Chen, C, Chen, XQ, Ma, F, Zeng, XJ and Wang, J

A knowledge-free path planning approach for smart ships based on reinforcement learning

A Knowledge-Free Path Planning Approach for Smart Ships based on Reinforcement Learning

Chen Chen[1,2,4], Xian-Qiao Chen[1,2], Feng Ma [3*], Xiao-Jun Zeng[4], **Jin Wang[4]**

School of Computer Science and Technology, Wuhan University of Technology, Wuhan, China[1]

Hubei Key Laboratory of Transportation Internet of Things, Wuhan University of Technology, Wuhan, China[2]

Intelligent Transportation System Center, Wuhan University of Technology, Wuhan, China[3]

School of Computer Science, University of Manchester, Manchester, UK[5]

Liverpool Logistics, Offshore and Marine (LOOM) Research Institute, Liverpool John Moores University, Liverpool, UK[5]

Abstract: With the development of artificial intelligence, intelligent and unmanned driving has received extensive attention. Compared with the rapid technological advance of unmanned vehicles, the research on unmanned ship technology is relatively rare. The autonomous navigation of cargo ships needs to meet their huge inertia and obey existing complex rules. Therefore, the requirements for smart ships are much higher than those for unmanned vehicles. A smart ship has to realise autonomous driving instead of manual operation, which consists of path planning and controlling. Toward to this goal, this research proposes a path planning and manipulating approach based on Q-learning, which can drive a cargo ship by itself without requiring any input from human experiences or guidance rules. At the very beginning, a ship is modelled in a simulation waterway. Then, a number of simple rules of navigation are introduced and regularized as rewards or punishments, which are used to judge the performance, or manipulation decisions of the ship. Subsequently, Q-learning is introduced to learn the action–reward model and the learning outcome is used to manipulate the ship's movement. By chasing higher reward values, the ship can find an appropriate path or navigation strategies by itself. After a sufficient number of rounds of training, a convincing path and manipulating strategies will likely be produced. By comparing the proposed approach with the existing Rapid-exploring Random Tree (RRT) and the Artificial Potential Field A* methods, it is shown that this approach is more effective in self-learning and continuous optimisation, and therefore closer to human manoeuvring.

Keywords: Q-learning; Path Planning; Smart Ships; Rewards; Value Function

Highlights:

[1] Novel approach in operating a vessel based on Q-learning for smart ships autonomously.

[2] Novel method to model regulations of navigation based on rewards.

1 Introduction and Background

Since the 1970s, the combination of robot technologies and vehicles has led to the emergence of drones, unmanned vehicles and unmanned boats [1]. Among them, the research of unmanned vehicles may be comprehensive. In contrast, there is much less research on unmanned ships, especially for cargo ships. Cargo ships are generally under-actuated due to their large tonnage, slow speed and relatively weak power. The autonomous navigation of cargo ships has to meet huge inertia and complex navigation rules, therefore the requirements for smart ships are much higher than those for unmanned vehicles. An operator of a cargo ship faces many challenges, including those associated with the dynamic environment, insufficient power and the uncertainties in perception. Hence, the artificial intelligence of cargo ship operating is considered to be very difficult to be built, the core function of which is path planning.

In the field of robotics, path planning is a prolonged topic. Artificial Potential Field (APF), A*, dynamic A*, Rapid-exploring Random Tree (RRT) and other algorithms have been studied and developed for many years. However, the above methods are generally based on models such as potential fields, a rule tree, or a probability tree. In general, these methods primarily take into account distances, smoothness and accessibility of paths without considering the dynamic characteristics of the corresponding agent. Therefore, the routes obtained by these planning methods often do not meet the requirements of ship kinematics and safety. In other words, traditional path planning methods could not be practical in the navigation of cargo ships.

At present, the development of artificial intelligence technology, especially the reinforcement learning, provides a new possibility to satisfy the requirements of the path planning of intelligent cargo ships. Reinforcement learning has attracted extensive attention in recent years, which emphasizes the learning of agents from the environment to behaviour mapping and seeks for the most correct or best action decision by maximizing value functions. Q-learning, DQN, A3C and DDPG are most frequently used reinforcement learning methods [2]. Different from other artificial intelligence algorithms, reinforcement learning methods do not need any human knowledge or pre-set rules. Considering the navigation difficulty for cargo ships and the benefits offered by reinforcement learning, in this research, Q-learning is introduced to address the problem of path planning for a cargo ship. The essence of path planning based on the Q-learning system is that agents can independently find the most effective path by enumerating possible solutions, which might be closer to human manipulating intelligence. The only prerequisite is to build a computing environment which is consistent with or close to the real world. Via the proposed approach, this research not only provides a novel way for smart ships to sail autonomously by considering all the characteristics of a cargo ship, but also presents a new application scenario for reinforcement learning.

The above idea is realised and implemented in this paper. Firstly, a ship agent is introduced in a simulation waterway. Based on Q-learning, an action-return mode is adopted for the evaluation and calculation of path under the constraints of navigation safety. Then an appropriate path can be found after sufficient training.

The remaining part of this paper is organised as follows. Section 2 reviews relevant literatures. Then Section 3 proposes a Q-learning based path planning approach. After that, Section 4 validates the approach in simulations. Finally the conclusion on the use of the approach is given in Section 5.

