Journal of
Marine Science and Engineering

Article

Path Planning of Coastal Ships Based on Optimized DQN Reward Function

Siyu Guo, Xiuguo Zhang *, Yiquan Du, Yisong Zheng and Zhiying Cao

School of Information Science and Technology, Dalian Maritime University, Dalian 116026, China; guosy@dlmu.edu.cn (S.G.); duyiquan@dlmu.edu.cn (Y.D.); zhengyisong@dlmu.edu.cn (Y.Z.); czysophy@dlmu.edu.cn (Z.C.)
* Correspondence: zhangxg@dlmu.edu.cn; Tel.: +86-185-5305-9562

Abstract: Path planning is a key issue in the field of coastal ships, and it is also the core foundation of ship intelligent development. In order to better realize the ship path planning in the process of navigation, this paper proposes a coastal ship path planning model based on the optimized deep Q network (DQN) algorithm. The model is mainly composed of environment status information and the DQN algorithm. The environment status information provides training space for the DQN algorithm and is quantified according to the actual navigation environment and international rules for collision avoidance at sea. The DQN algorithm mainly includes four components which are ship state space, action space, action exploration strategy and reward function. The traditional reward function of DQN may lead to the low learning efficiency and convergence speed of the model. This paper optimizes the traditional reward function from three aspects: (a) the potential energy reward of the target point to the ship is set; (b) the reward area is added near the target point; and (c) the danger area is added near the obstacle. Through the above optimized method, the ship can avoid obstacles to reach the target point faster, and the convergence speed of the model is accelerated. The traditional DQN algorithm, A* algorithm, BUG2 algorithm and artificial potential field (APF) algorithm are selected for experimental comparison, and the experimental data are analyzed from the path length, planning time, number of path corners. The experimental results show that the optimized DQN algorithm has better stability and convergence, and greatly reduces the calculation time. It can plan the optimal path in line with the actual navigation rules, and improve the safety, economy and autonomous decision-making ability of ship navigation.

Keywords: path planning; deep reinforcement learning; decision-making; obstacle avoidance

Citation: Guo, S.; Zhang, X.; Du, Y.; Zheng, Y.; Cao, Z. Path Planning of Coastal Ships Based on Optimized DQN Reward Function. *J. Mar. Sci. Eng.* **2021**, *9*, 210. https://doi.org/10.3390/jmse9020210

Academic Editor: Marco Cococcioni
Received: 18 January 2021
Accepted: 12 February 2021
Published: 18 February 2021

1. Introduction

In recent years, marine transportation has developed rapidly, and trade exchanges between countries have become more frequent. The marine transportation industry has greatly promoted the development of world economy, and ship intelligence plays an important role in marine transportation [1]. With the increasing demand for maritime traffic, the marine environment has become more complex, which greatly increases the risk of maritime traffic accidents. According to the statistics of the European Maritime Safety Agency (EMSA), there were more than 3000 marine accidents and casualties in 2017 [2]. Among these accidents, the incidence of ship collision accidents was the highest, among which accidents caused by human factors increased, causing serious casualties, property losses and marine environmental pollution. The important purpose of intelligent decision-making for coastal ships is to reduce the incidence of maritime traffic accidents and ensure navigation safety. Therefore, the ability to accurately avoid obstacles and plan a reasonable and safe shipping path, and at the same time quickly provide emergency decision-making plan is an important guarantee to solve the problem of ship safe driving, and it is also the key research field of many maritime experts and scholars [3].

The coastal ship path planning is aimed to obtain a collision free and constrained path in a specific environment, considering various factors. The ship path planning is divided into static and dynamic global path planning and local path planning. The paper of global path planning mainly provides a general macro path solution for ship navigation, and more geographic location information and meteorological information will be considered in the course of path formulation [4]. Local path planning is mainly to avoid dynamic obstacles in specific scenarios, and conduct real-time collision avoidance through a series of decisions [5].

This paper aims to solve the problem of global path planning of ships under coastal environmental conditions, which is an important part of the autonomous development of intelligent ships. In the development of ship decision-making system, this paper proposes a coastal ship path planning model based on deep Q network (DQN) algorithm, and verifies the robustness and efficiency of the model in the actual environment of electronic charts.

In this study, we divide the ship path planning problem into three sub problems:

1. Processing of environmental information

Environmental processing is to divide the marine environment under study into navigable areas and non-navigable areas, and to identify the environmental characteristics of each location to complete the final modeling of the marine environment. Environmental status information mainly includes marking navigable areas, marking non-navigable areas, and establishing obstacle information. The grid method [6] is a commonly used environment modeling method. Its basic idea is to use a grid with a certain resolution to represent the environment, and to identify the meaning of these grids to complete the system modeling of the environment.

2. Path search

Searching for a path means to predict a feasible path from the starting point to the target point without collision through a specific algorithm model based on the established environmental model. In all the predicted paths, through the optimization function iteratively constrained the normativity of the path, and finally selected an optimal path that satisfies the constraint conditions from all feasible paths as the actual ship path.

3. Path smoothing

In the actual navigation of a ship, not only the economy of the path must be considered, but the safety of the ship's navigation must be ensured at the same time. For example, the ship cannot pass through complicated obstacles and the turning angle of the ship should not be too large. Therefore, it is necessary to evaluate and optimize the optimal path. If the planned path passes through the middle of the obstacles, or the turning angle of the ship is too large at some place, these phenomena are not in line with the safe navigation rules of the ship, and the path needs to be smoothed. Through smooth processing, the obtained path will be more in line with the actual navigation specifications of the ship, and the safety of the ship's navigation will be improved.

According to the above analysis, the paper on the independent path planning of coastal ships can be divided into the following aspects. Firstly, it is necessary to process the environmental state information, distinguish navigable area and non-navigable area, and expand the complex obstacle area. Then, according to the established marine environment model, an appropriate intelligent algorithm is selected to search the optimal path. Finally, the safety and economy of the path are evaluated, and the relevant path smoothing method is adopted to make the planned path more in line with the actual navigation specifications.

The remaining sections of the paper are organized as follows. Related works are presented in Section 2. The optimization of DQN algorithm and the path planning model of coastal ships is presented in Section 3. Experimental verification and result analysis are presented in Section 4. The paper is concluded in Section 5.

2. Related Research

With the development of shipping industry, more and more researchers pay attention to the path planning of coastal ships. At present, the path planning algorithms of coastal ships mainly include traditional non-intelligent algorithms, bionic intelligent algorithms and machine learning algorithms.

Traditional non-intelligent algorithms mainly include the velocity obstacles (VO) algorithm, A* algorithm and the artificial potential field (APF) algorithm. Kuwata et al. [7] used the VO algorithm and combined with Convention on the International Regulations for Preventing Collisions at Sea (COLREGS) [8] to present an autonomous planning algorithm for unmanned surface vehicles (USV) to navigate safely in dynamic. The experimental results show that USV can better avoid obstacles and path planning, but there will be multiple minimum values, which leads to the situation that the optimal solution cannot be obtained. Petres et al. [9] uses the APF method to construct a virtual gravitational field to guide the ship to the target waypoint. By transforming the navigation restricted area into a virtual obstacle area and constructing a virtual repulsion field, this method realizes the obstacle avoidance and autonomous path planning in complex navigation environment. This APF has the advantages of simple algorithm structure and high computational efficiency, but it is easy to fall into local minimum and target point oscillation. Campbell et al. [10] proposed a real-time path planning method for USV based on an improved A* algorithm. The algorithm is integrated in the decision-making framework and combined with COLREGS. The results show that the method achieves real-time path planning for USV in complex navigation environments. However, this method relies on the design of the grid map, and the spacing and number of grids will directly affect the calculation speed and accuracy of the algorithm. Xue et al. [11] combined with the COLREGS international maritime collision avoidance rules, and proposed a path planning method based on the APF method. The experimental results show that this method can effectively achieve ship path search and collision avoidance. However, this method is difficult to deal with the problems of path planning and collision avoidance in an unknown and restricted navigation environment. The traditional non-intelligent algorithms usually have the advantages of simple structure and less computation, but in some cases, there are multiple minima, which cannot guarantee the optimal solution.

Bionic intelligent algorithms mainly include the ant colony optimization (ACO) algorithm and genetic algorithm (GA). Xin et al. [12] improved the traditional particle swarm optimization (PSO) algorithm through three methods: random group inversion, adaptive control acceleration coefficient and linear descent inertia weight method, and used the improved algorithm for path planning. Experiments show that the improved algorithm solves the premature problem of particle swarm optimization. Xin et al. [13] also introduced a strategy of increasing the number of offspring of GA algorithms by using multi-domain inversion, which effectively improved the local search capabilities of GA algorithms. Experiments show that the path planned by the improved algorithm is shorter and the convergence speed is faster. However, this method usually needs to continuously adjust the initial population parameters to avoid falling into the local optimum. Ding et al. [14] extracted the navigation environment information of ships in the electronic chart and carried out experimental modeling. The PSO algorithm was used to carry out unmanned ship path planning with path distance as the constraint condition. However, this method is prone to premature convergence and the loss of diversity of the population in the search space. Lazarowska et al. [15] showed a ship path planning method based on ACO algorithm. By changing the path planning problem to dynamic optimization, collision risk and voyage loss were taken as objective functions, and the optimal path and collision avoidance strategy were obtained under dynamic obstacle environment. However, such bionic intelligent algorithms usually need more prior knowledge and a large amount of calculation, which leads to the problems of long calculation time and local minimum, so it is mainly used in the auxiliary decision-making of path planning.

