytic expansions based on a cubic Bézier curve [26].

$$B(t) = (1 - t)^3 A_0 + 3t(1 - t)^2 A_1 + 3t^2(1 - t) A_2 + t^3 A_3, \quad 0 \leq t \leq 1 \tag{11}$$

where the Bézier curve (see Figure 1) is defined by four points, A_0, A_1, A_2, A_3 , in the plane. The point A_0 is the goal of hybrid A* search planner. The A_3 is placed at the berthing position, where the orientation of line segment A_2A_3 is the same as the berthing orientation. The adjustment of lengths L_s and L_e leads to a smooth berthing trajectory for the vehicle to follow. Such a Bézier-based planner is appropriate only under the assumption that the environment around the berthing position is collision-free.

2.6. Path-Following Method

The path-following algorithm makes the vehicle follow the target path smoothly and accurately. In this paper, the adaptive pure pursuit algorithm is used to generate the target course for the ASV. As shown in Figure 9, the target point can be determined by finding the first point on the target path whose distance to vehicle is larger than the look-ahead distance L_T . The adaptive look-ahead distance L_T will vary with the curvature of target path. When the vehicle goes around a sharp turn, look-ahead distance will be reduced. The course error θ_T can be computed based on the target point on the target path. A PD controller is applied to compute the thruster command based on the speed and course error. The proposed path-following method ensures the performance to control a ship with sufficient precision.

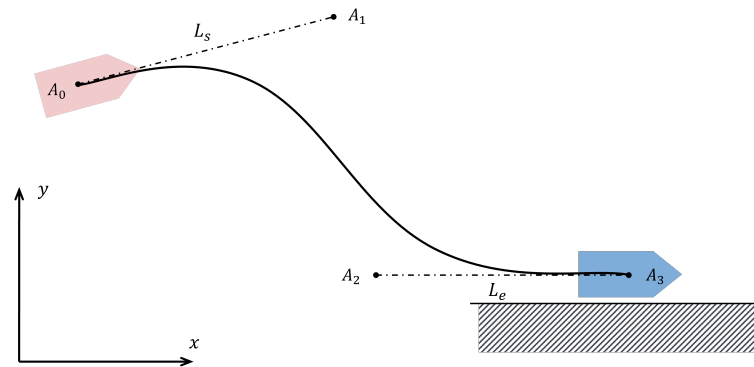


Figure 9. Pure pursuit algorithm: L_T indicates the look ahead distance, and θ_T means the angle error between target and estimated course.

2.7. Architecture of Automatic Berthing Algorithm

The architecture of path-planning algorithm for automatic berthing operation is illustrated in Figure 10. If the hybrid A* search planner fails to compute a feasible path, the unberthing path planner will be activated, which makes the vehicle move toward open waters. After the unberthing phase is finished, the hybrid A* search planner re-computes a smooth and collision-free path from the current pose to the goal. Combined with the Bézier curve, the automatic berthing algorithm provides a safe trajectory with high precision.

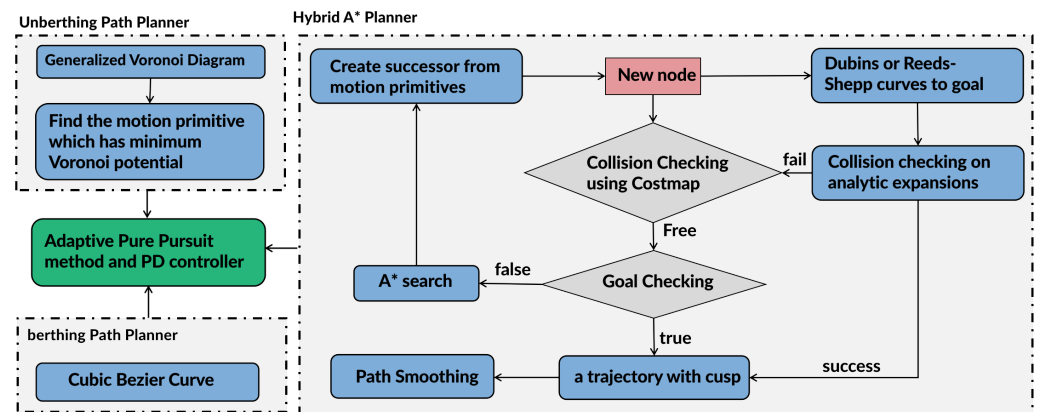


Figure 10. Algorithm overview.

3. Case Study: Simulations and Field Trials

The case study on two types of berthing operations (see Figure 11) is presented in this paper. The berthing precision is defined as follows.

$$P_b = \max\left(\frac{|d|}{\hat{d}}, \frac{|\theta - \theta_d|}{\hat{\theta}}, \frac{|u|}{\hat{u}}\right) \tag{12}$$

where $|d|$ and \hat{d} denote the tracking error and the maximum allowed tracking error, $|\theta - \theta_d|$ represents the heading error between the real-time and target heading angle, and $\hat{\theta}$ denotes the maximum allowed heading error. $|u|$ and \hat{u} denote the vehicle speed and the maximum allowed speed. If $P_b < 1.0$, it means the berthing precision is satisfactory. For the perpendicular dock, $\hat{d} = 1.0$ m, $\hat{\theta} = 0.05$ rad, $\hat{u} = 0.4$ m/s. For the parallel dock, $\hat{d} = 0.2$ m, $\hat{\theta} = 0.1$ rad, $\hat{u} = 0.4$ m/s.