2 Literature review

2.1 Path planning methods for ships

Compared with the path planning in the field of robots and manipulators on lands, there are only a limited number of studies on the path planning of cargo ships. For many years, the A* algorithm has been the dominant approach in the relevant research. A Swiss boat named Avalon [3] is capable of generating a persuasive path to a given destination and avoiding both static and dynamic obstacles based on the A* algorithm. Langbein [4] from Ulm University made use of the A* algorithm to develop a long-term path routing approach for autonomous sailboats. In particular, such an approach had been validated in a model test. Li *et al*. [5] developed an Autonomous Underwater Vehicle (AUV) optimal path planning method on a basic of A* for seabed terrain matching navigation to explore the underwater world. In short, A* is a popular choice for path finding, which is flexible. The algorithm given in [6] uses a heuristic function to estimate the distance from the current point to the end in the graph and determine its searching direction accordingly. Although A* is very efficient for avoiding static obstacles, it might not be very suitable for dynamic environments. Apart from the A* algorithm, APF is another popular method in the path planning. Ma and Chen [7] adopted the APF model to describe the collision potential caused by buoys, piers and encountered vessels and then estimated the collision probabilities. Xue *et al*. [8] applied an APF-based method in ship automatic navigation, which can find a promising route and avoid collision. In general, the APF model is elegant and considered to be practical in path planning. However, the major issue of the APF model is that the agent might fall into local minima. In this occasion, it is very difficult for the agent to reach its destination. Chen [9] developed an extension of RRT algorithm to overcome the actual demand of multi-waypoint path planning for unmanned ship. The RRT method is a sampling-based expansion algorithm. At each step of the tree-growth, given the generated sample (random seed), the growth of the tree is attracted to this seed until a certain branch of the tree can reach the destination [10]. Alvarez *et al*. [11] presented a genetic algorithm (GA) for path planning of an AUV, which turns out to be efficient. However, the computing speed of this GA based approach is too slow to meet the requirement of real-time updating. In addition to A* and APF, other methods are also used in Unmanned Surface Vessel (USV) and AUV. Cheng and Liu [12] applied a genetic annealing algorithm to trajectory optimization based on the ship dynamic collision avoidance space model. Petres *et al*. [13] proposed a novel fast marching-based approach for the path planning of AUV, which takes the control constraints of AUV into consideration in two-dimensional calculation. Liu and Bucknall [14] suggested a constrained fast marching method to solve the problem of USV formation path planning in dynamic environments.

2.2 Reinforcement learning

As discussed previously, although these algorithms have their own advantages in path planning, none of them fully take consideration of the dynamic characteristics and path rationality. Therefore, this research introduces the Q-learning algorithm to address the problem of path planning, which considers the path planning as a continuous optimisation problem under the trade-off between gain and loss. In 1956, Bellman [15] proposed a dynamic programming method, which laid the foundation of reinforcement learning. Watkins [16] put forward Q-learning in 1989, which is the most commonly used reinforcement learning algorithm. Q-learning is a form of model-free reinforcement learning, which can also be viewed as a method of asynchronous dynamic programming (DP). The model of Q-learning is very elegant and refined. It provides agents with the capability of learning to act optimally by experiencing the consequences of actions, without requiring them to build maps of the domains.

Q-learning was known as a Markov decision process [17] with the environment fully observable, which is described as the variables below [18]:

$\boldsymbol{S}$ is a finite set of possible states.

$\boldsymbol{A}$ is a finite set of actions.

$\boldsymbol{P}$ is a state transition probability matrix, where $\boldsymbol{P(S_t, S_{t+1}, a_t)}$ is the probability of arriving in state $\boldsymbol{S_{t+1}}$ when performing action in state $\boldsymbol{S_t}$

$\boldsymbol{R}$ is a reward function.

$\boldsymbol{\gamma}$ is a discount rate, $\boldsymbol{0 \leq \gamma \leq 1.}$

$\boldsymbol{\pi : \to S \times A}$ is a state transaction function.

In Q-learning, a series of different stages or episodes constitute the experience of an agent. In the t^{th} episode, the agent has the following steps: 1) Observe its current state S_t, 2) Select and perform an action a_t, 3) Observe the subsequent state S_{t+1}, 4) Receive an immediate return value R_t, 5) The Q_{t-1} value is adjusted by using the learning factor α_n, and the Q value is obtained according to the following formula:

$$Q(S_t, a_t) \leftarrow Q(S_t, a_t) + \alpha[R_t + \gamma Q(S_{t+1}, a_{t+1}) - Q(S_t, a_t)] \quad (1)$$

where $Q(S_t, a_t)$ is the value of an action a_t executed by the agent; R_t is the immediate reward; α is a learning rate and γ is the discount factor.

When this process is sufficient, the agent generates a memory that can select actions to maximize the total future reward in different states. Q-learning combines Monte Carlo sampling and Dynamic Programming bootstrapping. Therefore, it can learn online after each step and from incomplete sequence, with low variance and high efficiency. Compared with the path planning methods briefly described previously, Q-learning has been applied in complex and challenging environments, such as enlarged state spaces, increased computational complexity, energy requirements, safety issues and many other constraints. Demircan *et al*. [19] took advantage of Q-learning to find out an optimum route for electrical energy transmission lines under specified criteria. Konar *et al*. [20] provided a Q-learning-based path planning algorithm for a mobile robot with respect to time, the number of states traversed and energy consumption. Zolfpour *et al*. [21] presented a multi-agents reinforcement learning model for a route planning system to address the vehicle delay problems by studying the weights of various components in road network environments such as weather, traffic, road safety, and fuel capacity to create a priority route plan for vehicles between cities of Malaysia. Li [22] presented a reinforcement learning algorithm using linear function approximation to generate an optimal path by controlling the choice of four moving actions of the microrobot. Zhang *et al*. [23] developed a reinforcement learning path-following control strategy based on approximate policy iteration for a real mobile robot. Compared with traditional methods, this method offers better convergence and a higher path tracking accuracy.

There are very few applications of reinforcement learning in the field of waterway. Gaskett *et al*. [24] used the Q-learning algorithm to make AUV control its propeller according to input commands and sensor information in order to find the target autonomously. El-Fakdi and Carreras [25] demonstrated the feasibility of reinforcement learning techniques in underwater cable tracking tasks for AUV. Yoo and Kim [26] used reinforcement learning for marine vehicles path optimisation in ocean environments without considering obstacles.