Machine learning methods mainly include neural network and reinforcement learning. Zhou et al. [16] proposed a USV autonomous path planning based on deep reinforcement learning (DRL). At the same time, the USV motion model and collision risk assessment strategy were integrated into the path planning. The algorithm was verified by experiments in the constructed simulation environment and the actual ocean environment. Chen et al. [17] introduced a path planning and maneuvering method for unmanned cargo ships based on Q-learning. This method obtains the optimal behavior strategy through reward guidance, and through continuous training and learning, the ship can obtain the optimal path. However, this method cannot solve the problem of high-dimensional space, large q-table may lead to slow convergence. Zhang et al. [18] offered an autonomous navigation decision model based on hierarchical DRL in order to realize ship path planning in port environment. The model mainly includes scene partition layer and navigation decision layer. The experimental results show that the improved DRL algorithm can effectively improve the safety and obstacle avoidance ability of ship path planning. Yu et al. [19] used the improved DRL method to solve the problem of trajectory tracking control of underwater vehicles. The method is composed of an action selection network and an action evaluation network, and constantly modifies the action strategy. The experimental results show that the algorithm can solve the problem of trajectory tracking of underwater vehicles in complex curves to a certain extent. In order to solve the problem of path planning and autonomous obstacle avoidance of maritime surface ships (MASS) in uncertain environments, Wang et al. [20] proposed a decision-making model of MASS behavior based on improved DRL was proposed. The APF is used to improve the network structure of DRL. The simulation results show that the path planned based on the improved DRL meets the actual navigation specifications, and improves the learning and adaptive obstacle avoidance ability of MASS. The DRL overcomes the shortcomings of other algorithms, such as the large amount of computation and the need for a certain number of prior samples. At the same time, it also has strong learning ability and stability. However, there will be the problem of training time stamp and slow convergence speed.

The Markov decision process (MDP) is a typical decision process model. Usually in MDP, the decision maker (Agent) relies on a certain action strategy to perform an action in the environment, and the environment generates a new state and reward for the corresponding action after the Agent performs the action. Among them, a mapping of Agent from state to action is called action strategy π_θ, $\pi_\theta(a_t|s_t)$ represents the probability of Agent choosing action a_t under the condition of state s_t at a certain moment. The reward R can also be called reward value, which is a kind of feedback information from the environment to the action, which tells the Agent the quality of its action selection. Therefore, reinforcement learning [21] in discrete time can essentially be regarded as MDP. The principle of reinforcement learning is shown in Figure 1.

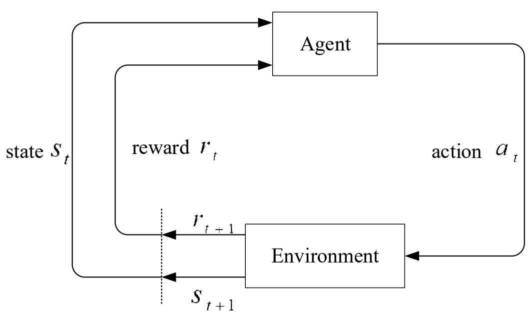


Figure 1. Reinforcement learning schematic.

The coastal ship path planning problem can be defined based on the sequential decision-making problem. The MDP suitable for ship path planning can be defined by the following elements: (S, A, P, R, γ). Among them, S represents the state information of the ship in a limited space; A represents the action space that the ship can perform, that is, the collection of all the behavior spaces of the ship in any state; P represents the probability that the ship will reach the next state after choosing action, which can be expressed as a conditional probability formula $P(s, s') = P(s'|s, a)$; R represents the reward function, the real-time reward obtained after the ship chooses action; γ represents the reward discount factor, and the return value at the next moment is attenuated according to this factor, where $\gamma \in (0, 1)$. It can be seen that the process of ship' path planning is a learning and decision-making process of strengthening learning. The ship selects an action according to the current state, which will affect the environment, and receives the feedback (reward or punishment) from the environment. The ship selects the next action according to the current feedback information and action strategy, and enters into the new environmental state. The principle of action selection is to maximize the positive feedback given by the environment.

DRL combines the perception ability of deep learning with the decision-making ability of reinforcement learning, and can realize direct control from the original input to the output through the end-to-end learning method, which has strong versatility [22]. Among them, the most representative is the DQN algorithm [23]. In the case of unknown environmental information, it can obtain reasonable behavior decisions by virtue of self-learning ability, and has strong universality and learning ability. On the other hand, DQN algorithm combines convolutional neural network with Q-learning in traditional reinforcement learning. In order to alleviate the instability of value function representation in nonlinear network, DQN mainly improves the traditional Q-learning algorithm in three aspects: (1) Using experience replay. At each time step, the sample information obtained by the Agent interacting with the environment is stored in the experience buffer pool. During training, each time a small batch of samples are randomly selected from the pool and the stochastic gradient descent (SGD) algorithm is used to update the network parameters. This method greatly reduces the correlation between the samples and improves the stability of the algorithm. (2) The DQN algorithm uses neural network to approximate the current value function, and uses a single network to generate the target Q value, which becomes the target value network. After the introduction of the target value network, the Q value of the target value network remains unchanged for a period of time, which reduces the correlation between the current value network Q value and the target Q value, and improves the stability of the network. (3) DQN algorithm reduces the reward value and error term to a limited interval, ensuring that both the Q value and the gradient value are in a reasonable range. When solving various tasks based on state awareness, DQN uses the same set of network models, parameter settings and training algorithms, and has achieved good results, which fully shows that it has strong adaptability and stability [24].

DRL method is gradually being applied to the transportation field with its strong self-learning ability and function fitting ability, especially in areas such as unmanned driving and path planning. In the marine field, some scholars have applied DRL related algorithms to surface unmanned vehicle and other aspects and have achieved good results. Therefore, DRL has broad application prospects and room for expansion in the marine field.

However, there are few studies on path planning of coastal ships based on DRL. Part of the research focuses on the combined use of DRL and traditional algorithms, such as combining DRL with APF to get an algorithm model. Because a reasonably designed reward function in DRL plays a key role in the effect of the entire model, thus guiding the ship to learn autonomously to avoid obstacles and reach the target point. Therefore, the combination of DRL and traditional algorithms weakens the main role of the reward function to a certain extent, which may lead to failure to learn the optimal path planning strategy. On the other hand, some DRL based path planning models use the distance between the ship and the object as a constant reward value, which will lead to low learning efficiency and slow convergence speed of the algorithm. In order to solve the problems of slow convergence speed and long training period of traditional reward function, this paper combines the COLREGS and referring to the actual navigation rules of ships, by setting the potential energy reward of the target point to the ship, adding the reward area near the target point and adding the danger area near the obstacle, So as to improve the convergence speed and stability of the algorithms, and ensure the safety and economy of the coastal ship navigation process.

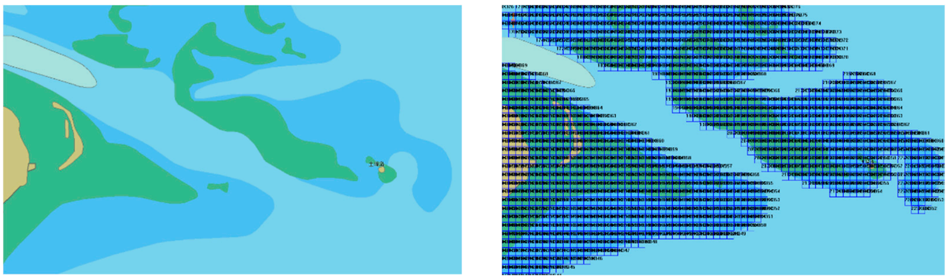
3. Model of Coastal Ship Path Planning

3.1. Environmental Status Information Processing

3.1.1. Gridding of Marine Environment

The grid method is a discrete modeling method. Its basic idea is to use a series of grids with a certain resolution to represent the environment, and mark these grids separately to complete the system modeling of the environment. The key of the grid method is to identify the map, because of its simple data structure and strong current, it is convenient for spatial analysis and surface simulation, and is conducive to the creation and maintenance of grid map. On the other hand, the size of the grid partition is negatively related to the amount of environmental information storage. When the grid partition is larger, the resolution of the environment will decline, and the optimal path may be missed in the dense obstacle environment. When the grid partition is smaller, the resolution of the environment is higher, and the ability to deal with the dense obstacles is stronger, but the online planning algorithm needs longer calculation time. At present, many researchers apply the grid method to ship path planning, and it is still one of the more widely used methods [25].

Figure 2 shows the process of using grid method to process marine environmental information. Among them, Figure 2a shows the state information before marine environment gridding, including the land and ocean information; Figure 2b shows the state information after marine environment gridding, and uses grid to label non navigable areas.



(**a**) Before gridding of marine environment. (**b**) After gridding of marine environment.

Figure 2. Gridding process of marine environmental: (**a**) before gridding of marine environment; (**b**) after gridding of marine environment.

In this paper, the grid method is used to treat the marine environment into a two-dimensional simulation ocean environment, and the specific values are used to represent the navigable area and the obstacle area, where the value 0 represents the navigable area, which is displayed as the white area in the simulation environment; the value 1 represents the obstacle, that is the non-navigable area, which is displayed as the black area in the simulation environment. In the two-dimensional coordinate system, each coordinate point corresponds to a state of the ship. There are only two states of the ship, navigable and prohibited. The marine environment information is extracted and processed as shown in Figure 3.

3.1.2. Ship Navigation in Accordance with COLREGS

The COLREGS must be considered before applying marine environmental information to path planning model, so as to ensure the practicability and effectiveness. Only according to COLREGS, the ship path planning can be applied to the actual navigation process. When a ship makes a decision from the perspective of the first person based on its environmental observation information, it is called "self-ship", while other ships and obstacles nearby are regarded as "other ships". When the self-ship's surrounding environment information is a navigable area, the ship should consider the economic cost and design the path according to the optimal way. When there is a non-navigable area around the ship, that is, there are obstacles around, the ship should avoid the obstacles safely under the premise of complying with COLREGS.

