Ocean Engineering 84 (2014) 37–44

Contents lists available at ScienceDirect

Ocean Engineering

journal homepage: www.elsevier.com/locate/oceaneng

Angular rate-constrained path planning algorithm for unmanned surface vehicles

Hanguen Kim [a], Donghoon Kim [b], Jae-Uk Shin [b], Hyongjin Kim [b], Hyun Myung [a,b,*]

[a] Department of Civil and Environmental Engineering, KAIST, 291 Daehak-ro (373-1 Guseong-dong), Yuseong-gu, Daejeon 305-701, Republic of Korea
[b] Robotics Program, KAIST, 291 Daehak-ro (373-1 Guseong-dong), Yuseong-gu, Daejeon 305-701, Republic of Korea

ARTICLE INFO

Article history:
Received 23 May 2013
Accepted 29 March 2014
Available online 17 April 2014

Keywords:
Path planning
Unmanned surface vehicle
Heading angle
Angular rate constraint

ABSTRACT

This study suggests a new approach based on Theta* algorithm to create paths in real-time, considering both angular rate (yaw rate) and heading angle of unmanned surface vehicles (USVs). Among the various path planning algorithms, a grid map-based path planning algorithm has frequently been employed for maintaining the control stability of nonholonomic USVs. This algorithm is powerful in that it generates a path with the fastest computation time. Most path planning algorithms for the ocean environment, however, have only been performed on an Euclidean group *SE*(2) grid map with no considerations of the vehicle performance. Other path planning algorithms for an *SE*(3) grid map generate resultant paths with consideration of vehicle heading, but most of them do not operate in real-time. Moreover, these algorithms always generate constant angle at each orientation node, and the size of graph must be expanded to provide finer angular resolution. To solve these problems, the angular rate-constrained Theta* (ARC-Theta*) is proposed to create paths in real-time with considerations of vehicle performance on the *SE*(2) weighted occupancy grid map. The performance of this proposed algorithm is verified with simulations and experiments. The results showed that the ARC-Theta* algorithm is effective for global path planning of USVs.

1. Introduction

As navigation technologies have advanced, the level of autonomy of unmanned vehicles (robots) has also progressed. Similarly, various unmanned surface vehicles (USVs) were introduced progressively (Manley, 2008; Kim et al., 2012a). Recently, advances in navigation technologies and increases in robot intelligence have increased the potential for greater autonomy. In accordance with the current phase of autonomy, unmanned robots are designed to perform missions and navigate by themselves (Huang et al., 2005). In this sense, a considerable amount of research has already been done, particularly in the field of path planning that seeks suitable paths for unmanned vehicles. The path planning algorithm for nonholonomic systems is able to help to maintain the stability of the control system. It can also provide a solution for overcoming stabilization challenges in control of nonholonomic systems, which is described in Brockett (1983) and Sun et al. (2001). Among the various path planning algorithms, a grid map-based path planning algorithm has frequently been employed for this purpose and is one of the best known techniques. The grid map-based path planning algorithm is powerful in that it generates a path with the fastest computation time. The grid map-based path planning algorithm started from Dijkstra (1959), who described about finding the shortest path between any two nodes. Then A* algorithm was introduced by extending the Dijkstra's algorithm (Hart et al., 1968). There are currently a lot of researches aimed at finding a substitute for the conventional A* algorithm. Nash et al. (2007) compared several algorithms, i.e., A*, A* with post-smoothing (Rabin, 2000), Field D* (Ferguson and Stentz, 2007), and Theta*. Nash et al. (2007) found that the Theta* algorithm performed best in terms of path length and computation time. Theta*, a variant of the A* algorithm allows a search in any direction within a specified distance. Thus, Theta* is appropriate for solving the ship navigation problem where real-time operation is of great importance. Because the conventional Theta* does not work in a weighted occupancy grid map, Choi and Yu (2011) suggested a modified Theta* algorithm that can create optimal paths on a weighted occupancy grid map to make vehicles follow way-points well. Typically, the conventional grid map-based path planning algorithm focuses on finding the shortest path and only considers obstacles to be avoided (Hart et al., 1968; Stentz, 1994).

* Corresponding author at: Department of Civil and Environmental Engineering, KAIST, 291 Daehak-ro (373-1 Guseong-dong), Yuseong-gu, Daejeon 305-701, Republic of Korea.
E-mail addresses: sskhk05@kaist.ac.kr (H. Kim), dh8607@kaist.ac.kr (D. Kim), jacksju@kaist.ac.kr (J.-U. Shin), hjkim86@kaist.ac.kr (H. Kim), hmyung@kaist.ac.kr (H. Myung).

http://dx.doi.org/10.1016/j.oceaneng.2014.03.034
0029-8018/

In many real-world applications, however, the path of a vehicle should consider a heading angle and angular rate (yaw rate). Since the conventional grid map-based path planning algorithm does not usually consider the vehicle heading angle at its start and goal positions as shown in Fig. 1a, the algorithm not only creates an unrealistic path but also might put the vehicle in dangerous situations. Also, if the angular rate is not considered as shown in Fig. 1b, the obstacle avoidance function may not work. Moreover, since unmanned surface vehicles (USVs) usually follow a chosen path at high speed, the path must be re-planned in real-time with accommodation for unexpected obstacles. Thus, a real-time path planning algorithm is needed.