We used the following parameters for the berthing planner: the costmap and hybrid A* search use the size 200 m × 200 m with a 0.5 m resolution. The proposed algorithms, including the path-planning, path-following and control, are implemented on a computer with an ARM 1.8 GHz CPU. We maintained the costmap and GVD at every iteration of the hybrid A* search, whose run time was on the order of 30–200 ms. The parameters in PD heading controller $K_p = 300, K_d = 200$ are set, and the look-ahead distance in the pure pursuit algorithm is 4.5 m.

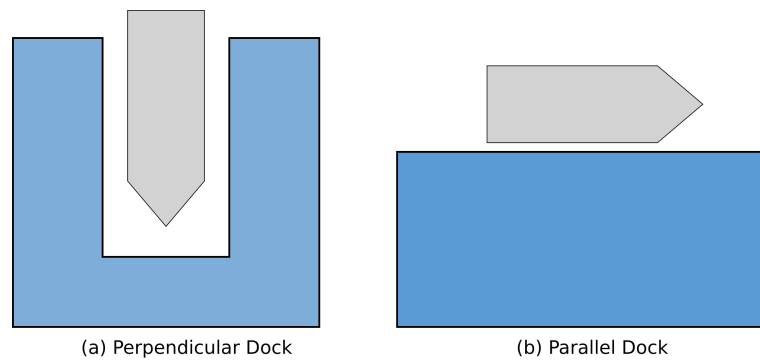


Figure 11. Types of berthing operations.

3.1. Simulation Results

The berthing simulation was performed with the 3DoF model of the vehicle. The environmental disturbances are not considered in the simulation. The simulation results are illustrated in Figures 12–14, where the blue dotted line represents the target path generated by the proposed path planner, and the black triangles indicate the simulated poses of vehicle. The green and red triangles indicate the initial pose and berthing pose.

3.1.1. Case 01

In test case 01, the initial position of the vehicle is very close to the obstacle (see Figure 12a). Firstly, the vehicle performs the reversing maneuvers based on the unberthing planner, as shown in Figure 12b. When the reverse path is completed, it regenerates a collision-free path toward the berthing position. Such a path maintains a safe distance from obstacles. The final berthing operation is guided by the Bézier curve with $L_s = L_e = 10$ m. It achieves a desirable berthing precision ($P_b = 0.58$), with $|d| = 0.013$ m, $|\theta - \theta_d| = 0.007$ rad, $|u| = 0.23$ m/s at the time $t = 109$ s.