From the above discussion, it can be concluded that there is very little research on automatic manoeuvring of ships by using reinforcement learning. The existing methods of ship route planning are traditional ones in the field of robots, which do not consider the characteristics of ships. Reinforcement learning is a trial-and-error learning algorithm, in which the agent interacts with the environment in real-time and tries constantly to obtain an appropriate strategy. Such a method can be used to address the problem that needs to make continuous decisions to achieve an objective. The path planning and control of ships are such problems that requite continuous decision-making. Therefore, this research proposes reinforcement learning to address the problems. Taking into account the ship kinematics equation in the learning process, it enables the automatic self-driving of ships, filling the gap between reinforcement learning and the manoeuvring of smart cargo ships.

3 A proposed approach

The objective of this research is not to find a shortest path for a cargo ship, which often is not an appropriate path for the corresponding ship, but to find a practical path which takes into consideration of the dynamic characteristic and a certain criterion. This problem can be addressed by Q-learning, which enables a smart ship agent to learn convincing actions in a dynamic environment with no prior-knowledge. To make the simulation agree with the dynamic characteristics of a cargo ship, the given research uses the first-order Nomoto model to establish the initial state Q-table. Based on the risk of the ship collision with an obstacle in simulation, a reward function is formulated. Subsequently, the Q-learning principle is used for repeated training. After sufficient training, the artificial intelligence of path planning is established, and a meaningful and reasonable path can be found. To verify the performance of the proposed approach, such a path is compared with those obtained the traditional A * and RRT algorithms.

3.1 Modelling of the Q-table

In this research, a state space of the ship agent is formulated based on its position and heading. In addition, an action space is associated with its rudder angle. It is known that the position and heading of the corresponding ship in a certain time can be inferred by the first-order KT model. Such a model was proposed in 1957 by Nomoto who regarded various motion changes of a ship caused by the steering as the response relationship. That is, the input is rudder angle and the output is the ship motion change. Based on such a relationship, the first order Nomoto equation [27] is then proposed as the first order KT equation, which can be expressed by Eq. (2).

$$T\dot{r}+r=K\delta \tag{2}$$

with the notation

$$\dot{\psi}=r \tag{3}$$

where ψ is the heading of a ship, Eq. (2) can be written as

$$T\ddot{\psi}+\dot{\psi}=K \tag{4}$$

where K is the turning ability coefficient, T is the turning lag coefficient, r is the yaw rate, and δ is the rudder angle.

The first order Nomoto model has been widely employed in simulating the movement of ships. The yaw dynamics is characterized by parameters K and T, which can be identified from standard manoeuvring tests [28].

Taking the ship as a rigid body, when the ship steers at any rudder angle δ**,** the bow of the ship rotates at a certain angle and the yaw rate is ***r***. The above formula can be seen as the bow rotation equation of the ship when it steers.

Assuming that the initial conditions are $t = 0$, $\delta=\delta_0$ and $r = 0$, then yaw rate r at any moment ***t*** can be computed through the first-order KT equation [29].

$$r = K\delta_0(1-e^{-t/T}) \tag{5}$$

Since the yaw rate r is actually the time derivative of Ψ, ship heading angle Ψ can be obtained as

$$\psi = K\delta_0(t-T+T\cdot e^{-t/T}) \tag{6}$$

In order to describe the motion of a ship, a ship motion coordinate system is established, as shown in Fig. 1. In this figure, G represents the position of the centre of gravity of the ship, XOY indicates the hydrostatic water plane, O is the origin, X_{OG} and Y_{OG} represent the projection of the centre of ship gravity G on the X and Y axes respectively, ψ is the heading of ship, and δ indicates the ship rudder angle. The position of the ship at any moment can be calculated by Eq. (7).

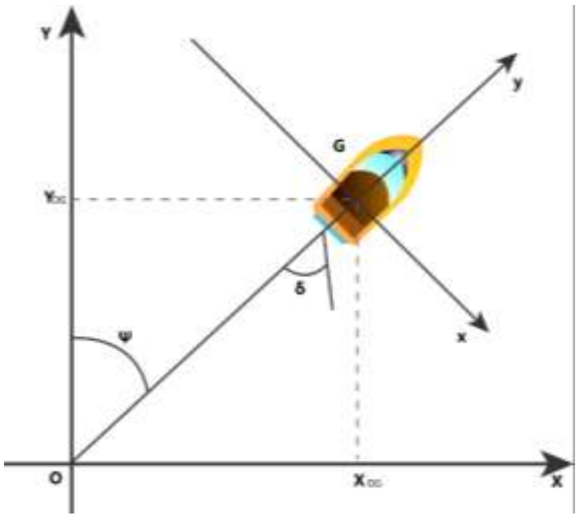

Fig.1. A ship motion coordinate system

$$\begin{cases} x(t)=x(0)+\int v \cdot sin\psi dt \\ y(t)=y(0)+\int v \cdot cos\psi dt \end{cases} \tag{7}$$

where $\boldsymbol{x}(\mathbf{0})$ and $\boldsymbol{y}$ $(\mathbf{0})$ are the initial positions, $\boldsymbol{v}$ is the ship speed, and $\boldsymbol{\psi}$ is the heading of ship. With the help of these equations above, the position and heading can always be inferred.

To enable the reinforcement learning, in this research, the return function is defined as follows. If the distance from the obstacle is greater than the minimum collision distance, it is set to 1. If the distance from the obstacle is smaller than the minimum collision distance, it is set to – 1000. In this occasion, the system will quit and start learning again. This is also in line with the navigation rules. In this way, it is possible to obtain the corresponding return value of the ship in any state after taking any action, so as to establish the Q-table of the corresponding ship.