A lot of research is being done to solve these problems. In one approach to create a practical path in a sailing area, grid map is organized as a weighted occupancy map using the approximate cell decomposition method (Latombe, 1991; Thrun et al., 2005). The weighted occupancy grid map is composed of free nodes, obstacle nodes, and other occupancy nodes. The other occupancy nodes are used to calculate the cost when the path planning algorithm generates the paths. This can simply and effectively create paths that various vehicles can follow. However, because it only contains information related to topography, it cannot provide the vehicle heading angle if a path planning algorithm based on a special Euclidean group $SE(2)$ grid map is applied. To provide a vehicle heading angle, other path planning algorithms based on an $SE(3)$ grid map generate resultant paths with consideration of the vehicle heading angle, but most of them do not operate in real-time. Moreover, these algorithms always generate a constant angle at each orientation node, and the size of graph must be expanded to provide finer angular resolution (Kim et al., 2012b). Laumond (1987) proposed a smooth trajectory planning method for a car-like mobile robot. This algorithm extended the regions of obstacles by using the set of positions at a distance reachable from the center of the robot. Then, smooth trajectories were calculated with consideration of the car's kinematics. However, this algorithm cannot generate trajectories in real-time due to its high computational load. Yong and Barth (2006) also proposed a shortest path planning algorithm for Dubin's nonholonomic robot. This algorithm creates center points of a pair of accommodation circles which define the left and right side paths with minimum turning radius. After creating the accommodation circles, the path is generated using the artificial potential field method. As a result, it can reflect the 3D kinematic state space of the vehicle. However, this assumes that there are no obstacles between the start and goal points, because this algorithm employs the artificial potential field method to find a path and thus, it cannot solve local minimum problems. In ocean environments, ship navigation must give great consideration to the maneuverability of the vehicle, which is usually a highly under-actuated system. Numerous studies about ocean navigation have been performed, but most have been focused on collision avoidance rather than the path planning problem (Tam et al., 2009). More advanced approaches to path planning such as genetic algorithms or fuzzy set theory have recently been reported by Szlapczynski (2006) and Patnaik et al. (2006). However, they have not given much consideration to the actual turning capability of USVs, which is critical not only for path planning but also for maneuvering of the vehicle as well. Most underwater vehicles are often assumed to be highly maneuverable, so that there are no explicit constraints on their turning ability, but this is not true for most surface vehicles (Kruger et al., 2007; Garau et al., 2005; Witt and Dunbabin, 2008; Soulignac, 2011; Isern-González et al., 2011; Prasanth Kumar et al., 2005; Yang et al., 2011).

This study presents a novel grid map-based path planning algorithm for generation of realistic paths with accommodation for both heading angle and angular rate. In Section 2, we present the newly devised approach to grid map-based path planning. The effectiveness of the proposed algorithm was verified by simulations and experiments, and by comparison with conventional algorithms. The results from these are presented in Section 3.

2. Angular rate-constrained theta*

2.1. Basic theta*

The newly devised algorithm in this study is based on Theta* which is similar to A*. The only difference is the method used to select the parent node. The parent node is the previous search node that is connected to a current search node. The A* selects a parent node based on the previous search node which is adjacent to the current search node. The basic Theta*, however, selects a parent node based on checking the line of sight (LOS) in $SE(2)$. Its main algorithm is described in Algorithm 1. When connecting between the current search node and its parent node, Theta* always checks the LOS, enabling it to search its neighbors in any direction within a specified distance. In Algorithm 1, the cell s is defined as $s=[x,y]$ and the cells s_{start} and s_{goal} are used as the start and goal nodes, respectively. *Open* is a *stable priority queue* that checks whether a node has been already visited. The *stable priority queue* performs *Push* operation in accordance with the cost associated with each queue element. The LOS function, *LineOfSight*() checks whether or

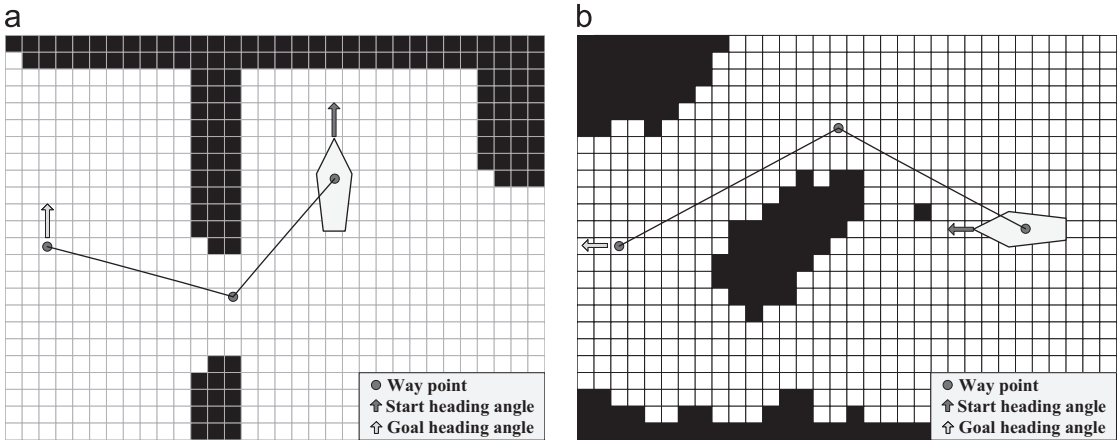

Fig. 1. Example paths un-followable because of (a) the heading angle at start and goal point, and (b) which might lead the vehicle to collide with an obstacle due to constraints of the angular rate.

not an obstacle exists along the path using Bresenham's line algorithm (see Bresenham, 1965, Algorithm 2). Using the Bresenham's line algorithm, the LOS function returns the cost between $s_{neighbor}$ and s when no obstacle is found on the line connecting $s_{neighbor}$ and s; otherwise, it returns false. The cost from node s to $s_{neighbor}$ is simply calculated using the Euclidean distance, $d(s_{neighbor}, s)$. For this reason, Theta* generates a shorter path than A* except the case when the LOS path is blocked by an obstacle. In this case the near neighbor node is selected as the new parent node in the same manner as in A*. In this paper, we use the center-node representation for a grid to avoid corner-node representation problem reported in Choi and Yu (2011).