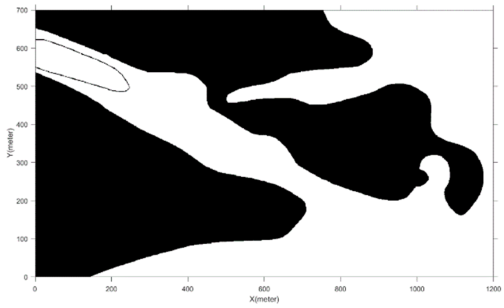


Figure 3. Marine environmental information extraction and processing.

According to COLREGS, the situation of ship encounter can be divided into three types: head-on situation, crossing situation and overtaking situation. In each case, each ship makes a decision according to the COLREGS, in which the give-way ship should take action to avoid collision, while the stand-on ship keeps its original state and continues to sail. When the marine environment has good visibility, the situation of ship encounter situation is shown in Figure 4.

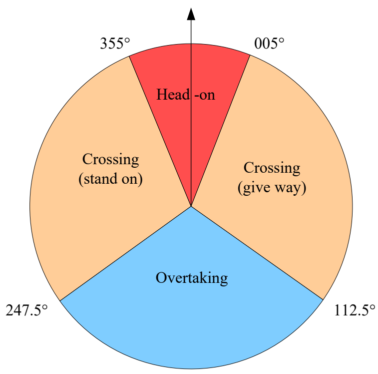


Figure 4. Ship encounter situation based on COLREGS.

As shown in Figure 4, taking the self-ship as the navigation center, the situation of ship encounter is specifically divided as follows: head-on situation has the regional range of (0°, 005°) and (355°, 360°). At this time, both sides of the meeting are giving way ships, and they should respectively turn to the right and more than 15 degrees. The crossing encounter situation includes (005°, 112.5°) and (247.5°, 355°). When the self-ship is located in the (005°, 112.5°) area, the ship belongs to the give-way ship, and at this time, it should take the right turning action to avoid collision with obstacles. When the self-ship is located in the (247.5°, 355°) area, the ship belongs to the stand-on ship, and the ship keeps the current path. The overtaking situation, the area is (112.5°, 247.5°). At this time, the self-ship belongs to stand-on ship, does not take action and keeps the current path.

On the other hand, in the case of restricted visibility at sea, there will be no obvious separation of responsibilities between stand-on ship and give-way ship. For this kind of sea situation, the COLREGS regulations that when the incoming ship is within the range of (0°, 90°) and (270° and 360°), the self-ship will turn to the right. For coming ships in the range of (90°, 180°) and (180°, 270°), the self-ship takes a turn towards other ships. In this article, we mainly study the situation of good visibility at sea.

3.2. Design of Coastal Ship Path Planning Model

3.2.1. Optimized Design of the Reward Function

The reward function is also known as immediate reward or enhanced signal. After the ship performs an action, the environment will make a feedback information according to the action, which is used to evaluate the performance of the action. The reward function is designed by the environment and the decision maker. It is usually a scalar, with positive value indicating reward and negative value representing punishment.

In the learning and training process of ship path planning model, the design of the reward function plays an important role, also determines the effect and efficiency of neural network training. The reward function can be used as the evaluation index of the effectiveness and safety of ship behavior decision-making, which has a result-oriented role. Traditionally, the reward function of the target to the Agent is usually defined as the fixed positive reward value when the next state of the ship is closer to the target point after the ship performs the action, otherwise, the fixed negative reward value is given. For the problem of ship path planning, the disadvantage is that it cannot know quantitatively the future impact of the current selected action on it, thus ignoring the optimal strategy, and will lead to low learning efficiency of the model and slow convergence speed of the algorithm.

In order to solve the problems of slow convergence speed and long training period of traditional reward function, this paper optimizes the reward function and proposes a new design method of the reward function. This paper mainly proposes three optimizations to the reward function. (1) The potential energy reward of the target point to the ship is set. (2) The reward area is added near the target point. (3) The danger area is added near the obstacle. Through the above optimized method, the ship can avoid obstacles to reach the target point faster, and the convergence speed of the model is accelerated.

1. The potential function reward of target point to ship is designed as follows:

$$r_t = \frac{d_t^{k-1} - d_t^k}{\left|d_t^{k-1} - d_t^k\right|} c^{\left|d_t^{k-1} - d_t^k\right|} \tag{1}$$

where, d_t^{k-1} is the distance from the ship to the target point at the $k-1$ moment; d_t^k is the distance from the ship to the target point at the k moment; c is a constant.

After each action, the dynamic reward is set according to the current state of the ship and the environment. According to Formula (1), when $d_t^{k-1} > d_t^k$, it means that the ship is closer to the target point after executing the current action. At this time, r_t is positive, which means that the environment gives the ship a positive reward. The value of positive reward changes exponentially with the difference between the ship's current time and the previous time from the target point, which can accelerate the ship approaching the target point, similar to the effect of potential energy. On the contrary, when the ship is far away from the target point, it will get a negative reward value.

2. The reward area near the target point is designed as follows:

The reward area means that when the ship is near the target point, but has not reached the target point, in order to help the ship quickly reach the target point, different rewards are given for different distances from the target point in the reward area. This method can speed up the convergence speed of the model. At the same time, in order to avoid falling into the local optimum, the reward in the reward area should not be too dense, and there should be a gap between the reward area and the reward reaching the target point. The design of reward field is shown in Formula (2).

$$r_y = \begin{cases} 2, & d_t^k > \partial \text{ and } d_t^k \leq \ell \\ 1, & d_t^k > \ell \text{ and } d_t^k \leq \lambda \end{cases} \tag{2}$$

where, ∂, ℓ and λ are the thresholds of reward area, and different rewards are given in different threshold ranges.

3. The dangerous area near the obstacle is designed as follows:

Dangerous area refers to that the ship is near an obstacle, but has not yet collided with the obstacle. In order to help the ship leave the obstacle quickly, it is necessary to increase the punishment near the obstacle. In order to avoid the ship falling into the local situation, the punishment in the dangerous area should not be too dense, and there should be a certain gap between the punishment in the dangerous area and that in the position of the obstacle. The dangerous area design is shown in Formula (3).

$$r_n = \begin{cases} -3, & d_o^k > \alpha \text{ and } d_o^k \leq \beta \\ -1.5, & d_o^k > \beta \text{ and } d_o^k \leq \delta \end{cases} \quad (3)$$

where, d_o^k represents the distance between the ship and the obstacle at the k moment, α, β and δ are the thresholds of the dangerous area, and different penalties are given in different threshold ranges.

Because the actual ship navigation process is continuous, the environment state and action space of DQN algorithm are limited. Therefore, combined with the above design of the reward function, the reward function is generalized to nonlinear piecewise function. The final design of the reward function R is as follows.

$$R = \begin{cases} 10, & d_t^k = 0 \\ -5, & s = 1 \\ r_t, & s = 0 \\ r_y, & s = 0 \\ r_n, & s = 0 \end{cases} \quad (4)$$

where, $s = 1$ indicates that the ship is in the non-navigable area, that is, the ship collides with the obstacles; $s = 0$ indicates that the ship travels in the navigable area; d_t^k represents the distance between the ship and the target point at time k, r_t represents the potential function reward value of the target point to the ship, r_y represents the reward value when the ship is near the target point, r_n represents the penalty value when the ship is near the obstacle.

3.2.2. Description of State Space and Action Space

The state space is used to represent the relevant characteristics of the ship's environment, which is usually used as the input information of DRL. In the process of navigation, the position of the self-ship relative to the target point and obstacles is always changing, which has uncertainty. In this paper, the input state information of the model comes from the real-time sensing information of ship sensors in electronic chart, including position information and azimuth information between ship and object.

Figure 5 shows the state space of the self-ship. The blue circle represents the self-ship, the red five-pointed star represents the target point, and the gray circle represents the obstacle area. The Cartesian coordinate system is established based on the environment of the self-ship. Assuming that the position of the ship in the environment is (x_s, y_s), the position of the target point is (x_t, y_t), and the position of the nearest obstacle to the ship is (x_o, y_o), d_t is the distance between the ship and the target point, and d_o is the distance between the ship and the obstacle. At the same time, the angle between the line from the ship to the target point and the positive half axis of the X axis can be calculated as φ_t, and the angle between the line between the ship and the obstacle and the positive half axis of the X axis is φ_o. Choosing $S = (d_t, \varphi_t, d_o, \varphi_o)$ as the state space of the ship in the environment can express the state of ship at any time in the environment.

The action space is used to describe what actions a ship can take in the current environment. The input of current value network is the state information of ship, and the output is the discrete action space Q value in DRL. In theory, the motion space of the ship in the environment can be in any direction, but too many actions will lead to a great increase in the training time of the model, resulting in the final path is too tortuous. On the other hand, too little action space will lead to the problem of "right angle" in ship planning path, which obviously does not conform to the actual navigation rules. Therefore, the selection of an appropriate action space has a great impact on the quality of path planning and training time.

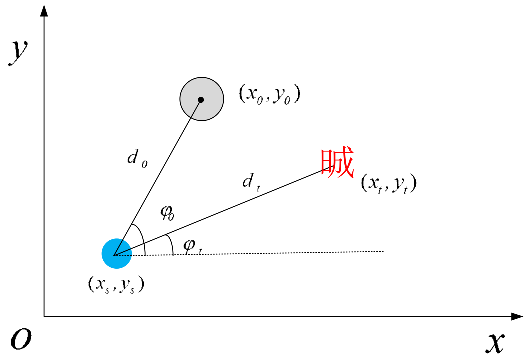


Figure 5. Schematic diagram of ship state space.

After setting the starting point and target point of the ship, the ship is regarded as a particle in the environmental. In the actual navigation process, the ship's navigation process is a continuous state, so the ship's action space is generalized into 8 discrete actions, the specific execution actions are up, down, left, right, upper left 45 degrees, right upper 45 degrees, left lower 45 degrees and right upper 45 degrees. Adding diagonal action can improve the ship's exploration of the corner situation in the environment, and avoid the increase of training round caused by too little action space. Formula (5) defines a set of motion spaces centered on the ship's position.

$$A = \{(r-1,c);(r+1,c);(r,c-1);(r,c+1);(r-1,c-1);(r-1,c+1);(r+1,c-1);(r+1,c+1)\} \tag{5}$$

In the above formula, A represents the set of action spaces, r represents the x-axis direction of the ship's two-dimensional coordinate system, and c represents the y-axis direction of the ship's two-dimensional coordinate system.