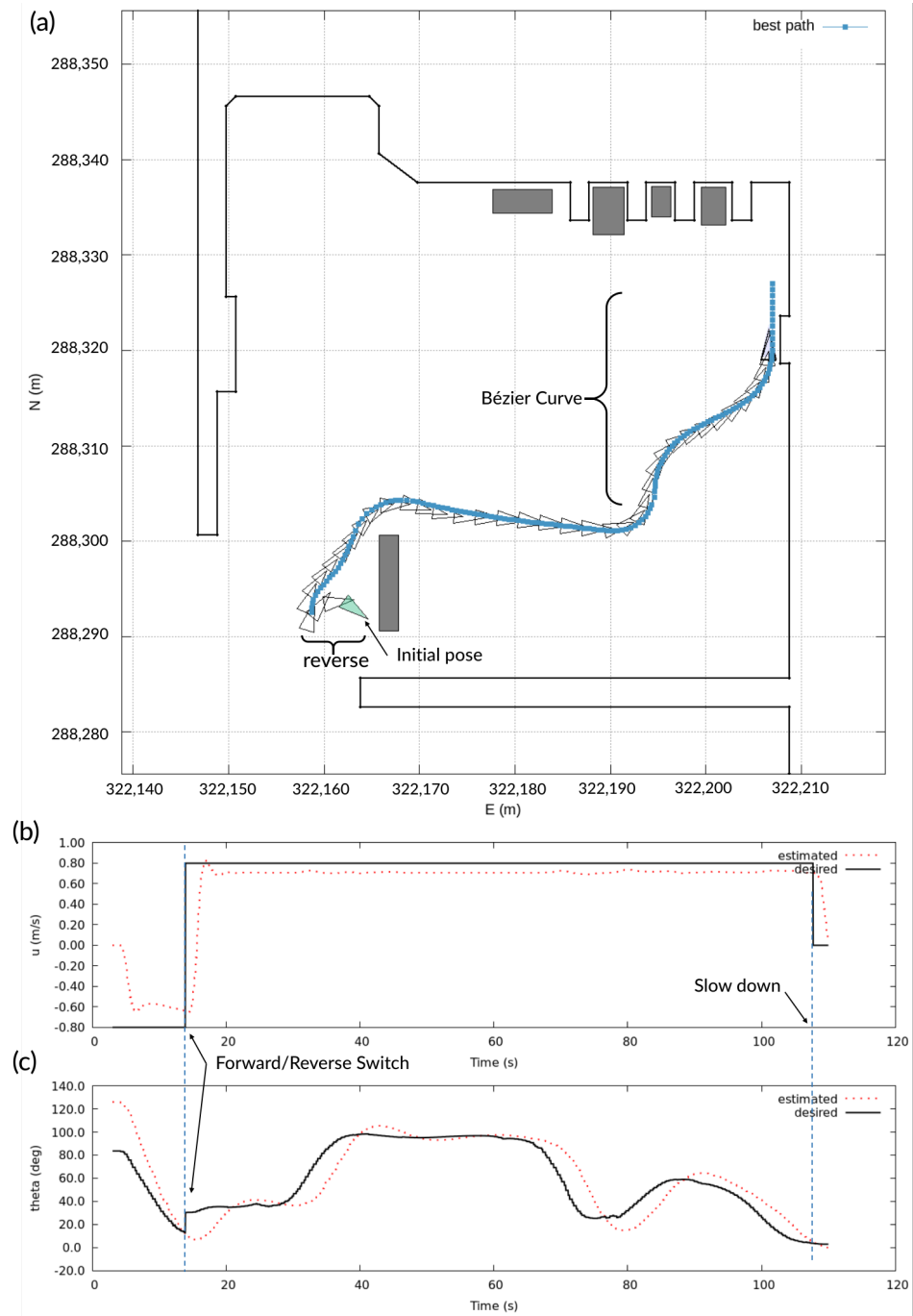


Figure 12. Simulation results of test case 01. (a) shows the planar trajectory, where the blue dotted line represents the target path and the black triangles indicate the simulated poses of vehicle; (b) shows the real-time estimated speed (red dotted line) and target speed (black solid line); (c) shows the real-time estimated heading (red dotted line) and target heading.

3.1.2. Case 02

In the test case 02, the initial vehicle’s direction is almost opposite to the berthing direction (see Figure 13). The vehicle generates a target path, a concatenation of the hybrid A* result and Bézier curve, without switching the direction of motion. This demonstrates the advantage of the Bézier curve. Furthermore, the adaptive pure pursuit method ensures desirable tracking performance, which leads to a good balance in terms of berthing precision and comfortable movement. At the time $t = 88$ s, it achieves a desirable berthing precision ($P_b = 0.45$), with $|d| = 0.002$ m, $|\theta - \theta_d| = 0.011$ rad, $|u| = 0.18$ m/s.

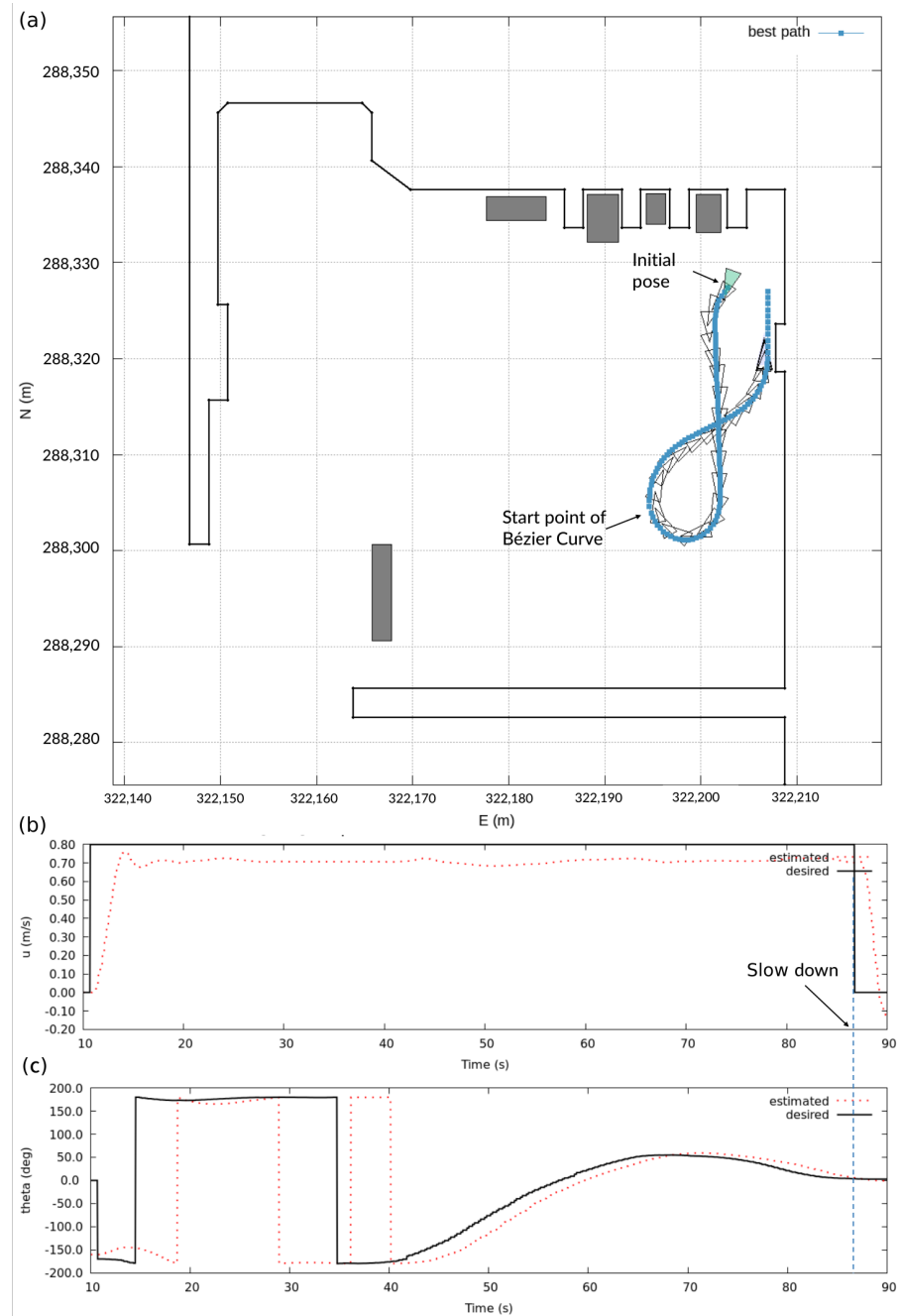


Figure 13. Simulation results of test case 02. (a) shows the planner trajectory, where the blue dotted line represents the target path and the black triangles indicate the simulated poses of vehicle; (b) shows the real-time estimated speed (red dotted line) and target speed (black solid line); (c) shows the real-time estimated heading (red dotted line) and target heading.