Algorithm 1. Basic Theta*(s_{start}, s_{goal}).

```
1:  s_start.Parent ← s_start
2:  while open ≠ ∅ do
3:    s ← open.Pop()
4:    if s = s_goal then
5:      return "path found"
6:    end if
7:    for each s_neighbor do
8:      if LineOfSight(s_neighbor, s) then
9:        s_neighbor.parent ← parent
10:       open.Push(s_neighbor)
11:   end if
12:   end for
13: end while
14: return "no path found"
```

Algorithm 2. LineOfSight($s_{neighbor}, s$).

```
1:  for each s adjacent to s_neighbor do
2:    if s_current = obstacle then
3:      return "obstacles exist"
4:    end if
5:  end for
6:  return d(s_neighbor, s)
```

2.2. Angular rate-constrained Theta* algorithm

One of the key concepts used in the newly devised algorithm is to restrict the range of LOS and angular rate by accommodating for the turning ability of the vehicle. Thus, the angular rate of each LOS is calculated when the node is expanded in Theta*. The angular rate r is defined as the ratio of speed V to turning radius R for a given vehicle:

$$r = \frac{V}{R}. \tag{1}$$

However, it is difficult to precisely predict the turning radius of a marine surface vessel due to the dynamic and unstable characteristics of the ocean environment. A Maritime Security Committee in IMO (International Maritime Organization) established a regulation (MSC.137(76)) called Standards for Ship Maneuverability (Maritime Security Committee, 2002). This document limits the minimum turning radius of a vessel to 5 times the ship length. Fig. 2 shows the example of the modified LOS named as Angular Rate-Constrained LOS (ARC-LOS) that reflects the regulation. The pseudo code of the ARC-LOS process is shown in Algorithm 3.

Algorithm 3. ARC-LineOfSight($s_{neighbor}, s$).

```
1:  current orientation ← arctan((s_neighbor.y − s.y) / (s_neighbor.x − s.x))
2:  if (current orientation − s.orientation)*V/d(s_neighbor, s) > r_max
    then
3:    return "cannot connect s_neighbor"
4:  end if
5:  for each s adjacent to s_neighbor do
6:    if !IsWalkable(s, s_neighbor) then
7:      return "obstacles exist"
8:    else
9:      costsum += s_current.mapcost
10:   end if
11: end for
12: s_neighbor.orientation ← current orientation − s.orientation
13: return costsum/d(s_neighbor, s)
```

First, in contrast to the basic Theta* algorithm, *s.orientation* variable is added to record the heading angle of the vehicle in *SE* (2). The r_{max} is the maximum angular rate of a vehicle. The ARC-LOS function always checks whether or not the current angular rate is greater than the maximum angular rate. If current angular rate is greater than the maximum angular rate, the ARC-LOS function returns a false state. A function, *IsWalkable*, checks the vehicle's occupancy states around the current search nodes on the weighted occupancy grid map. The occupancy states are calculated according to the vehicle orientation and size. The function

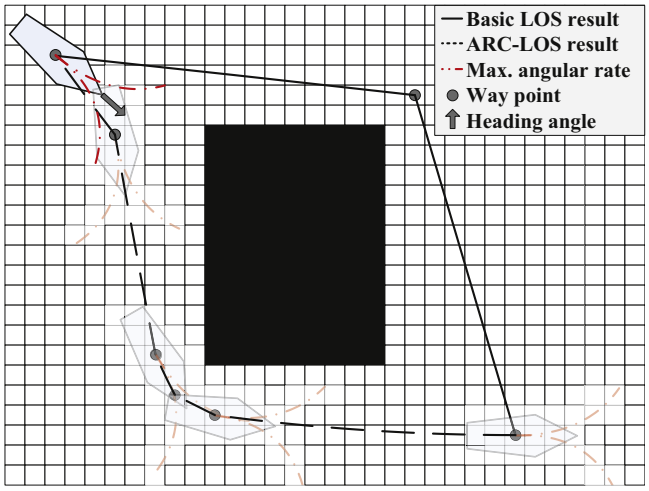

Fig. 2. Example of ARC-LOS result. The ARC-LOS checks the possibility of sailing by checking obstacles and the maximum angular rate.

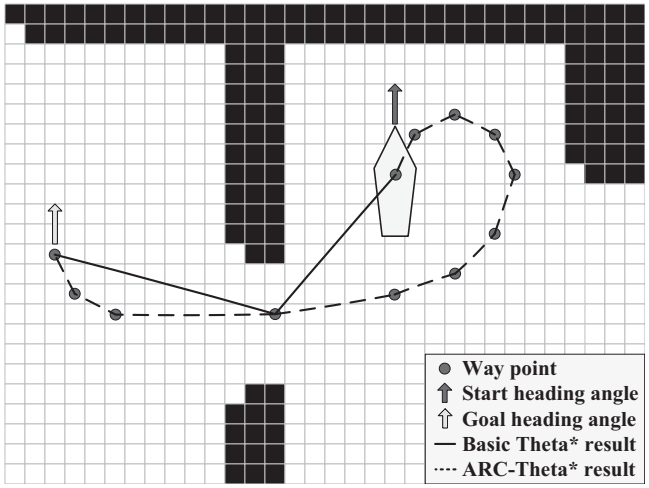

Fig. 3. Example path generated by using ARC-Theta* algorithm. The ARC-Theta* generates the path which satisfies the vehicle's angular rate and heading angle.

IsWalkable returns true when the vehicle can operate on the line connecting the $s_{neighbor}$ and s; otherwise, it returns false. However, if the LOS function considers too many occupancy states, its performance may diminish. Thus, performance can be improved by using a pre-calculated occupancy table. If LOS is ensured, the ARC-LOS function returns the average occupancy cost according to the distance. By using this method, way-points with atypical angles can be created, unlike with the existing $SE(3)$ path planning algorithm. Even so, the method cannot satisfy the arrival heading angle at the goal point because the ARC-LOS algorithm only verifies LOS between current and previous nodes (explored node and its parent node). To mitigate this problem, Dubin's Curve algorithm is applied at the start point and the goal point. Dubin's Curve algorithm is used to get an optimal path under the rules that the vehicle turns left or right when the angular rate of the vehicle is given, with the assumption that the vehicle cannot reverse (LaValle, 2006). To apply Dubin's Curve algorithm at the start and goal points, at first, a path from the start

Table 1
Simulation setup.