3.2.3. Action Exploration Strategy

In the training and learning process of DRL, it is necessary to deal with the relationship between exploration and exploitation. The exploration emphasizes the discovery of more information in the environment, not limited to the known information; the exploitation emphasizes maximizing the reward from the known information. Appropriate action exploration strategy can make the Agent try more new actions and avoid falling into local optimum.

The greedy strategy selects the action that maximizes the value function each time. However, this method does not consider the exploration. For the state-action values that have not appeared in the sampling, there is no corresponding Q value because there is no evaluation, and the action will not be selected later. Therefore, greedy strategy only focuses on exploitation, does not involve exploration, and stores disadvantages in action exploration.

The $\varepsilon - greedy$ strategy has both exploration and exploitation. It selects an action randomly from all action spaces with the probability of ϵ and chooses the greedy strategy with the probability of $1-\varepsilon$, that is to extract the state-action value with the highest average reward, so as to prevent the Agent from falling into the local optimal situation. Generally, ε gradually decreases with the increase of training times, which fully reflects that in the early stage of training, more attention is paid to the exploration of environment, while in the later stage of training, the use of high-quality movements is gradually emphasized.

In the training process of ship path planning model, on the one hand, the ship needs to try and error to obtain the optimal search strategy, that is, exploration; on the other hand, it needs to consider the whole path planning, so as to maximize the expected value of the model to obtain future rewards, that is, exploitation. In this paper, $\varepsilon - greedy$ is used as the action exploration strategy. When the search behavior maximizes the state-action value function, the probability of selecting this action is $1-\varepsilon$. Otherwise, the probability of selecting other random actions is ε. The ship exploration and exploitation strategy is shown in Formula (6).

$$a = \begin{cases} \text{argmax} Q(s,a), & \text{with probability } 1-\varepsilon \\ \text{random}, & \text{otherwise} \end{cases} \quad (6)$$

where, a represents the action of ship selection, $Q(s,a)$ represents the state-action value function under state s and $\varepsilon \in (0,1]$, while in other cases, an action will be selected randomly under the probability of ε.

3.2.4. Training and Learning Process of Model

In order to solve the problem of ship path planning in the coastal environment, this paper proposes an optimized DQN algorithm to solve the problem. Figure 6 shows the schematic diagram of path planning for coastal ships based on optimized DQN algorithm. Firstly, the sensor system is used to obtain the environmental information of the ship, and combined with COLREGS to ensure the safety and effectiveness of ship action. Secondly, the current environment state information of the ship is taken as the input of the current value network, and the behavior based on the current policy is generated through training, and the action corresponding to the maximum state-action value $\text{argmax}_a Q(s,a;\theta)$ is selected. Then, the ship performs the action and obtains the corresponding reward value r, and stores the current state s, action a, next moment state s' and reward r as historical experience information in the experience replay memory. At each time, a number of sample data are randomly selected for current value network training, so that the trained current value network can fit the optimal action value. Finally, the loss function is calculated by combining the Q value of the current network and the target network, and the network parameters are updated by using historical experience data. When the current value network is trained enough, its weight will approach the best parameter. At the same time, the current value network copies its own network parameters to the target network every N time steps to reduce the correlation between the two networks.

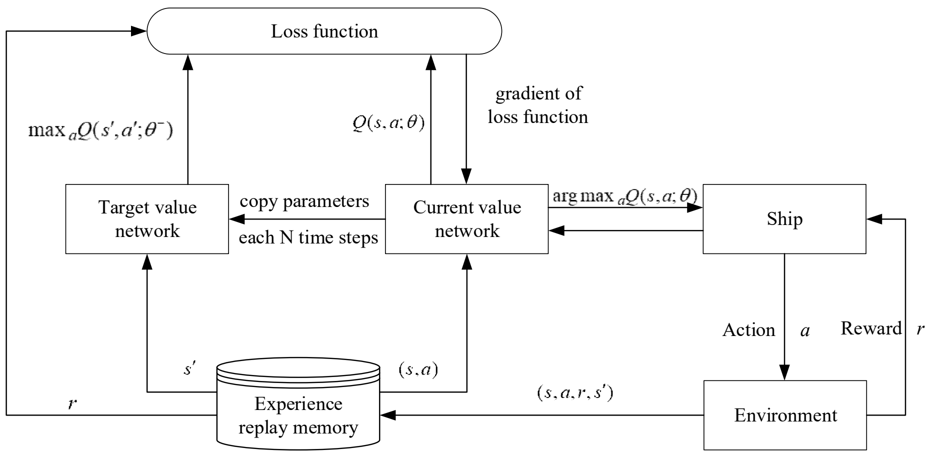


Figure 6. Schematic diagram of coastal ship path planning based on optimization deep Q network (DQN) algorithm.

In summary, the coastal ship path planning model is mainly divided into the following stages: first, before the training and learning, it is necessary to design the environmental state, ship action, action strategy and reward function properly; secondly, the model obtains a maximum future return strategy in the interaction process through learning, and constantly modifies it to the optimal strategy through the value function; Finally, when the training and learning process is completed, the optimal control strategy based on state information is obtained, the ship can use this model to avoid all obstacles and plan an optimal path in line with the navigation rules to guide the ship to safely travel from the starting point to the target point.

3.3. Execution Process of Coastal Ship Path Planning Model

In this paper, the real marine environment is abstracted, and the optimized DQN algorithm is used to plan the path that meets the navigation specification. The information of ship and environment is collected by sensors (marine radar equipment), and the distance and angle between the ship and the target point and obstacle are taken as the input data of the algorithm. Through the model training learning, the cumulative reward income in the learning process is maximized, and the optimal action strategy is finally determined. When the model training is completed, the ship can avoid the obstacles and reach the destination, planning a safe and economic path.

The execution process of coastal ship path planning model is described as follows:

1. Firstly, the ship's environment is processed, and the state information of the ship and the environment is obtained by the sensing equipment.
2. Call the model and take the state information as the input data of the model. After the model calculation, we can get the action that the ship should take under the current state.
3. The ship controller obtains the action and executes the action in the current ship state.
4. Obtain the state information of the next moment after the ship performs the action, and determine whether the ship state after execution is the end state.
5. If the current state is not the end state, the ship state information at the next moment will be handed over to the model for further use. If the current state is in the end state, it indicates that the ship has reached the target point, and the calculation and call of the model are finished.

The pseudo code of path planning for coastal ships is as follows (Algorithm 1).

Algorithm 1. DQN algorithm for path planning of coastal ships

1. Initialize replay memory D
2. Initialize action-value Q current value network with random weights θ
3. Initialize action-value $\hat{Q}$ target value network with random weights $\theta^{-} = \theta$
4. For episode = 1 to M do
5. Input initial ship and environment observation states:
6. Input initial unmanned ships and environment observation states sequence:

$$s_1 = \left(d_t^1, \varphi_t^1, d_o^1, \varphi_o^1\right)$$

7. For t = 1 to T do
8. With probability ε select a random action a_k, otherwise select $a_k = \text{argmax}Q(s, a; \theta)$
9. Ship execute action a_k in environment and calculate the reward r of k time
10. Obtain the state of the ship at k +1 time: $s_{k+1} = \left(d_t^{k+1}, \varphi_t^{k+1}, d_o^{k+1}, \varphi_o^{k+1}\right)$
11. Store (s_k, a_k, r_k, s_{k+1}) transition in D
12. Sample random mini-batch $D_{\min} = (s_i, a_i, r_i, s_{i+1})$ of transitions from D
13. Calculate target value function y_k:

$$y_k = \begin{cases} r_k, & \text{if episode terminates} \\ r_k + \gamma \text{max}_{a'} \hat{Q}(s_{k+1}, a_k'; \theta^{-}), & \text{otherwise} \end{cases}$$

14. Perform a gradient descent step on $(y_k - Q(s_k, a_k; \theta))^2$ with respect to the current value network parameters θ
15. Every C setps reset $\hat{Q} = Q$

$$\theta^{\mu'} \leftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'}$$
$$\theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'}$$

16. End for
17. End for

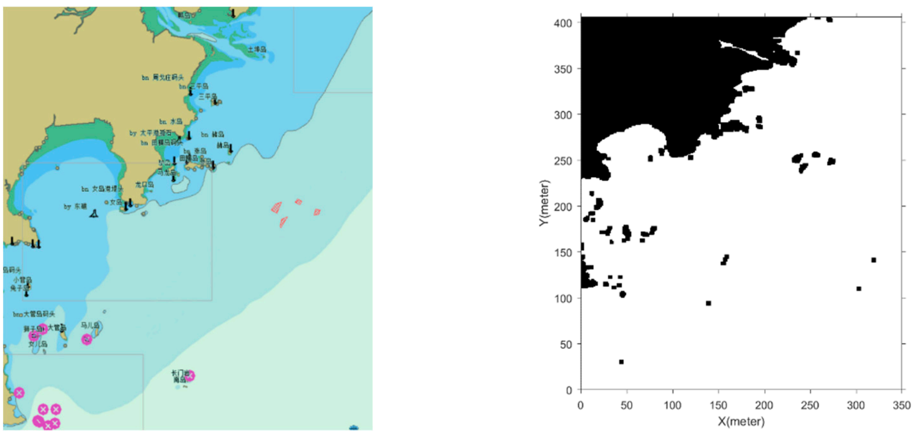
4. Experimental Verification and Result Analysis

In this section, the reliability and effectiveness of the path planning model of coastal ships are verified by simulation experiments. It mainly includes the establishment of training experimental environment and the setting of algorithm parameters, the model prediction of path and the smooth processing of path.