3.1.3. Case 03

In the test case 03, the vehicle must generate a feasible path in the narrow channel, whose minimum width is 2.7 m. A box-shaped obstacle is placed in front of the berthing position in order to increase the complexity of the path planning. As shown in Figure 14, the vehicle firstly fails to compute a safe navigation path within the narrow channel. Then, it moves in the reverse direction based on the unberthing planner. When the unberthing path is finished, the vehicle recomputes a smooth path that safely navigates within the narrow channel. After the vehicle crosses the narrow channel, it bypasses the obstacle and keeps a comfortable distance from the obstacle around the corner. At the time $t = 110$ s, it achieves a desirable berthing precision ($P_b = 0.5$), with $|d| = 0.003$ m, $|\theta - \theta_d| = 0.009$ rad, $|u| = 0.21$ m/s.

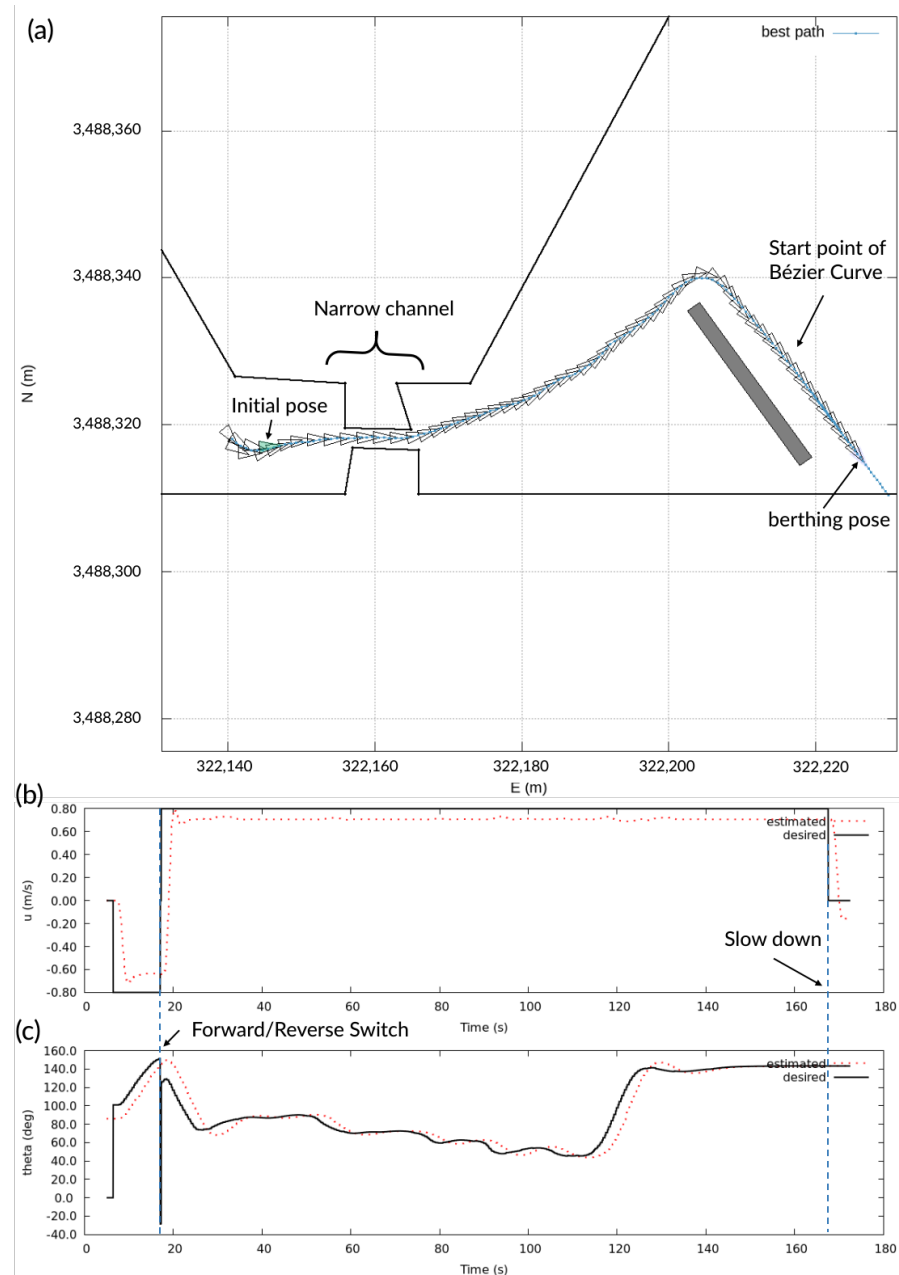


Figure 14. Simulation results of test case 03. (a) shows the planar trajectory, where the blue dotted line represents the target path and the black triangles indicate the simulated poses of vehicle; (b) shows the real-time estimated speed (red dotted line) and target speed (black solid line); (c) shows the real-time estimated heading (red dotted line) and target heading.

3.2. Experimental Results

The experiment was developed on a catamaran (see Figure 2) by integrating the GNSS sensor, thrusters, wireless modem, batteries, etc. The GNSS sensor with the Continuously Operating Reference Station (CORS) provides the positioning, heading and velocity of the vehicle, with a positioning accuracy of 0.05 m and heading accuracy of 0.1 degree. We tested the planning and tracking capabilities under calm waters. The experimental results are illustrated in the figures, where the blue dotted line represents the target path, the GNSS-measured poses of the vehicle are described by black triangles, and the initial pose and berthing pose are represented by green and blue triangles, respectively.