Property (unit)	Value
Grid map size (pixel)	500×250
Pixel resolution (m/pixel)	1.0
Ship length (m)	15.0
Ship beam (m)	4.5
r_{max} (rad/s)	0.39
v_{max} (m/s)	20.0

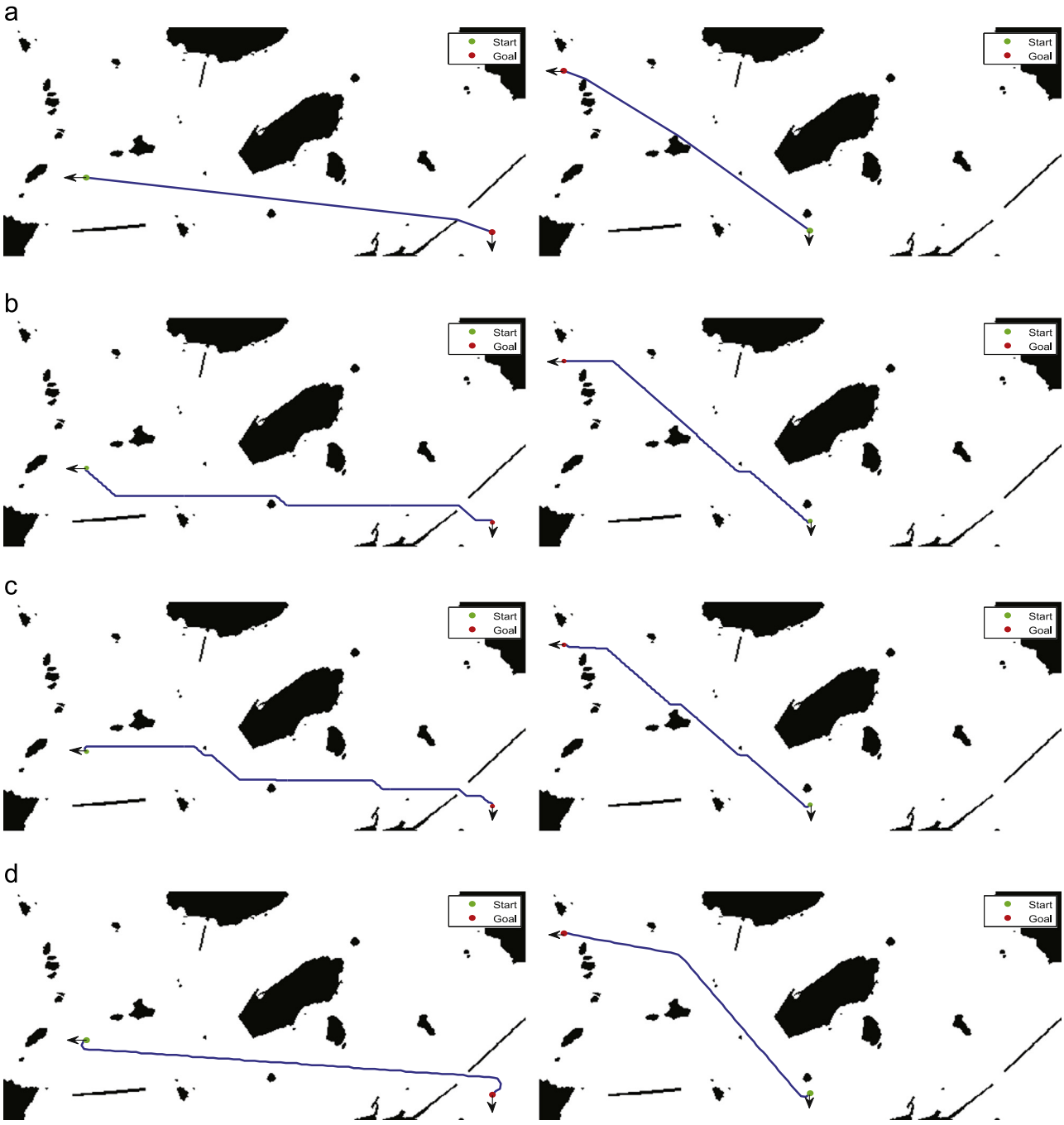

Fig. 4. Simulation results for (a) Theta*, (b) 3D A*(45° resolution), (c) 3D A*(22.5° resolution), and (d) ARC-Theta*. The left column figures have 270° of start heading and 180° of goal heading angle and the right column figures have 180° of start heading and 270° of goal heading angle. (a) Results of Theta*. The heading angle is not considered, (b) Results of 3D A* with 45° resolution. These do not satisfy the angular rate constraint of the USV, (c) Results of 3D A* with 22.5° resolution. These show the step-like path compared to the proposed algorithm and (d) Results of ARC-Theta*. In comparison to the 3D A*, these have shorter paths while satisfying the vehicle's angular rate constraint.

point to the goal point is created by Theta* (with the ARC-LOS function). Then additional way-points are calculated using Dubin's Curve algorithm. This is possible because Theta* creates way-points using LOS so that the LOS between any two contiguous way-points is ensured. Thus, the influences of obstacles on the path can be avoided by creating Dubin's Curve except in cases where the start angle or goal angle do not satisfy the LOS condition. If obstacles exist on the path generated with maximum angular rate at the start and goal points, the problem can be solved by decreasing the angular rate and re-calculating Dubin's Curve. Fig. 3 shows an example by comparing the resultant paths of the basic Theta* and the proposed algorithm. The proposed algorithm generates the path which satisfies the vehicle's performance. The pseudo code of the final version of the path planning algorithm is shown in Algorithm 4. A function, *CreateDubinsCurves*, generates the Dubin's Curve at the start point and goal point after applying Theta* with the ARC-LOS function. Last, the Angular Rate-Constrained Theta* (ARC-Theta*) algorithm generates a path which satisfies the vehicle's performance condition on an *SE*(2) weighted occupancy grid map thanks to ARC-LOS and Dubin's Curve algorithm. These paths are created quickly with more flexibility for the orientation of the vehicles compared to existing *SE*(3) grid map-based path planning algorithms.