3.2 Learning process

Through the modelling in Section 3.1, the selection of ship actions and the state of the next step can be linked by the first-order KT model. Hence, Q-learning can be introduced to establish autonomous path planning intelligence. The learning process of Q-learning can be designed to start from any initial state and then select actions according to an action policy. After taking the selected action, the agent observes the following state and finds the reward, and then updates the Q-value of the previous state and action based on the maximum Q-value and reward of the new state. The Q-learning action policy employs the ε-greedy policy, which balances "exploration" and "exploitation" [30]. Exploitation refers to select an action with the largest value function. Exploration, on the other hand, means that other attempting actions still have a chance to be chosen. This will make the agent learn experience from the environment, ensuring that the agent searches all possible actions to avoid being trapped in local optimal actions. The target policy is a greedy policy, which is also a deterministic policy. Only when the value function is maximized, the probability is 1 and the other action probability is set to be 0. The algorithm is given as follows:

Algorithm 1 *Q-learning Algorithm*

1: Initialize $\mathbf{Q}(\mathbf{s}, \mathbf{a})$, $\forall \mathbf{s} \in \mathbf{S}, \mathbf{a} \in \mathbf{A}(\mathbf{s})$, arbitraily

2: Assign a learning factor $\boldsymbol{\alpha}$, a discount factor $\boldsymbol{\gamma}$

3: Repeat (for each episode)

4: Initialize state $\boldsymbol{S_t}$

5: Repeat (for each step of episode)

Choose $\boldsymbol{a_t}$ from $\boldsymbol{S_t}$ according to ε-greedy policy,

Take action $\boldsymbol{a_t}$, observe $\boldsymbol{r_t}$ and next state $\boldsymbol{S_{t+1}}$

Update Q-value by using Eq. (1)

$\boldsymbol{S}$=$\boldsymbol{S'}$, $\boldsymbol{a} = \boldsymbol{a'}$

Until $\boldsymbol{S}$ is terminal

Until all $Q(s, a)$ is of convergence

Output the last policy:

$$\pi(s) = \underset{a}{arg\,max}\, Q(s, a)$$

In summary, the Q-learning algorithm presented stores Q-values at each state for the optional action. When the learning process is completed, all the states are determined; the Q-table can be used for the coming path-planning applications.

4 A case study

To verify the effectiveness of the proposed method, the PyCharm platform was used to establish a simulation environment of a waterway. In this platform, the Q-learning-based path planning function was implemented by Python. Subsequently, the RRT and A* methods were also used to plan the paths respectively in the same platform and under the same scenario, and the results of the three methods were compared. For the given case study, it is worth highlighting that dynamic characteristics of a ship are different from those of a vehicle. In fact, a cargo ship always tries to maintain its speed and direction, since speed change and sharp turn would hurt its engine or could lead to capsize. Basically, the engine speed of a cargo ship stays on a constant value in most cases, and the rudder maintains a small angle in turning. In order to simulate the dynamic characteristics of a ship, it is fair to simplify the model, and make the assumption that the simulated ship always sails on a constant speed.

4.1 An experimental platform

This research established a virtual navigation environment for ships, which is shown in Fig. 2. The size of this map is set to 800×600, where the bottom-left corner is taken as the origin (0, 0). In this map, the orange area is considered as the land, and the blue area is considered as a waterway, which is 400 in width, and 600 in length. It should be pointed out that the left-bottom marginal point of this waterway is (200, 0). In this case study, there are 4 objective scenarios designed for this experiment. The four scenarios are labelled from 1 to 4, each of which represents the number of obstacles in the corresponding scenario. Therefore, there are 4 obstacles in scenario 4, which is most complicated. The size of the obstacles is set to be 100×50. In scenario 1, there is only obstacle 1 located at the lower left corner (300, 200). In scenario 2, obstacle 2 is added at (400, 450). Similarly, obstacle 3 has been added in scenario 3 at (350, 325), obstacle 4 has been added in scenario 4 at (400, 100). At the same time, a cargo ship is simulated with a length $\boldsymbol{L}$ of 59 and a width $\boldsymbol{W}$ of 30. Certainly other sizes are also applicable in this test. The initial position of this ship is (400, 30), and its initial heading angle $\boldsymbol{\Psi}$ is set to zero.

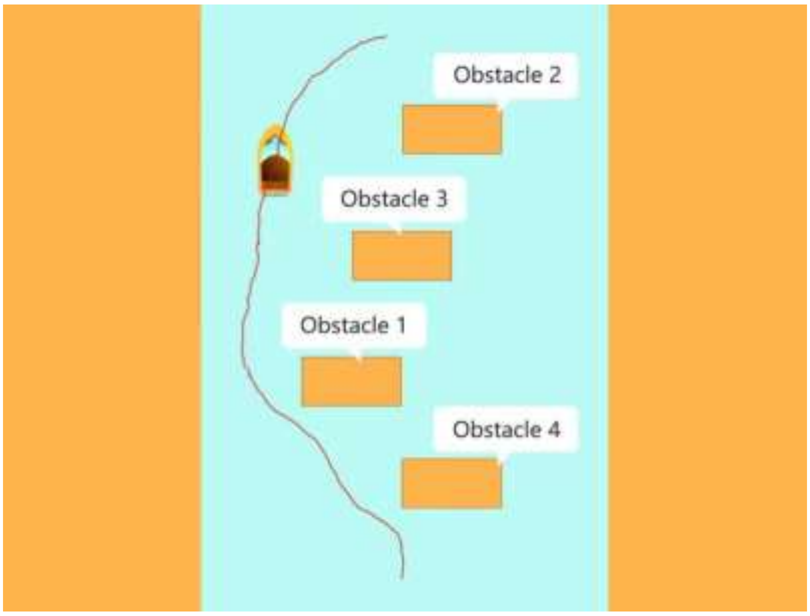

Fig.2. Path taken by Q-learning algorithm

4.2 Path planning based on the proposed approach

To simplify the training process, the map was rasterized and divided into grids at the size of 5 ×5, and the heading is represented as 20 steps, as the interval is 18°. Hence, the state space of the ship is related to its coordinates and headings, which can be represented by

$$\left[\left(x_1,\ y_1,\ \psi_1\right),\left(x_2,\ y_2,\ \psi_2\right),\ldots,\left(x_n,\ y_n,\ \psi_n\right)\right] \tag{8}$$

where $\boldsymbol{x}$ is the X-coordinate, $\boldsymbol{y}$ is the Y-coordinate and ψ is the heading of the ship.