3.2.1. Case 04

The perpendicular dock is set up in test case 04. The initial pose of the vehicle is close to the surrounding obstacles. However, Figure 15 shows that the vehicle generates a smooth path with the positive target speed, as the planner applies a penalty for driving in reverse. For the Bézier curve, the distance between the starting point (A_0) and berthing point (A_3) is 18 m in order to ensure sufficient precision in the tracking error and heading angle, especially when the vehicle begins to enter the dock. At the time $t = 82$ s, it achieves a desirable berthing precision ($P_b = 0.5$), with $|d| = 0.005$ m, $|\theta - \theta_d| = 0.025$ rad, $|u| = 0.08$ m/s.

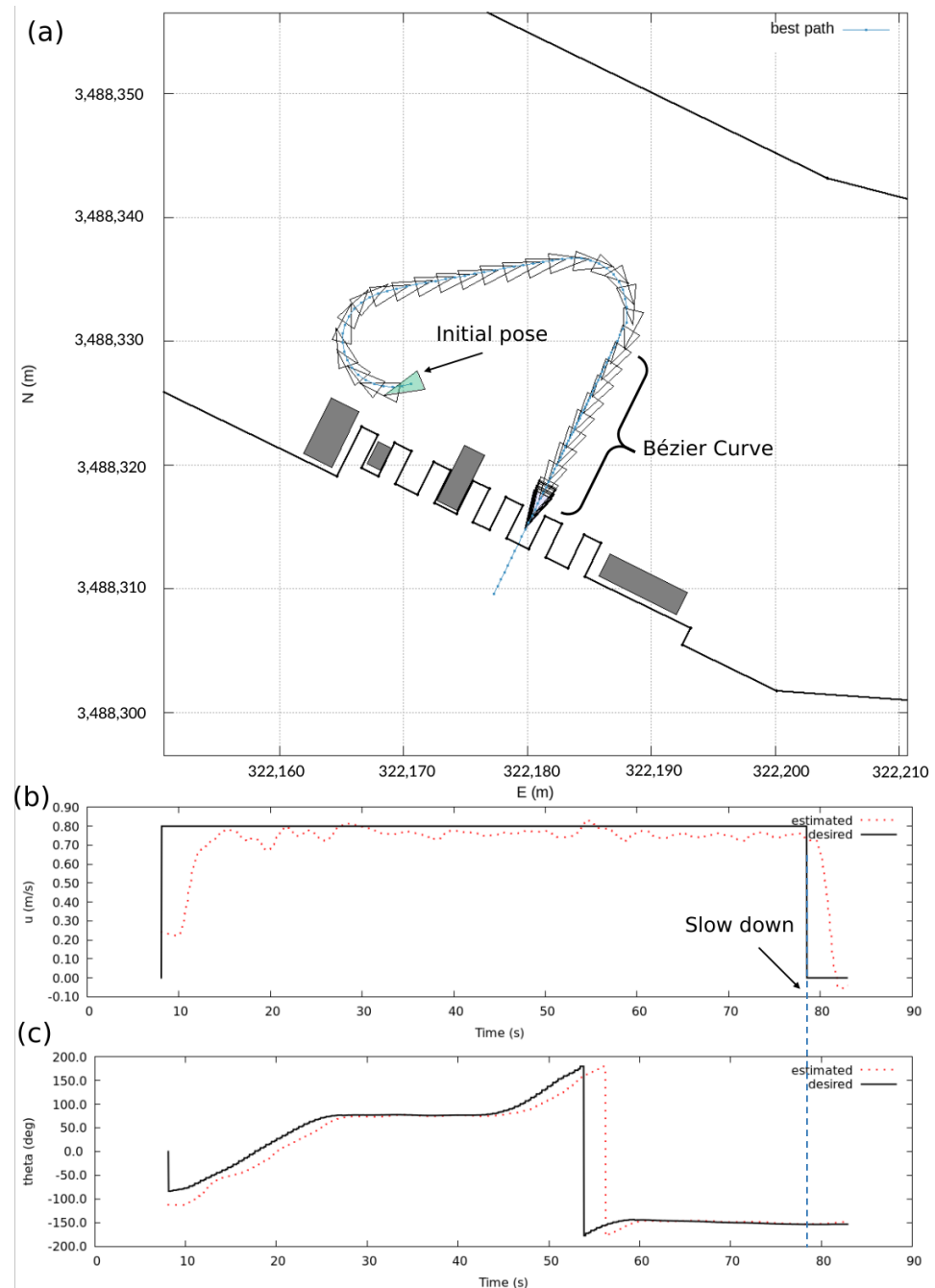


Figure 15. Experimental results of test case 04. (a) shows the planar trajectory, where the blue dotted line represents the target path and the black triangles indicate the simulated poses of vehicle; (b) shows the real-time estimated speed (red dotted line) and target speed (black solid line); (c) shows the real-time estimated heading (red dotted line) and target heading.

3.2.2. Case 05

Figure 16 shows the results of test cast 05. Firstly, the vehicle is required to travel in reverse, as the forward movement has a high risk of collision with obstacles. There is a large tracking error between the vehicle and target path at the moment when the vehicle experiences a change in the direction of motion. These errors become small as the vehicle travels toward the berthing position. At the time $t = 114$ s, it achieves a desirable berthing precision ($P_b = 0.8$), with $|d| = 0.08$ m, $|\theta - \theta_d| = 0.002$ rad, $|u| = 0.07$ m/s.