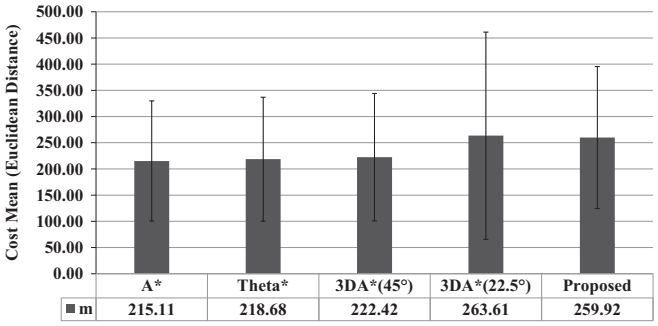

Fig. 5. Compared average costs of various path planning algorithms. Compared with 3D A* with 22.5° resolution, the proposed algorithm generates a shorter path while satisfying the vehicle's performance. The interval on each bar denotes the standard deviation of the cost.

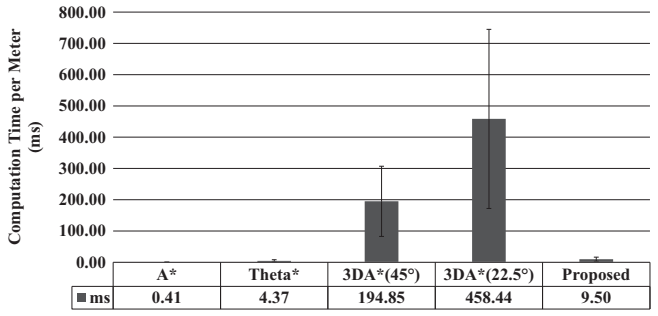

Fig. 6. Average computation time per meter (unit traveling distance) for various path planning algorithms. Considering the speed of the USV (20 m/s in this simulation), the proposed algorithm generates a path within 1 s. Therefore, the proposed algorithm is possible to generate a path in real-time. The interval on each bar denotes the standard deviation of the computation time per meter.

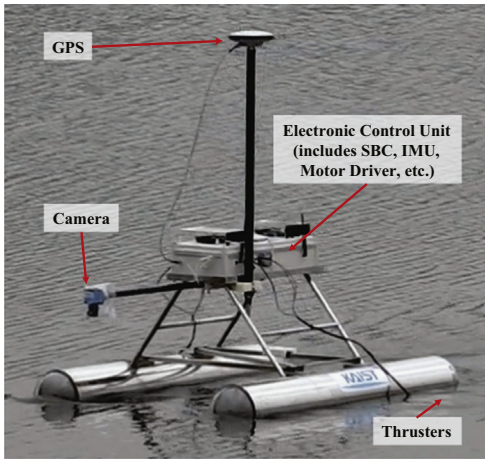

Fig. 7. The JEROS prototype. This USV has the real-time GNC (guidance, navigation and control) system.

Algorithm 4. ARC-Theta*(s_{start}, s_{goal}).

```
Basic Theta*(s_start, s_goal)
with ARC LineOfSight(s_neighbor, s)
if CreateDubinsCurves((s_start, s_goal)) then
    return "path found"
else
    return "no path found"
end if
```

3. Simulations and experiments

3.1. Simulation results

The proposed algorithm was simulated using C++ and OpenCV. All simulations were performed on Intel i5 2.80 GHz quad core CPU with 8 GB RAM. In these simulations the proposed algorithm was compared with basic Theta* and 3D (x, y, θ) A*. The 3D A* extends the structure of the graph by considering the grid resolution per rotation angle of the vehicle (Kim et al., 2012b). The structure of the graph consists of the position node as a reference node and the orientation nodes that belong to the position node. The orientation node can set only specific angles for the connection between the nodes. Therefore, the 3D A* can consider the approximate angular rate of the vehicle by limiting the orientation node. The newly proposed algorithm and other conventional algorithms were simulated on a random geometry map using the Monte Carlo method. The size of random geometry map was set to 500 by 250 pixels grid map which has 1.0 m resolution each pixel, and simulations were performed 2000 times for each different start and goal states. The simulation setup was based on a small USV, and the specifications are listed in Table 1. We assume that the USV is hull craft type ship in the fleet class (U.S. Navy, 2007). This type of USV has a fast sailing speed of about 10–20 m/s, and the usual purpose of this ship is surveillance or delivery of some objects. Since the form of USV is a *rigid-hulled inflatable* type boat, which usually has a small angular rate. For these reason, the maximum angular rate for these simulations is set to 0.39 rad/s, and the maximum vehicle speed is set to 20.0 m/s. In this paper, however, the angular rate and the maximum vehicle speed were not considered in the basic Theta*. In the

Table 2
Experimental setup.

Property (unit)	Value
Ship length (m)	1.5
Ship beam (m)	1.1
Turning radius (m)	1.0
v_{max} (m/s)	1.0
r_{max} (rad/s)	1.0
Tracking radius (m)	3.0

case of 3D A*, it partially satisfies the maximum angular rate by limiting the orientation node. Therefore, the basic Theta* was performed on the basic *SE*(2) grid map, and 3D A* was performed on *SE*(3) grid map with the searching angle resolution of 45° and 22.5°, respectively. These algorithms were not performed on the weighted occupancy grid map. Fig. 4 shows the results for Theta*, 3D A* (45° and 22.5°), and the newly proposed algorithm.