4.1. Experimental Environment Construction and Algorithm Parameter Setting

Based on the electronic chart platform, this paper selects the real sea environment as the environment space of model training. Using the grid method in Section 3.1.1, the marine environment is rasterized with 0.1 nautical mile as the grid size, and finally the marine environment is quantified as "0" and "1" format data. Then, a 400 $\times$ 350 two-dimensional simulation environment is built, based on the quantized data in Python. In this two-dimensional coordinate system, each coordinate position corresponds to the quantized marine environment information, each coordinate can be mapped to each element of the environmental state, which can be used to represent the navigable area and no navigation area. At the same time, in the simulation environment, each coordinate position uses 0 or 1 to represent the current environment state value, where 0 represents the navigable area of the ship, which is displayed as a white area in the environment model; 1 represents the obstacle, which is the non-navigable area, which is displayed as a black area in the environment model.

Figure 7 shows the two-dimensional simulation training marine environment required for the experiment. Figure 7a shows the real marine environment information, including the information of the ocean, shore-based islands and sunken ships. Figure 7b shows the corresponding simulation experiment environment information of the real marine environment after processing by the grid method. On the other hand, the safe driving distance of the ship is related to the size of the ship itself. Large ships often need a long safe distance and take action earlier. Therefore, this paper expands the obstacles in the marine environment, and increases the boundary of the obstacles by 0.1 nautical miles on the basis of the original proportion.



(**a**) Real marine environment. (**b**) Simulation experiment environment.

Figure 7. Two-dimensional simulation training marine environment: (**a**) real marine environment; (**b**) simulation experiment environment.

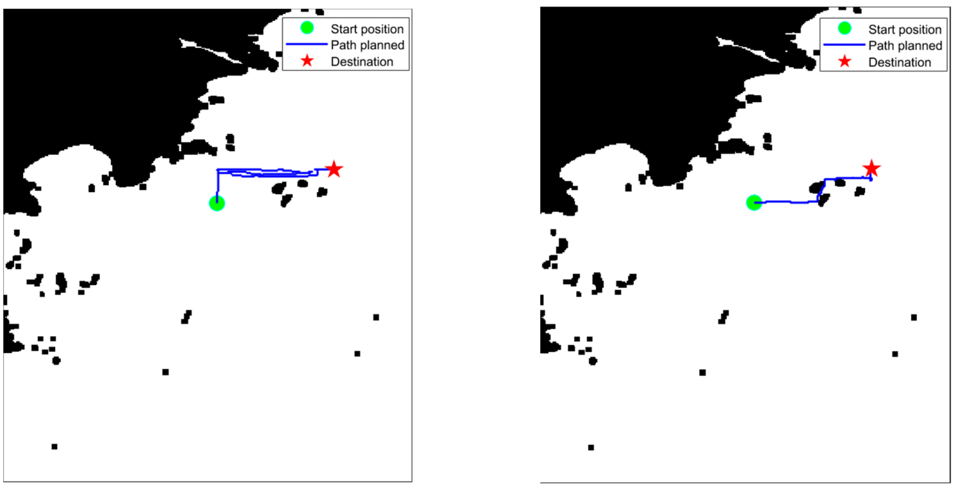
The goal of the model is composed of two parts: goal orientation and action decision. When there are no or few obstacles in the marine environment, the action decision made by the model will guide the ship to approach the target point quickly until it reaches the target point. When the obstacles in the marine environment affect the ship approaching the target point, the model helps the ship to avoid the obstacles and move towards the target point by action decision-making, and improves the quality of action decision-making through the interaction between the ship and the environment and the constraints of rules in the navigation process. The parameter settings of the model in the training process are shown in Table 1. Among them, only a part of the neurons in the activation function ReLu will be activated at the same time, which makes the network very sparse and efficient. Therefore, the sparse model implemented by ReLu can better fit the data and learn the relevant features autonomously. Because the activation function ReLu has the advantages of simple calculation, high training efficiency and overcoming the gradient disappearance, it is selected as the activation function of the neural network of the algorithm. It can better enhance the ship's autonomous learning ability, speed up the model training speed, and realize the stability and effectiveness of path planning.

Table 1. Model training parameter definition.

Parameters Name	Parameters Value	Description
Action space size	8	Optional action of the ship
Learning rate α	0.01	Learning rate of neural network
Decay factor γ	0.9	Decay factor of cumulative reward
Exploration rate ε	1	Exploration rate of action
Explore decay rate	0.995	Exploration decay rate of action
Experience replay memory D	10,000	Stores historical experience data
Sample size $D_{\min}$	512	Size of extracted empirical data
Hidden layers size	3	Size of hidden layers in networks
Number of neurons	128	Number of neurons in hidden layer
Activation function	ReLu	Neuron activation function

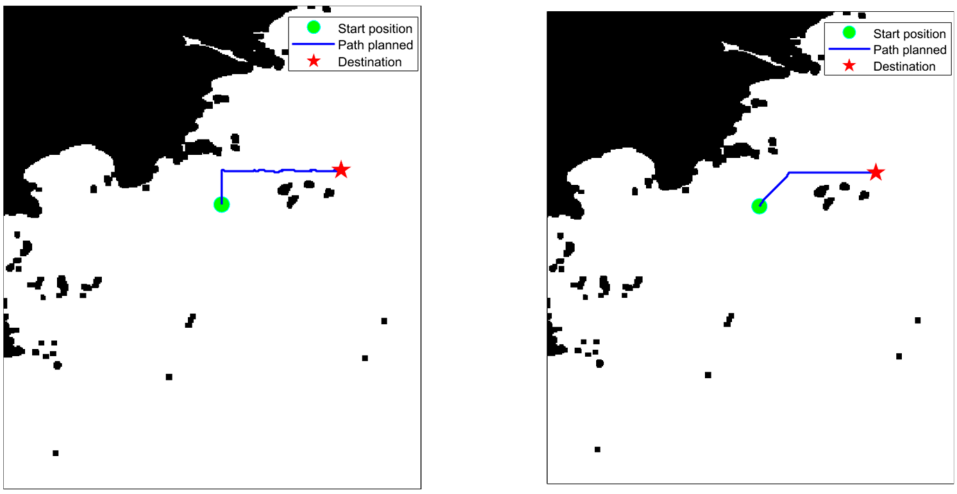
4.2. Model Validation Results

In order to verify the effectiveness of the coastal ship path planning model based on optimized DQN, this chapter introduces the ship path planning results under different training iterations, as shown in Figure 8. The black area represents the type of obstacle, that is area where the ship is prohibited from sailing, and the white area represents the navigable sea area. At the same time, the initial position of the ship is set as (183, 239) and the green dot is used to indicate the initial position. The target point position is (283, 268) and the target position is indicated by a red five-pointed star. The planned path of the ship is represented by a solid blue line.



(**a**) 500th iteration. (**b**) 1000th iteration.

Figure 8. *Cont.*

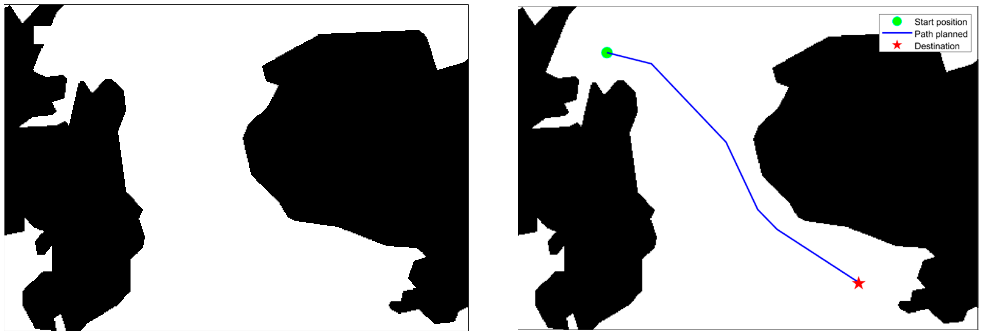


(**c**) 2000th iteration. (**d**) 3000th iteration.

Figure 8. Path planning results under different iterations. (**a**) 500th iteration; (**b**) 1000th iteration; (**c**) 2000th iteration; (**d**) 3000th iteration.

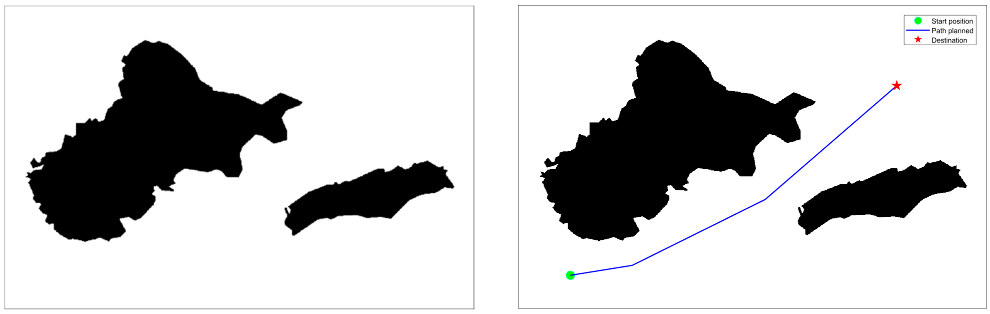
As shown in Figure 8a, in the initial 500 iterations, although the planned path finally reached the target point, there were more turn back paths. This is because the DRL algorithm in training is through constant trial and error behavior, so as to find the optimal solution, so there will be more trial and error behavior in the early stage of training. On the other hand, ships constantly exploring in the early stage, which cannot accurately judge the obstacles and target areas in the environment, and there are many exploration behaviors when facing obstacles. Therefore, it can be seen from the figure that there are many turn back phenomena in the path planned by the model. Figure 8b shows the prediction result after 1000 iterations. The model gradually plans the path with less redundant paths and reaches the target point. However, the planned path passes through some obstacles and collides with obstacles many times in the process, which obviously does not meet the requirements of the ship safety regulations. Figure 8c shows the prediction result after 2000 iterations. After continuous self-learning, the predicted path is guaranteed in terms of safety, and the collision phenomenon is gradually reduced, and finally it reaches the target position successfully. However, the planned path still fluctuates greatly, and the corner angle is too large to meet the actual navigation requirements. Figure 8d shows the result after 3000 iterations. The ship does not pass through the obstacles and successfully avoids all obstacles, and finally reaches the target point. At this time, the planned path fluctuation is weak, and gradually tends to be stable. This is because the random exploration rate of action space is reserved, so there is slight fluctuation in the whole path.