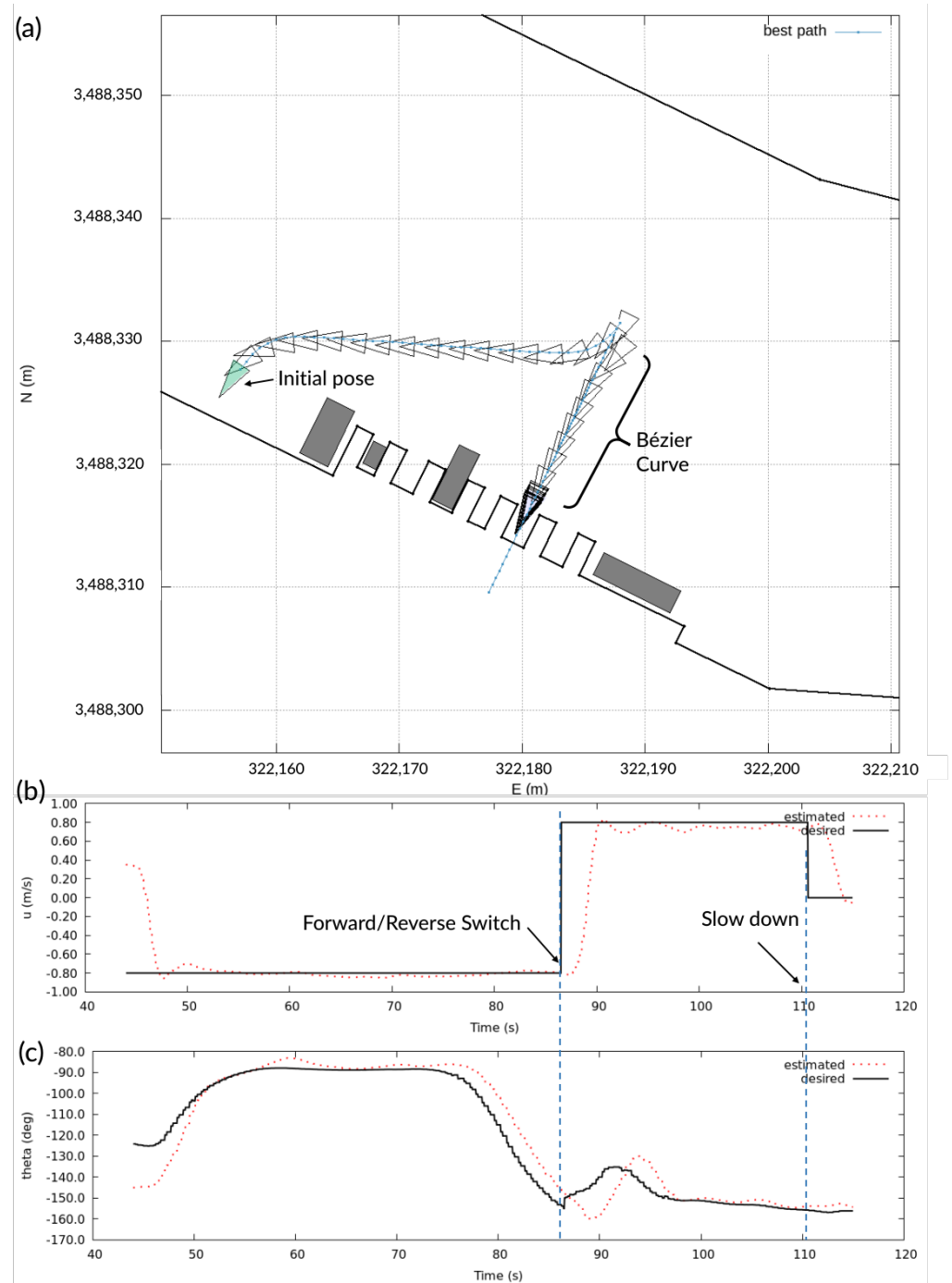


Figure 16. Experimental results of test case 05. (a) shows the planar trajectory, where the blue dotted line represents the target path and the black triangles indicate the simulated poses of vehicle; (b) shows the real-time estimated speed (red dotted line) and target speed (black solid line); (c) shows the real-time estimated heading (red dotted line) and target heading.

3.2.3. Case 06

Figure 17 shows the successful unberthing and berthing operation in the test case 06. At first, it fails to compute a feasible path using the hybrid A* search method. Therefore, the vehicle generates the unberthing path with the negative target speed at the time duration from 22 s to 33 s (see Figure 17b). Once the unberthing operation is finished, the planning module succeeds at generating a smooth path using the hybrid A* search. At the time $t = 119$ s, it achieves a desirable berthing precision ($P_b = 0.5$), with $|d| = 0.05$ m, $|\theta - \theta_d| = 0.002$ rad, $|u| = 0.06$ m/s.