The path generated by the proposed algorithm is similar to that projected by Theta*, but always satisfies the vehicle's maximum angular rate as well as the start and goal heading angles. These results show that the proposed algorithm generated the most natural path among all the algorithms. Fig. 5 shows average costs and their variance between start and goal positions for all the path planning methods. The cost is the average metric length of path that has been generated in the simulation. The results show that the costs of *SE*(2) grid map-based path planning algorithms such as A* and Theta* are almost the same. The average cost of 3D A* with a 45° resolution is similar to the basic A*, because 3D A* with a 45° resolution expands the orientation node with a fixed set of heading angles. The 3D A* with a 22.5° resolution, however, can satisfy the vehicle's performance although it has a fixed set of heading angles. In this case, the average cost of 3D A* is higher than other algorithms since it generates a non-smooth path due to the angle constraints. In the proposed algorithm, the search is performed on a weighted occupancy grid map with accommodation of the capability of the vehicle. Thus, the average cost of the proposed algorithm is lower than the 3D A* with 22.5° resolution. In Fig. 6, the proposed algorithm is also faster than the other *SE*(3) path planning algorithms. Considering the speed of the USV, the resultant paths are generated within 1 s in order to be operated in real-time. Since the maximum speed of the USV was set to 20 m/s in this simulation, *SE*(2) path planning algorithms including the proposed algorithm can generate the 20 m path within 1 s, which means the proposed algorithm can generate a path in real-time. In our simulation results, the proposed algorithm generated more natural paths and has shown better performance than the other *SE*(3) path planning algorithms.

3.2. Experimental results

In order to validate the proposed algorithm, an experiment was conducted using the platform shown in Fig. 7. The JEROS (Jellyfish Elimination RObotic Swarm) is an autonomous twin-hull-type USV prototype designed and developed for jellyfish detection and removal in the ocean environment by Kim et al. (2012a) as shown in Fig. 7. This prototype (length 1.5 m, width 1.1 m, weight about 50 kg in air) is a small twin-hull-type ship, with propulsion and electrical control systems. The two parallel hulls of the ship make the USV more stable during disturbances caused by waves, currents, and wind, compared to mono-hull-type ships. The propulsion system, a thruster (500 W at 24 V) installed at the rear of each hull, controls the heading of the USV by using the difference of the forces of the thrusters while the USV is moving forward. The thrusters are controlled by the electrical control system made up of a SBC (Single Board Computer), a microcontroller, GPS and IMU (Inertial Measurement Unit) sensors, motor drivers, wireless communication modules, and batteries. The real-time GNC (Guidance, Navigation and Control) system developed in C++ is run on the SBC with a 2.0 GHz dual core CPU. The navigation system calculates desired heading angles to follow given paths using the Line-of-Sight (LoS) guidance algorithm (Fossen, 2002). The LoS guidance algorithm computes the LoS vector to calculate a control input to steer the vehicle. The LoS vector is formed by connecting the robot position to an intersecting point on the path at a distance of the tracking radius ahead of the robot. Some parameters for experiments are shown in Table 2. This system acquires the absolute location and heading angle of the USV through Novatel OEM-V1 GPS with 1.5 m accuracy at 20 Hz and through IMU, respectively. The desired heading angle is controlled by thruster forces generated in the control system.

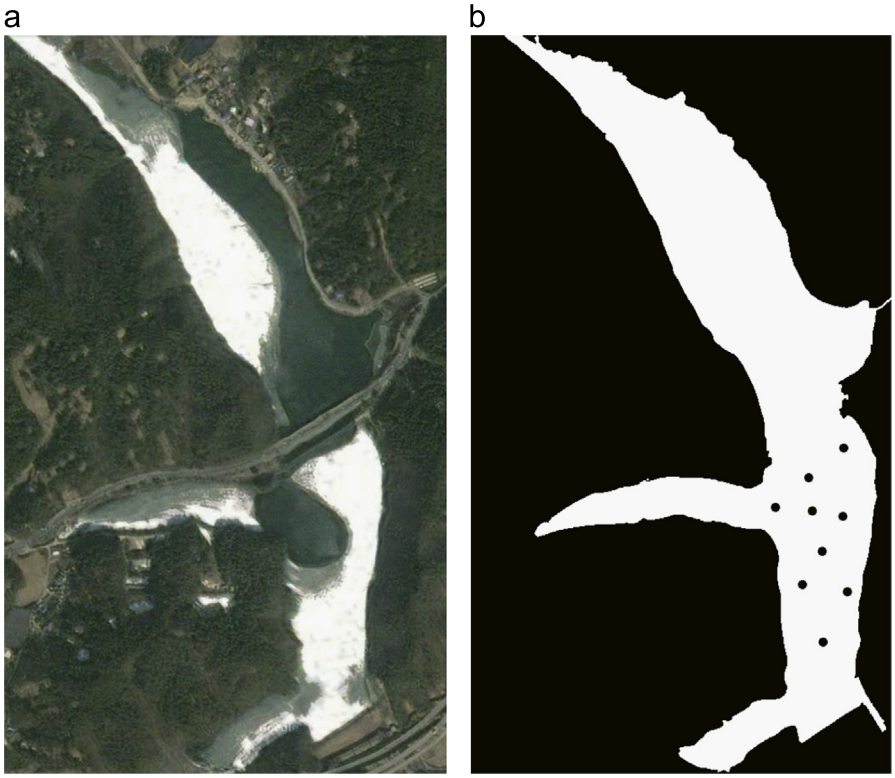

Fig. 8. (a) Aerial view and (b) grid map of experimental area. Obstacles were installed at random positions.

The experiment was carried out in Bangdong Reservoir, Daejeon, South Korea as shown in Fig. 8. Obstacles were installed at random positions, and the start and goal positions were set for 3D A* with two resolutions and the proposed algorithm. Table 2 shows the experimental setup.