As discussed previously, the speed of the simulated ship is set to be constant. Therefore, the rudder angle is the only action option for the ship. In fact, the steering angle of a cargo ship generally does not have many choices, which is between ±35°. The Q-learning algorithm requires the action space of an agent being discrete, and therefore the discretization process is applied. Without loss of generality but simplifying the model and computing, the action space of the ship is set to 5 options, [-35, -15, 0, 15, 35]. That is, the ship can select only five rudder angles in any state in this case study.

Taking scenario 4 (the most complicated one) of Fig.2 as an example, according to the size of the map, the shape of the waterway and the dynamic characteristics of the cargo ship, and a rule that the ship cannot collide with the land and obstacles during the simulation; the reward value is defined by the following rules:

(1) If $\boldsymbol{300 - W/2 < x < 400}$+$W/2$, and $\boldsymbol{200 - L/2 < y < 250 + L/2}$, the ship has collided with obstacle 1, $\boldsymbol{r = -1000}$. The learning process will exit and re-start.

(2) If $\boldsymbol{400 - W/2 < x < 500 + W/2}$, and $\boldsymbol{450 - L/2 < y < 500 + L/2}$, the ship has collided with obstacle 2, $\boldsymbol{r = -1000}$. The learning process will exit and re-start.

(3) If $\boldsymbol{350 - W/2 < x < 450 + W/2}$, and $\boldsymbol{325 - L/2 < y < 375 + L/2}$, the ship has collided with obstacle 3, $\boldsymbol{r = -1000}$. The learning process will exit and re-start.

(4) If $\boldsymbol{400 - W/2 < x < 500 + W/2}$, and $\boldsymbol{100 - L/2 < y < 150 + L/2}$, the ship has collided with obstacle 4, $\boldsymbol{r = -1000}$. The learning process will exit and re-start.

(5) If $\boldsymbol{x - W/2 < 200}$, or $\boldsymbol{x + W/2 > 600}$, the ship has collided with the land, $\boldsymbol{r = -1000}$. The learning process will exit and re-start

(6) If $\boldsymbol{y - L/2 < 0}$, or $\boldsymbol{y + L/2 > 600}$, the ship has sail out of the map, $\boldsymbol{r = -1000}$. The learning process will exit and re-start

(7) If $400 < \boldsymbol{x + W/2 < 450}$, or $\boldsymbol{y + L/2 > 550}$, the ship has reached the destination, $\boldsymbol{r = 1000}$.

(8) Other situations, the ship can be considered as sailing properly in the simulation world, $\boldsymbol{r = 1}$.

In this experiment, the ship manoeuvrability coefficients are given as follows: $\mathbf{K = 0.08}$,$\boldsymbol{T = 10.8}$. In fact, other $\boldsymbol{K}$ and $\boldsymbol{T}$ values are also applicable, which can be set up according to any specific vessel. On the other hand, the learning factor $\boldsymbol{\alpha}$ is set as $\mathbf{0.3}$ and the discount factor $\boldsymbol{\gamma}$ as $\mathbf{0.99}$; the variation of these values might change the learning speed and sometimes need some pre-analysis and simulation to determine. In this study, the learning factor $\boldsymbol{\alpha}$ and the discount factor $\boldsymbol{\gamma}$ are assigned with typical values. The Q-table are updated every 20 seconds so as to reduce the computation, and the value function update formula can be presented as:

$$\boldsymbol{Q(S_t, a_t) \leftarrow Q(S_t, a_t) + 0.3[R_t + 0.99Q(S_{t+1}, a_{t+1}) - Q(S_t, a_t)]} \tag{9}$$

Table 1 gives a table of the value function of an episode, which is calculated every 20 seconds. For example, in the last state 14, $\boldsymbol{x - W/2 < 200}$. It means that the ship collides with the shore and the value function is -300, the training ends and restarts.

Table 1 The Q-table of an episode

	x	y	ψ	Action	Reward

State1	396.05	49.34	-30.51	0	3.08
State2	385.90	66.57	-30.51	0	3.17
State3	375.74	83.80	-30.51	0	3.24
State4	365.59	101.03	-30.51	0	3.28
State5	359.18	119.70	0.00	-35	1.89
State6	355.23	139.04	-30.51	-15	1.72
State7	343.65	155.29	-43.58	-15	2.18
State8	328.70	168.50	-56.65	-15	4.52
State9	311.15	177.99	-69.73	15	4.59
State10	292.39	184.92	-69.73	0	0.02
State11	273.62	191.85	-69.73	35	0.08
State12	254.11	194.84	-100.23	0	0
State13	234.43	191.29	-100.23	0	0
State14	214.75	187.74	-100.23	0	-300

During 300,000 times of learning, the value function table was continuously updated, and the policy with the largest value function was improved. Finally, the path planned by this algorithm is shown by the red line in Fig.2. In fact, the learning can be implemented for ever. The more training, the more satisfied result would be found.

4.3 Comparing with the existing methods

In the same scenario, Q-learning, RRT and A* algorithms are used for path planning respectively. Then the paths planned by the three methods under the four scenarios are shown in Fig.3 to Fig.6.