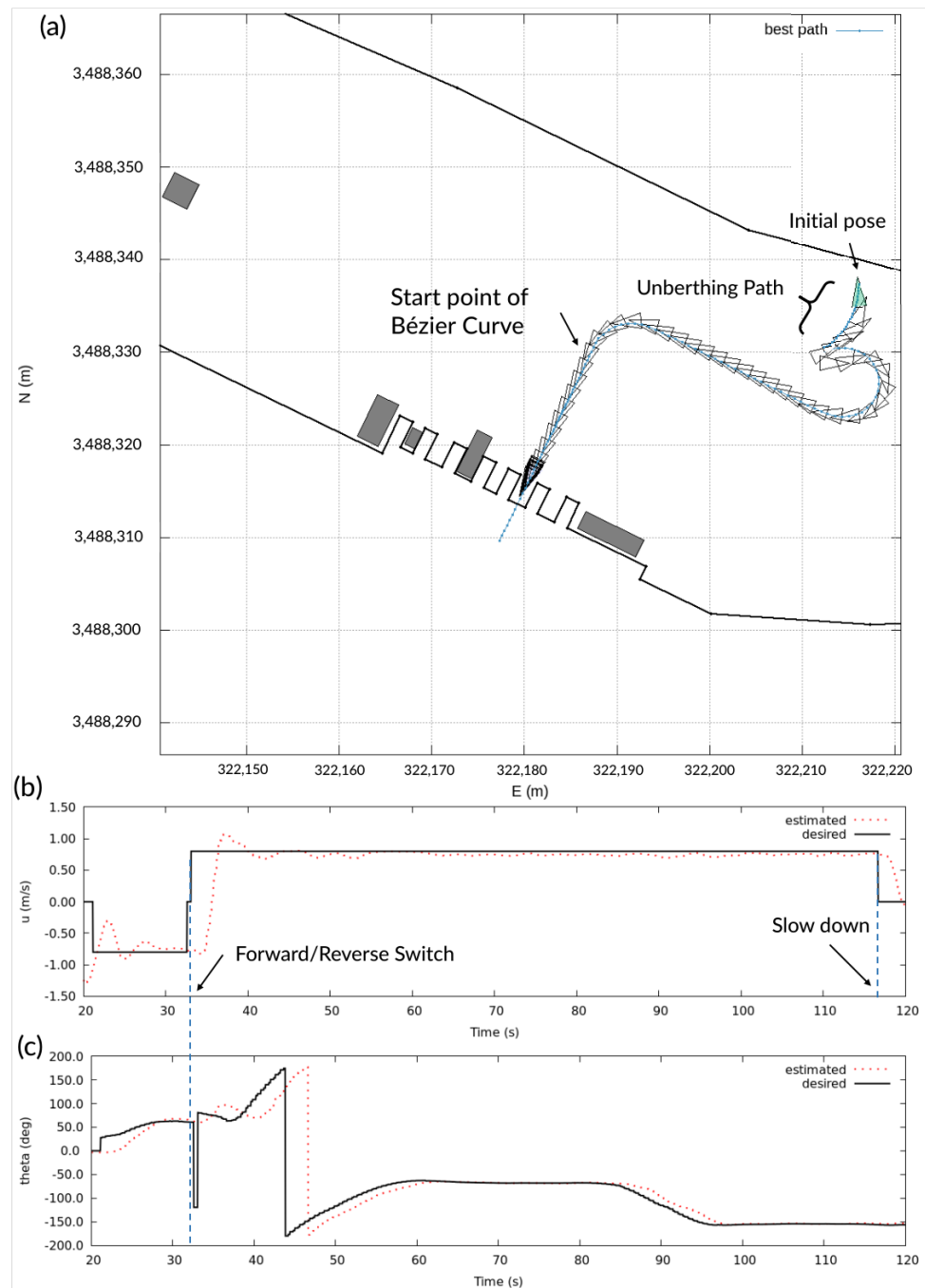


Figure 17. Experimental results of test case 06. (a) shows the planar trajectory, where the blue dotted line represents the target path and the black triangles indicate the simulated poses of vehicle; (b) shows the real-time estimated speed (red dotted line) and target speed (black solid line); (c) shows the real-time estimated heading (red dotted line) and target heading.

4. Conclusions

We propose a graph search-based path-planning method for the automatic ship berthing operation. This process can be divided into two phases: transit and berthing. The transit phase utilizes the hybrid A* search planner to generate a smooth and collision-free path. The efficiency of such a search can be improved using the adaptive costmap and SSSP based on the GVD of the surrounding obstacles. The berthing phase applies the cubic Bézier curve for the final berthing operation. It enables safer operation through gentle and precise control. The simulation and experimental results successfully demonstrate the performance of the proposed path-planning strategy in terms of its safety and berthing precision. Extensive experimental research should be conducted on the application of simultaneous localization and mapping (SLAM) in the automatic berthing system, which will increase the levels of autonomy.

Author Contributions: Writing—original draft, X.L.; Writing—review & editing, Z.Y.; Project administration, Z.H.; Funding acquisition, W.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This paper is partly supported by the National Key RD Program of China (2022ZD0119900), Shanghai Science and Technology program (22015810300), Hainan Province Science and Technology Special Fund (ZDYF2021GXJS041), and the National Natural Science Foundation of China (U2141234).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is unavailable due to privacy or ethical re-strictions.

Conflicts of Interest: The authors declare no conflicts of interest.