The maximum tracking speed of JEROS was set to 1.0 m/s, the maximum angular rate to 1.0 rad/s, and the tracking radius of the LoS guidance was set to 3.0 m. The shortest distance from the starting point to the destination was about 500 m, and the performance of each algorithm was evaluated using the total tracking time. Fig. 9 shows the results of path tracking for all the algorithms and their details. Table 3 shows that the tracking time of the proposed algorithm was the shortest, using the same starting points and destination. Since the way-points of 3D A*

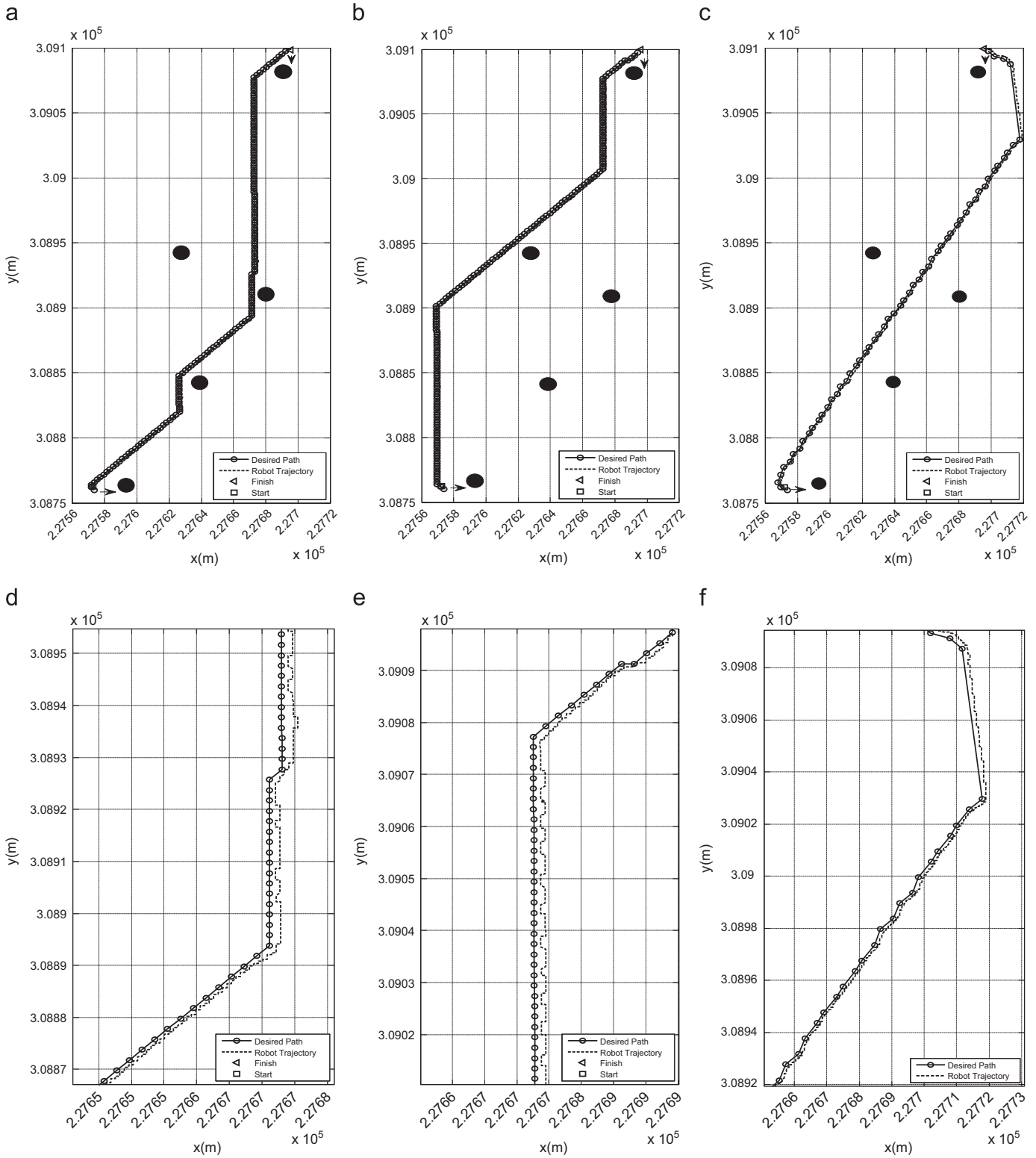

Fig. 9. Experimental results for (a) 3D A*(45° resolution), (b) 3D A*(22.5° resolution), (c) ARC-Theta*. Detailed path for each algorithm is shown in (d)–(f). All of experimental runs had the same start and goal orientation: the start heading angle was 180° and the goal heading angle was 90°.

Table 3
Results of the total tracking time of each path.

Algorithm	Time (seconds)
3D A* with 45°	710.3
3D A* with 22.5°	705.2
Proposed	692.2

were generated without considering the turning radius of the test platform, the detailed tracking result of 3D A* showed that most of the way-points located outside the turning radius were not tracked properly. These results showed that the proposed algorithm can generate a global path in real-time, and that its resultant path enables the vehicle to better follow paths in the marine environments compared to the other $SE(3)$ grid map-based path planning algorithms.

4. Conclusion

In this paper, we proposed a new ARC-Theta*, which can create paths in real-time with accommodation for the actual heading angles and steering performance of USVs. The performance of the algorithm was evaluated using both simulation and experiment. The path generation time and cost of paths projected by the newly proposed algorithm were compared to those by other path planning algorithms using Monte Carlo simulation. The simulation results showed that, even though the proposed algorithm considered the performance of the vehicle, its computation time was as fast as the $SE(2)$ grid map-based path planning algorithm. Also, the experimental results showed superior performance of path tracking time of the proposed algorithm in comparison to that of the 3D A* algorithm. In conclusion, we feel this new algorithm is suitable for global path planning for use with USVs. In future work, we plan to reflect additional factors such as nautical rules, in our path planning algorithm. Another extension in this research may be to apply this algorithm to vehicles for which paths are limited by other ships or obstacles in a harbor.