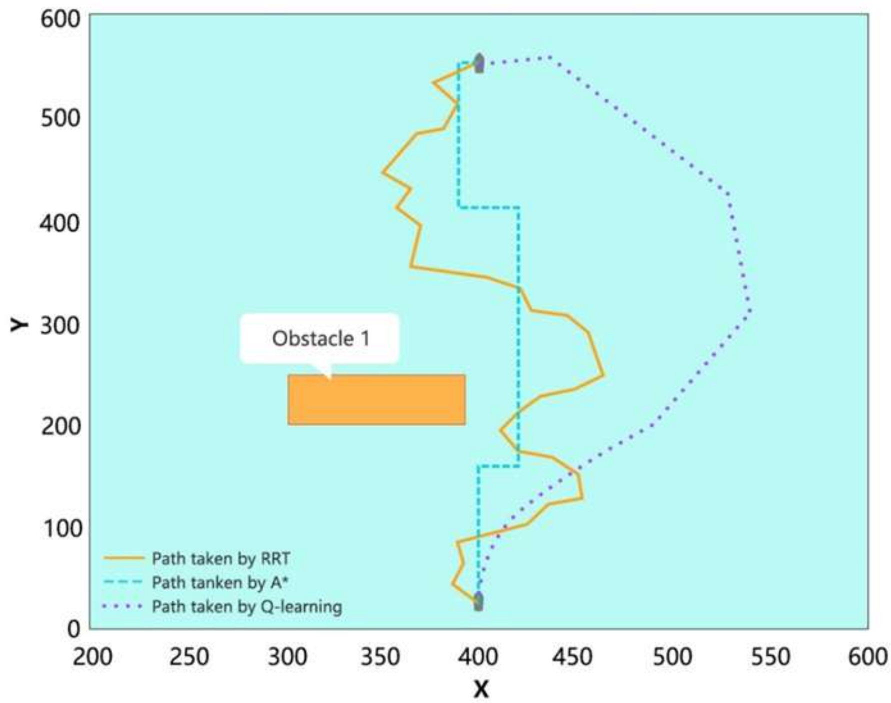

Fig.3. Paths planned by different methods in scenario 1

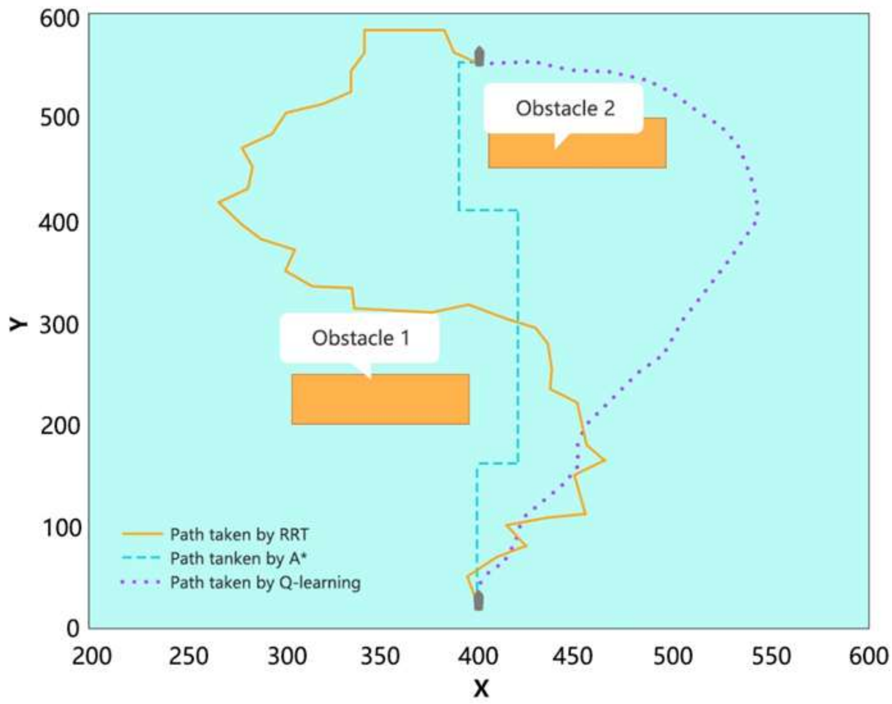

Fig.4. Paths planned by different methods in scenario 2

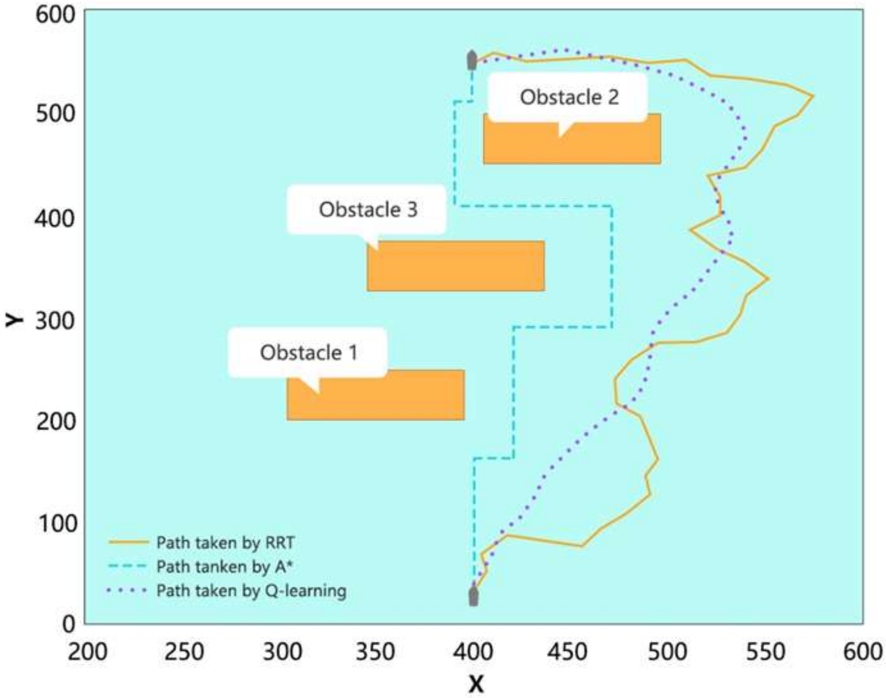

Fig.5. Paths planned by different methods in scenario 3

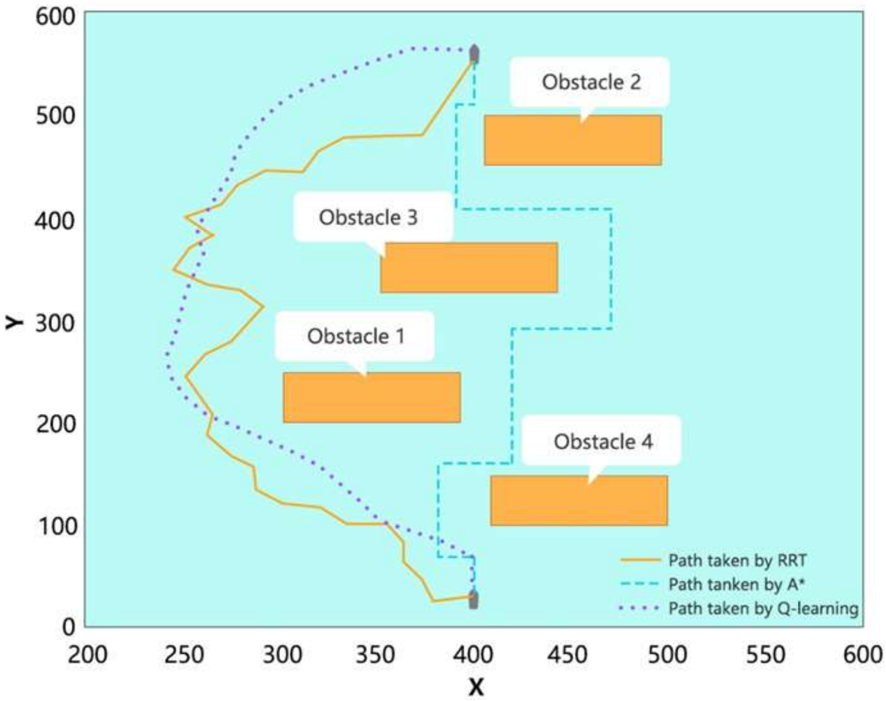

Fig.6. Paths planned by different methods in scenario 4

The length of the paths, the number of sharp turns, the number of 90◦ turns planned by the three methods are compared respectively, and the results are shown in Table 2. Sharp turns refer to angles greater than 35° or angles less than -35°.

Table 2 The distances, the number of sharp turns and the number of 90° turns produced by the three methods

Scenario	The Distance of Path			No. of sharp turns			No. of 90° turns		
	A*	RRT	Q-learning	A*	RRT	Q-learning	A*	RRT	Q-learning
Scenario 1	580	690.72	631.47	5	16	0	5	0	0
Scenario 2	600	838.53	664.79	5	1	0	5	0	0
Scenario 3	680	831.25	663.99	8	18	0	8	0	0
Scenario 4	720	733.72	667.13	10	16	0	10	0	0

Based on Table 2, it can be concluded that the path planned by the A* algorithm is the shortest in scenario 1 and scenario 2, which is better than the one produced by Q-learning. However, with the increasing number of obstacles, the path distance of the A* algorithm turns out to be longer than that produced using Q-learning algorithm in scenario 3 and scenario 4. In addition, the path of the A* algorithm in all scenarios contains many 90° sharp turns. For cargo ships, such sharp turns are not practical and cannot be realized. The path distance produced by the RRT algorithm is the longest. Although there is no 90° turn in this path, it requires the ship steering the rudder at sharp degrees, which does not conform to the reality of ship manipulating. Therefore, the path planned by the Q-learning algorithm proposed in this research is a realistic and feasible one, and therefore superior to the path produced by the RRT algorithm.

It is worth noting that the path distance of Q-learning is the shortest in scenarios 3 and 4. With the increasing of obstacles, the superiority of Q-learning algorithm becomes increasingly obvious. At the same time, this method has already taken into account the dynamic characteristics of the ship in its learning process. In fact, the output of the proposed approach not only provide a practical path for the corresponding ship, but also the sequential manipulating strategies. In other words, the controlling process of this ship has been completed simultaneously. Therefore, the path produced by this method can be easily implemented in real applications and enables the self-adjustment with the environments where the ship operates.