References

- Ohtsu, K.; Takai, T.; Yoshihisa, H. A Fully Automatic Berthing Test Using the Training Ship Shioji Maru. *J. Navig.* **1991**, *44*, 213–223. [[CrossRef](#)]
- Djouani, K.; Hamam, Y. Minimum time-energy trajectory planning for automatic ship berthing. *IEEE J. Ocean. Eng.* **1995**, *20*, 4–12. [[CrossRef](#)]
- Maki, A.; Sakamoto, N.; Akimoto, Y.; Nishikawa, H.; Umeda, N. Application of optimal control theory based on the evolution strategy (CMA-ES) to automatic berthing. *J. Mar. Sci. Technol.* **2020**, *25*, 221–233. [[CrossRef](#)]
- Mizuno, N.; Uchida, Y.; Okazaki, T. Quasi Real-Time Optimal Control Scheme for Automatic Berthing. *IFAC-PapersOnLine* **2015**, *48*, 305–312. [[CrossRef](#)]
- Ahmed, Y.A.; Hasegawa, K. Automatic Ship Berthing using Artificial Neural Network Based on Virtual Window Concept in Wind Condition. *IFAC Proc. Vol.* **2012**, *45*, 286–291. [[CrossRef](#)]
- Ahmed, Y.A.; Hasegawa, K. Automatic ship berthing using artificial neural network trained by consistent teaching data using nonlinear programming method. *Eng. Appl. Artif. Intell.* **2013**, *26*, 2287–2304. [[CrossRef](#)]
- Wang, S.; Sun, Z.; Yuan, Q.; Sun, Z.; Wu, Z.; Hsieh, T.H. Autonomous piloting and berthing based on Long Short Time Memory neural networks and nonlinear model predictive control algorithm. *Ocean Eng.* **2022**, *264*, 112269. [[CrossRef](#)]
- Shuai, Y.; Li, G.; Cheng, X.; Skulstad, R.; Xu, J.; Liu, H.; Zhang, H. An efficient neural-network based approach to automatic ship docking. *Ocean Eng.* **2019**, *191*, 106514. [[CrossRef](#)]
- Skulstad, R.; Li, G.; Fossen, T.I.; Vik, B.; Zhang, H. A Hybrid Approach to Motion Prediction for Ship Docking—Integration of a Neural Network Model Into the Ship Dynamic Model. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–11. [[CrossRef](#)]
- Martinsen, A.B.; Lekkas, A.M.; Gros, S. Autonomous docking using direct optimal control. *IFAC-PapersOnLine* **2019**, *52*, 97–102. [[CrossRef](#)]
- Miyauchi, Y.; Sawada, R.; Akimoto, Y.; Umeda, N.; Maki, A. Optimization on planning of trajectory and control of autonomous berthing and unberthing for the realistic port geometry. *Ocean Eng.* **2022**, *245*, 110390. [[CrossRef](#)]
- Martinsen, A.B.; Bitar, G.; Lekkas, A.M.; Gros, S. Optimization-Based Automatic Docking and Berthing of ASVs Using Exteroceptive Sensors: Theory and Experiments. *IEEE Access* **2020**, *8*, 204974–204986. [[CrossRef](#)]
- Shimizu, S.; Nishihara, K.; Miyauchi, Y.; Wakita, K.; Suyama, R.; Maki, A.; Shirakawa, S. Automatic berthing using supervised learning and reinforcement learning. *Ocean Eng.* **2022**, *265*, 112553. [[CrossRef](#)]
- González, D.; Pérez, J.; Milanés, V.; Nashashibi, F. A Review of Motion Planning Techniques for Automated Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 1135–1145. [[CrossRef](#)]
- Dolgov, D.A. *Practical Search Techniques in Path Planning for Autonomous Driving*; American Association for Artificial Intelligence: Washington, DC, USA, 2008.

16. Dolgov, D.; Thrun, S.; Montemerlo, M.; Diebel, J. Path Planning for Autonomous Driving in Unknown Environments. In *Experimental Robotics*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 55–64.
17. Bitar, G.; Martinsen, A.B.; Lekkas, A.M.; Breivik, M. Two-Stage Optimized Trajectory Planning for ASVs Under Polygonal Obstacle Constraints: Theory and Experiments. *IEEE Access* **2020**, *8*, 199953–199969. [[CrossRef](#)]
18. Reeds, J.; Shepp, L. Optimal paths for a car that goes both forwards and backwards. *Pac. J. Math.* **1990**, *145*, 367–393. [[CrossRef](#)]
19. Fossen, T.I. *Handbook of Marine Craft Hydrodynamics and Motion Control*; John Wiley & Sons: Hoboken, NJ, USA, 2011.
20. Wirtensohn, S.; Reuter, J.; Blaich, M.; Schuster, M.; Hamburger, O. Modelling and identification of a twin hull-based autonomous surface craft. In Proceedings of the 2013 18th International Conference on Methods and Models in Automation and Robotics (MMAR), Miedzyzdroje, Poland, 26–29 August 2013; pp. 121–126. [[CrossRef](#)]
21. Hu, Z.; Yang, Z.; Zhang, W. Path planning for auto docking of underactuated ships based on Bezier curve and hybrid A* search algorithm. *Chin. J. Ship Res.* **2024**, *19*, 220–229. [[CrossRef](#)]
22. Fortune, S. A sweepline algorithm for Voronoi diagrams. In Proceedings of the Second Annual Symposium on Computational Geometry, Yorktown Heights, NY, USA, 2–4 June 1986; pp. 313–322.
23. Masehian, E.; Amin-Naseri, M. A voronoi diagram-visibility graph-potential field compound algorithm for robot path planning. *J. Robot. Syst.* **2004**, *21*, 275–300. [[CrossRef](#)]
24. LaValle, S.M. *Planning Algorithms*; Cambridge University Press: Cambridge, UK, 2006.
25. Sucan, I.A.; Moll, M.; Kavraki, L.E. The open motion planning library. *IEEE Robot. Autom. Mag.* **2012**, *19*, 72–82. [[CrossRef](#)]
26. Marcucci, T.; Nobel, P.; Tedrake, R.; Boyd, S. Fast Path Planning Through Large Collections of Safe Boxes. *arXiv* **2024**, arXiv:2305.01072.

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