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (grant number NRF-2013R1A1A1A05011746). It was also supported by the Ministry of Trade, Industry and Energy (MOTIE), Korea, under the Human Resources Development Program for Convergence Robot Specialists support program supervised by the National IT Industry Promotion Agency (NIPA) (NIPA-2012-H1502-12-1002). Mr. H. Kim and Mr. H. Kim are supported by Korea Ministry of Land, Infrastructure and Transport (MOLIT) as U-City Master and Doctor Course Grant Program.

References

Bresenham, J.E., 1965. Algorithm for computer control of a digital plotter. J. IBM Syst. 4, 25–30.
Brockett, R.W., 1983. Asymptotic stability and feedback stabilization. In: Differential Geometric Control Theory, Birkhauser, Boston, USA, pp. 181–191.
Choi, S., Yu, W., 2011. Any-angle path planning on non-uniform costmaps. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 5615–5621.
Dijkstra, E., 1959. A note on two problems in connexion with graphs. Numer. Math. 1, 269–271.
Ferguson, D., Stentz, A., 2007. Field D*: an interpolation-based path planner and replanner. Robot. Res. 28, 239–253.
Fossen, T., 2002. Marine Control Systems: Guidance, Navigation and Control of Ships, Rigs and Underwater Vehicles. Marine Cybernetics Trondheim, Norway.
Garau, B., Alvarez, A., Oliver, G., 2005. Path planning of autonomous underwater vehicles in current fields with complex spatial variability: an A* approach. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 194–198.
Hart, P.E., Nilsson, N.J., Raphael, B., 1968. A formal basis for the heuristic determination of minimum cost paths. IEEE Trans. Syst. Sci. Cybern. 4, 100–107.
Huang, H., Pavek, K., Novak, B., Albus, J., Messin, E., 2005. A framework for autonomy levels for unmanned systems (ALFUS). In: Proceedings of the AUVSI's Unmanned Systems North America, pp. 849–863.
Isern-González, J., Hernández-Sosa, D., Fernández-Perdomo, E., Cabrera-Gámez, J., Dominguez-Brito, A., Prieto-Maranón, V., 2011. Path planning for underwater gliders using iterative optimization. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 1538–1543.
Kim, D., Shin, J.u., Kim, H., Lee, D., Lee, S.M., Myung, H., 2012a. Experimental tests of autonomous jellyfish removal robot system JEROS. In: Robot Intelligence Technology and Applications (RiTA), pp. 395–403.
Kim, H., Park, B., Myung, H., 2012b. Curvature path planning with high resolution graph for unmanned surface vehicle. In: Proceedings of the Robot Intelligence Technology and Applications (RiTA), pp. 147–154.
Kruger, D., Stolkin, R., Blum, A., Briganti, J., 2007. Optimal AUV path planning for extended missions in complex, fast-flowing estuarine environments. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 4265–4270.
Latombe, J., 1991. Robot Motion Planning. Kluwer Academic Publishers, Norwell, USA.
Laumond, J., 1987. Finding collision-free smooth trajectories for a non-holonomic mobile robot. In: Proceedings of International Joint Conference on Artificial Intelligence, pp. 1120–1123.
LaValle, S., 2006. Planning Algorithms. Cambridge University Press, New York, USA.
Manley, J.E., 2008. Unmanned surface vehicles, 15 years of development. In: Proceedings of the MTS/IEEE OCEANS, pp. 1–4.
Maritime Security Committee, 2002. Standards for Ship Maneuverability. URL: ⟨http://legacy.sname.org/committees/tech_ops/O44/imo/maneuverstandards.pdf⟩.
Nash, A., Daniel, K., Koenig, S., Felner, A., 2007. Theta*: any-angle path planning on grids. In: Proceedings of the National Conference on Artificial Intelligence, pp. 1177–1183.
Patnaik, S., Jain, L., Tzafestas, S., Resconi, G., Konar, A., 2006. Innovations in Robot Mobility and Control. Springer, New York, USA.
Prasanth-Kumar, R., Dasgupta, A., Kumar, C., 2005. Real-time optimal motion planning for autonomous underwater vehicles. Ocean Eng. 32, 1431–1447.
Rabin, S., 2000. A* speed optimizations. Game Progr. Gems 1, 272–287.
Soulignac, M., 2011. Feasible and optimal path planning in strong current fields. IEEE Trans. Robot. 27, 89–98.
Stentz, A., 1994. Optimal and efficient path planning for partially-known environments. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 3310–3317.
Sun, Z., Ge, S., Huo, W., Lee, T., 2001. Stabilization of nonholonomic chained systems via nonregular feedback linearization. Syst. Control Lett. 44, 279–289.
Szlapczynski, R., 2006. A new method of ship routing on raster grids, with turn penalties and collision avoidance. J. Navig. 59, 27–42.
Tam, C., Bucknall, R., Greig, A., 2009. Review of collision avoidance and path planning methods for ships in close range encounters. J. Navig. 62, 455–476.
Thrun, S., Burgard, W., Fox, D., et al., 2005. Probabilistic Robotics. MIT Press, Cambridge, MA.
U.S. Navy, 2007. The Navy Unmanned Surface Vehicle (USV) Master Plan. URL: ⟨http://www.navy.mil/navydata/technology/usvmppr.pdf⟩.
Witt, J., Dunbabin, M., 2008. Go with the flow: Optimal AUV path planning in coastal environments. In: Proceedings of the Australian Conference on Robotics and Automation, pp. 1–9.
Yang, Y., Wang, S., Wu, Z., Wang, Y., 2011. Motion planning for multi-HUG formation in an environment with obstacles. Ocean Eng. 38, 2262–2269.
Yong, C., Barth, E., 2006. Real-time dynamic path planning for Dubins' nonholonomic robot. In: Proceedings of the 45th IEEE Conference on Decision and Control, pp. 2418–2423.