5 Conclusions and discussions

This research proposed a novel approach of ship path planning based on the Q-learning algorithm. To make the approach practical, the first-order Nomoto equation was used, by which the following position and heading angle of the ship can be inferred according to the present position, rudder angle and heading of the ship. Moreover, the position, rudder angle and heading of the ship are also the fundamental factors of the state space and action space for the ship. According to the characteristics of the simulation waterway, the action reward value was defined and the value function updating formula was proposed. In the learning or training, the value function table was continuously updated. Finally, a rational path can always be found. The feasibility and effectiveness of the proposed approach wass validated and compared with the traditional path planning methods, A* and RRT algorithms. This research provides an effective route planning method for ships manoeuvring.

In the simulation process of this experiment, there are no other dynamic obstacles such as ships in the waterway, which should be taken into consideration in future research. Moreover, DQN or other strategy gradient methods which are considered to be capable of addressing the problem of the space explosion are worth investigation and experiment. Furthermore, ship collision avoidance rules could be taken into consideration in the ship agents model reward function during the process of learning.

Acknowledgements

The first author is financed by the China Scholarship Council under Grant 201706950028. This work is supported by the National Natural Science Foundation of China (Grant no. 61503289).

References

[1]Seuwou, P., Banissi, E., Ubakanma, G., Sharif, M.S., Healey, A., 2017. Actor-network theory as a framework to analyse technology acceptance model’s external variables: the case of autonomous vehicles, in: International Conference on Global Security, Safety, and Sustainability. Springer, pp. 305–320.