In order to verify the credibility and effectiveness of the model, this paper selects other complex marine environment for the experimental verification of ship path planning. Figure 9 shows the path planning results of the model in marine environment case 1. Figure 9a shows the initial environment of marine environment case 1, and Figure 9b shows the path planning results after 3000 iterations. The starting point coordinates of the ship are (86, 265) and the target point coordinates are (326, 45). It can be seen from Figure 9b that the trajectory planned by the model avoids the obstacles and keeps a safe distance from the obstacles, and finally reaches the target point safely. Figure 10 shows the path planning results of the model in marine environment case 2. Figure 10a shows the initial environment of marine environment case 2, and Figure 10b shows the path planning results after 3000 iterations. The starting point coordinates of the ship are (110, 70) and the target point coordinates are (800, 470). It can be seen from Figure 10b that the path planned by the model can reach the target position safely, which is in line with the specifications and requirements of the actual navigation. Through the above experimental verification and analysis, it can be seen that the model can better plan the path in line with the actual navigation specifications in different marine environment.



(**a**) Initial marine environmental information (**b**) 3000th iteration in case 1

Figure 9. Path planning results in case 1 of marine environment. (**a**) Initial marine environmental information; (**b**) 3000th iteration in case 1.



(**a**) Initial marine environmental infomation (**b**) 3000th iteration in case 2

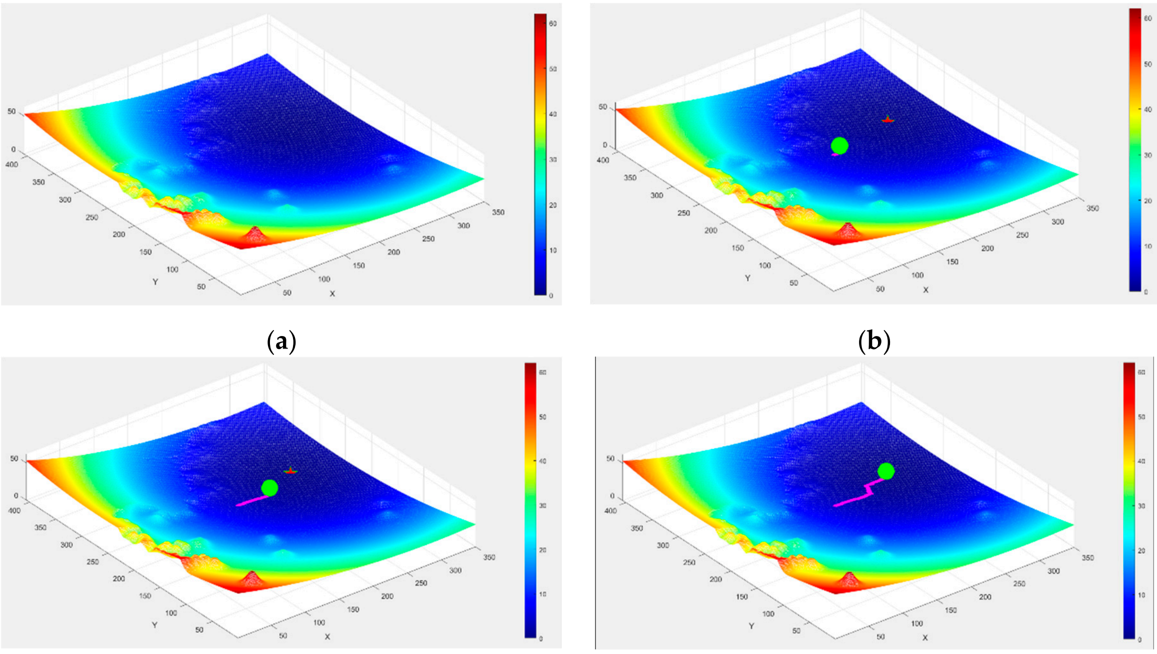
Figure 10. Path planning results in case 2 of marine environment. (**a**) Initial marine environmental information; (**b**) 3000th iteration in case 2.

On the other hand, this research applies result of path planning to the obstacle potential map in three-dimensional environment, as shown in Figure 9. Different colors in the figure indicate the level of potential energy. In the figure, the red represent positive potential energy and it corresponds to the relevant obstacles. The potential energy of 0 in blue represents the navigable sea area. The higher the potential energy value is, the larger the influence range of obstacles is, and vice versa. The interaction process between the actual predicted path and the marine environment can be better observed from the graph.

Figure 11a,b shows the initial obstacle potential images in the actual marine environment, as well as the initial positions of the ship's starting point and target point. Figure 11c shows the ship's course along the path predicted. It can be seen that the current ship does not reach the area with high potential energy value, that is, there is no collision with obstacles. Figure 11d shows the effect picture of the path predicted by the model. The predicted complete path does not reach the area with high potential energy value, that is avoids the obstacle area and finally reaches the target safely.

4.3. Path Smoothing

Trajectory data compression algorithms are generally divided into two categories: one is to linearize the motion trajectory by segments [26], which is the most commonly used algorithm due to its simple algorithm form and low computational complexity, the other is nonlinear trajectory fitting [27], which is closer to the real track, but faces the problems of complex algorithm and large amount of calculation. Because the ship's trajectory is along the established path and the real path is composed of several channel points, the linear compression algorithm is more suitable for the ship's actual path, and also can save some computing resources. Among many linear compression algorithms, the Douglas–Peucker algorithm is the most representative.



(**a**) (**b**) (**c**) (**d**)

Figure 11. Combination of obstacle potential and DQN-based ship path planning. (**a**) Initial marine environment of obstacle potential; (**b**) Initial position of starting point and target point; (**c**) Model prediction path process; (**d**) Model prediction end of path.

4.3.1. Douglas–Peucker Algorithm

The Douglas–Peucker algorithm, referred to as DP algorithm, is an algorithm that approximately represents a curve as a series of points and reduces the number of points [28]. It was proposed by Douglas and Peucker in 1973 and perfected by other scholars in the following decades.

The basic idea of DP algorithm is as follows: the first and last positions of each curve are connected by imaginary connecting lines, the distance between all points and the straight line is calculated, and the maximum distance value $d_{\max}$ is found. Compared with the limit threshold $d_{\max}$ by $D_{\max}$, the appropriate points are retained and the points that do not meet the requirements are removed. The steps of the algorithm are as follows:

1. Connect a straight line, AB, between two points A and B at the beginning and end of a trajectory curve, which is the chord of the curve, traverse all other points on the curve, calculate the distance from each point to the line AB, find the point C with the maximum distance, and mark the maximum distance as $d_{\max}$.
2. The distance $d_{\max}$ is compared with the pre-defined threshold $D_{\max}$. If $d_{\max} < D_{\max}$, the AB curve segment is approximated and the curve segment is processed.
3. If $d_{\max} > D_{\max}$, the curve AB is divided into AC and CB at point C, and the two sections are processed in steps (1) (2) respectively.
4. When all the curves are processed, the broken line formed by connecting each segmentation point in turn is the path of the original curve.

The DP algorithm has a simple structure and high computational efficiency. It is a post compression method which needs to define the starting point and target point, and is an offline compression method. In the aspect of ship path planning, the initial path obtained by the planning is smoothed by DP algorithm, which will be more suitable for the requirements of ship's navigation trajectory, and improve the safety and economy of navigation. This paper will further optimize the path predicted by model based on DP algorithm, so that the final path meets the actual navigation needs.

4.3.2. Path Smoothing Based on DP Algorithm

The path planning result retains the random action exploration in order to fully explore the environmental information, so there are some redundant path corners. However, in the actual navigation process, unnecessary turning action should be avoided to reduce operational risk and improve economy. Therefore, it is necessary to smooth these improper path corners.

In this chapter, the path planned by the model is combined with DP algorithm to smooth and optimize the path results, so as to improve the safety and economy of ship driving. Figure 12 shows the path optimized by DP algorithm. It can be seen from the figure that the optimized path removes redundant corners, and the overall path is smoother, which is in line with the actual navigation specifications. Figure 13 shows the actual ship path planning results of the optimized path in the electronic chart.

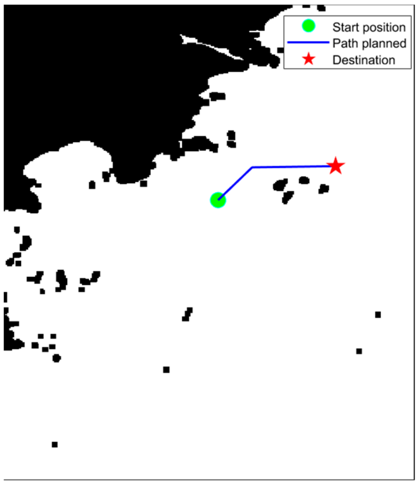


Figure 12. The path optimized by the Douglas–Peucker (DP) algorithm.

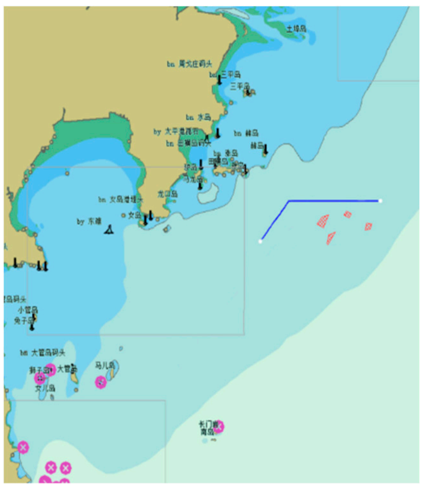


Figure 13. Results of path planning in electronic chart.