[2]Gupta, J.K., Egorov, M., Kochenderfer, M., 2017. Cooperative multi-agent control using deep reinforcement learning, in: International Conference on Autonomous Agents and Multiagent Systems. Springer, pp. 66–83.

[3]Erckens, H., Büsser, G.-A., Pradalier, C., Siegwart, R.Y., 2010. Navigation strategy and trajectory following controller for an autonomous sailing vessel. IEEE RAM 17, 47–54.

[4]Langbein, J., Stelzer, R., Frühwirth, T., 2011. A Rule-Based Approach to Long-Term Routing for Autonomous Sailboats. Springer Berlin Heidelberg.

[5]Li, Y., Ma, T., Chen, P., Jiang, Y., Wang, R., Zhang, Q., 2017. Autonomous underwater vehicle optimal path planning method for seabed terrain matching navigation. Ocean Eng. 133, 107–115.

[6]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.

[7]Ma, F., Chen, Y.-W., 2018. Probabilistic Assessment of Vessel Collision Risk: An Evidential Reasoning and Artificial Potential Field-Based Method, in: Multi-

Criteria Decision Making in Maritime Studies and Logistics. Springer, pp. 123–149.
[8]Xue, Y., Clelland, D., Lee, B.S., Han, D., 2011. Automatic simulation of ship navigation. Ocean Eng. 38, 2290–2305.
[9]Chen, X., Liu, Y., Hong, X., Wei, X., Huang, Y., 2018. Unmanned Ship Path Planning Based on RRT, in: International Conference on Intelligent Computing. Springer, pp. 102–110.
[10]Dong, Y., Zhang, Y., Ai, J., 2017. Experimental test of unmanned ground vehicle delivering goods using RRT path planning algorithm. Unmanned Syst. 5, 45–57.
[11]Alvarez, A., Caiti, A., Onken, R., 2004. Evolutionary path planning for autonomous underwater vehicles in a variable ocean. IEEE J. Ocean. Eng. 29, 418–429.
[12]Cheng, X., Liu, Z., 2007. Trajectory optimization for ship navigation safety using genetic annealing algorithm, in: Natural Computation, 2007. ICNC 2007. Third International Conference On. IEEE, pp. 385–392.
[13]Petres, C., Pailhas, Y., Patron, P., Petillot, Y., Evans, J., Lane, D., 2007. Path planning for autonomous underwater vehicles. IEEE Trans. Robot. 23, 331–341.
[14]Liu, Y., Bucknall, R., 2015. Path planning algorithm for unmanned surface vehicle formations in a practical maritime environment. Ocean Eng. 97, 126–144.
[15]Bellman, R., 1956. Dynamic programming and Lagrange multipliers. Proc. Natl. Acad. Sci. 42, 767–769.
[16]Watkins, C.J.C.H., Dayan, P., 1992. Q-learning. Mach. Learn. 8, 279–292.
[17]Bellman, R., 1957. A Markovian decision process. J. Math. Mech. 679–684.
[18]Sato, M., Abe, K., Takeda, H., 1988. Learning control of finite markov chains with an explicit trade-off between estimation and control. IEEE Trans. Syst. Man. Cybern. 18, 677–684.
[19]Demircan, S., Aydin, M., Durduran, S.S., 2011. Finding optimum route of electrical energy transmission line using multi-criteria with Q-learning. Expert Syst. Appl. 38, 3477–3482.
[20]Konar, A., Goswami, I., Singh, S.J., Jain, L.C., Nagar, A.K., 2013. A Deterministic Improved Q-Learning for Path Planning of a Mobile Robot. IEEE Trans. Syst. Man, Cybern. Syst. 43, 1141–1153.
[21]Zolfpour-Arokhlo, M., Selamat, A., Hashim, S.Z.M., Afkhami, H., 2014. Modeling of route planning system based on Q value-based dynamic programming with multi-agent reinforcement learning algorithms. Eng. Appl. Artif. Intell. 29, 163–177.
[22]Li, J., Li, Z., Chen, J., 2008. Reinforcement learning based precise positioning method for a millimeters-sized omnidirectional mobile microrobot, in: International Conference on Intelligent Robotics and Applications. Springer, pp. 943–952.
[23]Zhang, P., Xu, X., Liu, C., Yuan, Q., 2009. Reinforcement learning control of a real mobile robot using approximate policy iteration, in: International Symposium on Neural Networks. Springer, pp. 278–288.

[24]Gaskett, C., Wettergreen, D., Zelinsky, A., 1999. Reinforcement learning applied to the control of an autonomous underwater vehicle, in: Proceedings of the Australian Conference on Robotics and Automation (AuCRA99).

[25]El-Fakdi, A., Carreras, M., 2013. Two-step gradient-based reinforcement learning for underwater robotics behavior learning. Rob. Auton. Syst. 61, 271–282.

[26]Yoo, B., Kim, J., 2016. Path optimization for marine vehicles in ocean currents using reinforcement learning. J. Mar. Sci. Technol. 21, 334–343.

[27]Nomoto, K., Taguchi, T., Honda, K., Hirano, S., 1957. On the steering qualities of ships. Int. Shipbuild. Prog. 4, 354–370.

[28]Liu, Y., Noguchi, N., Ali, R.F., 2017. Simulation and test of an agricultural unmanned airboat maneuverability model. Int. J. Agric. Biol. Eng. 10, 88–96.

[29]Zhang, J., Yan, X., Chen, X., Sang, L., Zhang, D., 2012. A novel approach for assistance with anti-collision decision making based on the International Regulations for Preventing Collisions at Sea. Proc. Inst. Mech. Eng. Part M J. Eng. Marit. Environ. 226, 250–259.

[30]Sutton, R.S., Barto, A.G., 1998. Reinforcement learning: An introduction. MIT press.