Table 2 shows the number of path corners and path length before and after optimization using DP algorithm. From the data in the table, it can be seen that the optimized path has better performance in the number of path corners and path length, thus improving the economy and safety of ship navigation.

Table 2. The number of path corners and path length before and after optimization by DP.

	Before Optimization	After Optimization
Number of path corners	6	1
Path length/meter	137.612	111.318

4.4. Comparative Analysis of Experimental Results

In this section, by comparing the traditional DQN algorithm with the algorithm proposed in this paper, the number of training iterations and steps of the model are compared and analyzed, as shown in Figure 14.

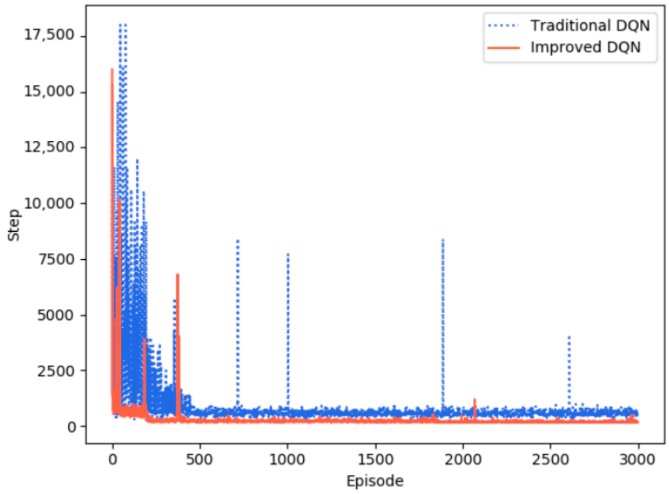


Figure 14. The convergence trend comparison results of step iteration.

The abscissa in the figure represents the number of training iterations, the ordinate represents the number of steps required from the starting point to the target point of each iteration. The blue dotted line represents the iterative trend result of traditional DQN algorithm, and the red solid line represents the iteration trend result of the algorithm proposed in this paper. The graph clearly and intuitively shows the convergence speed and training effect of the two algorithms. It can be seen from the figure that the number of round steps of the proposed algorithm began to decrease around the 200th round, showing a gradual convergence trend, while the traditional DQN algorithm began to show a convergence trend around the 450th round. At the same time, it can be seen that the number of round steps of the traditional DQN algorithm is maintained at about 138 steps and does not converge to the minimum number of steps, but the proposed algorithm has converged to the minimum number of steps of about 103 steps, which shows that the path calculated by the traditional DQN algorithm has more redundancy. In the later training process, the proposed algorithm has smaller fluctuation frequency and better stability than the traditional DQN algorithm. Experimental results show that the proposed algorithm has faster iteration speed and better decision-making ability, and can quickly reach the target point with less steps.

On the other hand, the length of path trajectory and the number of path corners determine the safety and economy of path planning, and the time of path planning also determines whether the path can be used in time. In this research, the optimized DQN algorithm is compared with other path planning algorithms from three aspects: path length, number of corners and planning time.

The other path planning algorithms used in this paper include the following six: traditional DQN algorithm, Q-learning algorithm [17], deep deterministic policy gradient (DDPG) algorithm [19], A* algorithm, BUG2 algorithm and APF algorithm. In the same marine environment, this paper analyzes and compares the experimental results and performance data of the above algorithms. Among them, the action space of Q-learning algorithm and traditional DQN algorithm is the same as that of the algorithm proposed in this paper. The action space of the DDPG algorithm is ($-35°$ and $35°$) and the continuous action output is obtained.

Figure 15 shows the path planning result of different algorithms in the same marine environment. Figure 15a is the path planning result of 3000 rounds of training based on the traditional DQN algorithm. It can be seen from the figure that the ship can autonomously learn to plan a safer path, but the path has more redundant corners, which increases the risk of driving the ship and does not conform to the actual navigation specifications of the ship meet the requirements of path economy. Figure 15b is the path planning result of 3000 rounds of training based on the traditional Q-learning algorithm. It can be seen from the figure that the ship finally plans a path, but there are still many redundant corners in the path, and the algorithm training takes a long time due to the size limitation of Q-table. Figure 15c is the path planning result of 3000 rounds of training based on the traditional DDPG algorithm. It can be seen from the figure that the path planned is a curve. This is because the result obtained by DDPG algorithm is the continuous action value, so the action taken by the ship changes in real time. Because the coastal ships need to take as few actions as possible in the process of navigation, the path obtained by this method does not meet the actual navigation requirements. Figure 15d is the result of path planning based on A* algorithm. It can be seen from the figure that the planned path is closer to the obstacles, and there are more path corners, which increases the risk of ship driving and is not applicable to the actual navigation specifications. Figure 15e is the result of path planning based on BUG2 algorithm. Since the obstacle information in the environment is required to be polygon when planning path, the obstacle is filled as a circle circumscribed square in the experiment. When planning the path, the algorithm needs to circle around the obstacle to determine the point closest to the target point. It can be seen from the figure that the planned path has a large angle point, not in line with the actual navigation rules of the ship. Figure 15f is the result of path planning based on the APF algorithm. The algorithm regards the whole environment as a large magnetic field. The target point generates a gravitational magnetic field on the ship, and the obstacle generates a repulsive magnetic field on the ship. It can be observed from the figure that although the trajectory of the path planned based on APF is short and has a certain distance from the obstacle, the planned path has radian and is not suitable for the actual navigation of the ship.

By comparing the above algorithm with the path planned in this paper (Figure 8d). The path planned by the proposed algorithm is smoother, has less path corners, and keeps a certain safe distance from obstacles, which is more in line with the actual navigation specifications of ships.

Meanwhile, based on the above experimental data, the performance of different algorithm is compared and analyzed in terms of path length, path planning time and corner number. Table 3 shows the comparative information of experimental data. It can be observed from the table that the path length planned by this paper proposed algorithm is 137.612 m, taking 0.6105 s, and 6 path corner; the path length planned by the traditional DQN algorithm is 154.241 m, taking 0.9254 s, and the path corner is 16. The path length planned by the Q-learning algorithm is 146.617 m, taking 0.7613 s, and the path corner is 6. The path length planned by the traditional DDPG algorithm is 172.315 m, taking 0.8251 s, because the path also is arc, the corner is not calculated. The path length planned by the A* algorithm is 116.421 m, taking 1.2253 s, and the path corner is 28. The path length planned by BUG2 algorithm is 92.728 m, which takes 2.3268 s, and the path corner is 6. The path length planned by APF algorithm is 135.513 m and takes 1.6216 s, because the path also is arc, the corner is not calculated. From the above experimental results and data analysis, it can be seen that under the premise of meeting the actual navigation specifications, the algorithm proposed in this paper has shorter path length and time consumption, as well as fewer corners, and has better actual path planning effect.

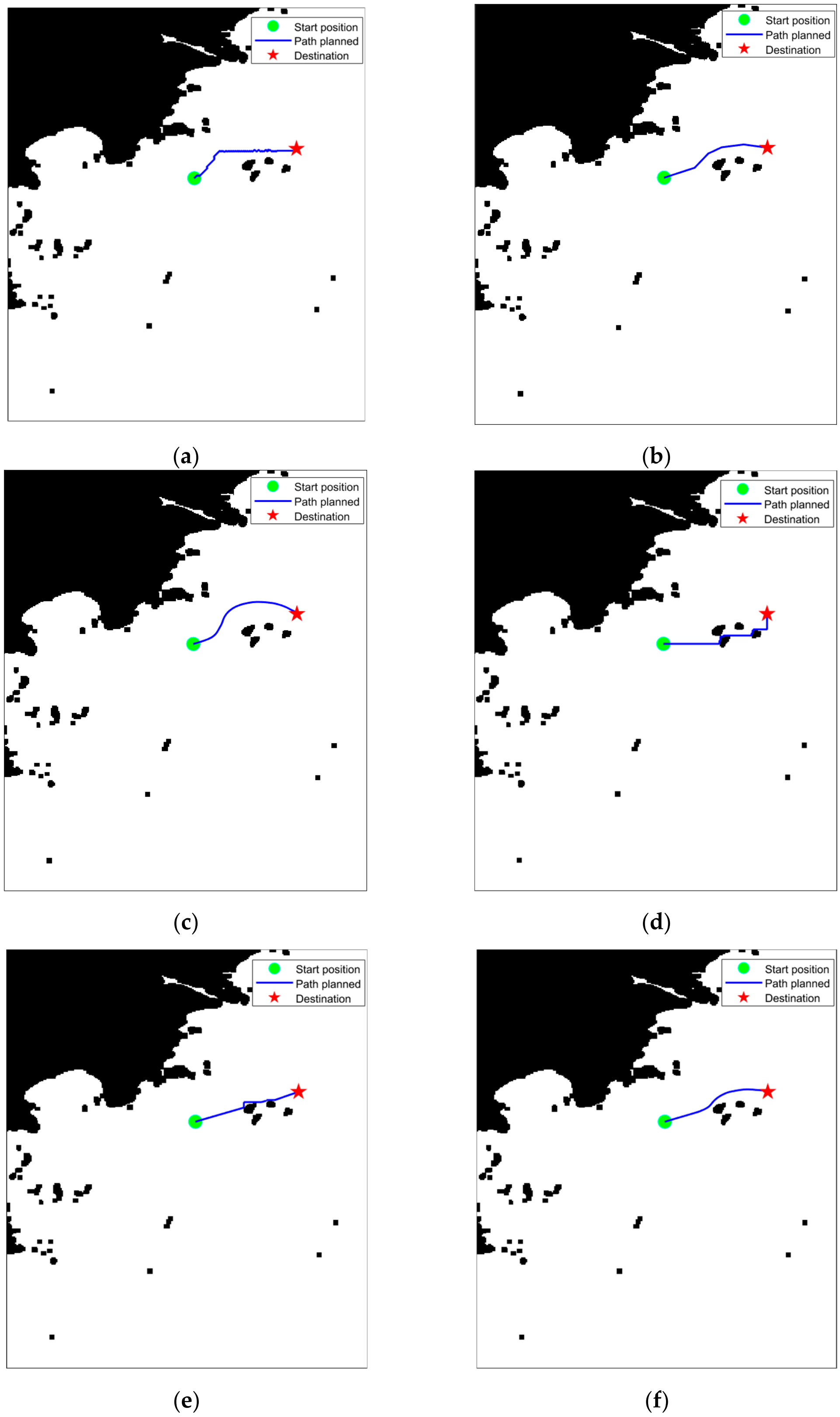


Figure 15. Comparison of path planning results under different algorithms. (**a**) Path planning based on traditional DQN; (**b**) Path planning based on Q-learning algorithm; (**c**) Path planning based on DDPG algorithm; (**d**) Path planning based on A* algorithm; (**e**) Path planning based on BUG2 algorithm; (**f**) Path planning based on APF algorithm.

Table 3. The comparison of experimental data.

Method	Path Length/Meter	Time/s	Number of Path Corners
Optimized DQN	137.612	0.6105	6
Traditional DQN	154.241	0.9254	16
Q-learning	146.617	0.7613	6
DDPG	172.315	0.8251	-
A*	116.421	1.2253	28
BUG2	92.728	2.3268	6
APF	135.513	1.6216	-

5. Conclusions

For the traditional path planning algorithm, due to the lack of autonomous learning ability and historical experience, data cannot be recycled, resulting in a slow convergence speed and the actual planning path is not smooth and so is redundant. In this paper, a global path planning model for coastal ships based on an optimized DQN algorithm is proposed, and the path planning problem is divided into three parts: environment status processing, path search and smooth path. The traditional reward function generally adopts the idea of fixed reward value, but this method will lead to slow convergence speed of traditional DQN algorithm and easy to fall into local iteration problem. In order to solve this problem, this paper optimized the reward function. By setting the potential function reward of the target point, adding the reward area near the target point and the penalty area near the obstacle, a ship path planning model based on optimized DQN is established. Firstly, the actual marine environment information for algorithm training is obtained based on a grid method. Secondly, the structure of the model is designed according to the ship navigation rules, and the reward function is optimized. Through learning and training, the experimental results show that the coastal ships take reasonable actions under the premise of path specification, successfully complete the autonomous path planning, and realize the end-to-end learning method of ships. Finally, the proposed algorithm is compared with other algorithms. The results show that the optimized DQN algorithm has a fast convergence speed, high accuracy and small navigation error. At the same time, it has a shorter planning time and more secure and reliable path results, which further verifies the effectiveness of the method. However, this paper does not consider the marine environment where static obstacles and dynamic obstacles exist simultaneously in an actual verification environment. Considering the path planning of ships in more complex sea areas, and verifying it in the actual environment is the focus of the next research in this paper.

Author Contributions: Conceptualization, S.G.; methodology, X.Z. and S.G.; software, S.G. and Y.Z.; writing—original draft preparation, S.G. and Y.D.; writing—review and editing, X.Z. and S.G.; resources, Z.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Key R&D Program of China (Grant No. 2018YFB16-01502) and the LiaoNing Revitalization Talents Program (Grant No. XLYC1902071).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Roberts, G.N.; Sutton, R.; Zirilli, A.; Tiano, A. Intelligent ship autopilots—-A historical perspective. *Mechatronics* **2003**, *13*, 1091–1103. [CrossRef]
2. European Maritime Safety Agency (EMSA). *Annual Overview of Marine Casualties and Incidents*; EMSA: Lisbon, Portugal, 2018.
3. Norstad, I.; Fagerholt, K.; Laporte, G. Tramp ship routing and scheduling with speed optimization. *Transp. Res. Part. C Emerg. Technol.* **2011**, *19*, 853–865. [CrossRef]

4. Tsai, C.-C.; Huang, H.-C.; Chan, C.-K. Parallel Elite Genetic Algorithm and Its Application to Global Path Planning for Autonomous Robot Navigation. *IEEE Trans. Ind. Electron.* **2011**, *58*, 4813–4821. [CrossRef]
5. Lyu, H.; Yin, Y. COLREGS-Constrained Real-time Path Planning for Autonomous Ships Using Modified Artificial Potential Fields. *J. Navig.* **2019**, *72*, 588–608. [CrossRef]
6. Boschian, V.; Pruski, A. Grid modeling of robot cells: A memory-efficient approach. *J. Intell. Robot. Syst.* **1993**, *8*, 201–223. [CrossRef]
7. Kuwata, Y.; Wolf, M.T.; Zarzhitsky, D.; Huntsberger, T.L. Safe Maritime Autonomous Navigation With COLREGS, Using Velocity Obstacles. *IEEE J. Ocean. Eng.* **2014**, *39*, 110–119. [CrossRef]
8. Mankabady, S. Volume 1: International Shipping Rules. In *The International Maritime Organization*; Croom Helm: London, UK, 1987; pp. 299–300.
9. Petres, C.; Romero-Ramirez, M.-A.; Plumet, F. Reactive path planning for autonomous sailboat. In Proceedings of the 15th International Conference on Advanced Robotics (ICAR), Tallin, Estonia, 20–23 June 2011; pp. 112–117.
10. Campbell, S.; Naeem, W. A Rule-based Heuristic Method for COLREGS-compliant Collision Avoidance for an Unmanned Surface Vehicle. *IFAC Proc.* **2012**, *45*, 386–391. [CrossRef]
11. Xue, Y.; Clelland, D.; Lee, B.; Han, D. Automatic simulation of ship navigation. *Ocean. Eng.* **2011**, *38*, 2290–2305. [CrossRef]
12. Xin, J.; Li, S.; Sheng, J.; Zhang, Y.; Cui, Y. Application of Improved Particle Swarm Optimization for Navigation of Unmanned Surface Vehicles. *Sensors* **2019**, *19*, 3096. [CrossRef] [PubMed]
13. Xin, J.; Zhong, J.; Yang, F.; Cui, Y.; Sheng, J. An Improved Genetic Algorithm for Path-Planning of Unmanned Surface Vehicle. *Sensors* **2019**, *19*, 2640. [CrossRef] [PubMed]
14. Ding, F.; Zhang, Z.; Fu, M.; Wang, Y.; Wang, C. Energy-efficient Path Planning and Control Approach of USV Based on Particle Swarm optimization. In Proceedings of the OCEANS Conference, Charleston, SC, USA; 2018; pp. 1–6.
15. Lazarowska, A. Ship's trajectory planning for collision avoidance at sea based on ant colony optimization. *J. Navig.* **2015**, *68*, 291–307. [CrossRef]
16. Zhou, X.; Wu, P.; Zhang, H.; Guo, W.; Liu, Y. Learn to Navigate: Cooperative Path Planning for Unmanned Surface Vehicles Using Deep Reinforcement Learning. *IEEE Access* **2019**, *7*, 165262–165278. [CrossRef]
17. Chen, C.; Chen, X.-Q.; Ma, F.; Zeng, X.-J.; Wang, J. A knowledge-free path planning approach for smart ships based on reinforcement learning. *Ocean. Eng.* **2019**, *189*, 106299. [CrossRef]
18. Zhang, X.; Wang, C.; Liu, Y.; Chen, X. Decision-Making for the Autonomous Navigation of Maritime Autonomous Surface Ships Based on Scene Division and Deep Reinforcement Learning. *Sensors* **2019**, *19*, 4055. [CrossRef]
19. Yu, R.; Shi, Z.; Huang, C.; Li, T.; Ma, Q. Deep reinforcement learning based optimal trajectory tracking control of autonomous underwater vehicle. In Proceedings of the 2017 36th Chinese Control Conference (CCC), Dalian, China, 26–28 July 2017.
20. Wang, C.-B.; Zhang, X.-Y.; Zhang, J.-W.; Ding, Z.-G.; An, L.-X. Navigation behavioural decision-making of MASS based on deep reinforcement learning and artificial potential field. *J. Phys. Conf. Ser.* **2019**, *1357*, 012026. [CrossRef]
21. Sutton, R.; Barto, A. Reinforcement Learning: An Introduction. *IEEE Trans. Neural Netw.* **1998**, *9*, 1054. [CrossRef]
22. Arulkumaran, K.; Deisenroth, M.P.; Brundage, M.; Bharath, A.A. Deep Reinforcement Learning: A Brief Survey. *IEEE Signal. Process. Mag.* **2017**, *34*, 26–38. [CrossRef]
23. Volodymyr, M.; Koray, K.; David, S. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533.
24. Mnih, V.; Kavukcuoglu, K.; Silver, D. Playing atari with deep reinforcement learning. *arXiv* **2013**, arXiv:1312.5602.
25. Kim, H.; Kim, D.; Shin, J.-U.; Kim, H.; Myung, H. Angular rate-constrained path planning algorithm for unmanned surface vehicles. *Ocean. Eng.* **2014**, *84*, 37–44. [CrossRef]
26. Paul, W.O.; Jeong-Hyon, H. Compression of trajectory data: A comprehensive evaluation and new approach. *Geoinformatica* **2014**, *18*, 435–460.
27. Zhang, D.; Zhang, X. Compression algorithm of GPS trajectory data based on space-time characteristics. *J. Transp. Inf. Saf.* **2013**, *3*, 6–9.
28. Douglas, D.H.; Peucker, T.K. Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or Its Caricature. *Can. J. Cardiol.* **1973**, *10*, 112–122. [CrossRef]