algorithms

MDPI

Article

Improved DQN for Dynamic Obstacle Avoidance and Ship Path Planning

Xiao Yang and Qilong Han *

School of Computer Science and Technology, Harbin Engineering University, Nantong Street, Harbin 150001, China; yangxiaoharbin@163.com
* Correspondence: hanqilong@hrbeu.edu.cn

Abstract: The avoidance of collisions among ships requires addressing various factors such as perception, decision-making, and control. These factors pose many challenges for autonomous collision avoidance. Traditional collision avoidance methods have encountered significant difficulties when used in autonomous collision avoidance. They are challenged to cope with the changing environment and harsh motion constraints. In the actual navigation of ships, it is necessary to carry out decision-making and control under the constraints of ship manipulation and risk. From the implementation process perspective, it is a typical sequential anthropomorphic decision-making problem. In order to solve the sequential decision problem, this paper improves DQN by setting a priority for sample collection and adopting non-uniform sampling, and it is applied to realize the intelligent collision avoidance of ships. It also verifies the performance of the algorithm in the simulation environment.

Keywords: ship collision avoidance; decision-making problem; reinforcement learning; DQN

check for updates

Citation: Yang, X.; Han, Q. Improved DQN for Dynamic Obstacle Avoidance and Ship Path Planning. *Algorithms* **2023**, *16*, 220. https://doi.org/10.3390/a16050220

Academic Editor: Frank Werner

Received: 16 March 2023
Revised: 15 April 2023
Accepted: 20 April 2023
Published: 25 April 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With the rise of artificial intelligence, developing the intelligence of transportation vehicles has become one of the essential objectives of artificial intelligence research and exploration. The intelligent navigation of ships is of great practical significance for improving the economic benefits of shipping enterprises. The development of artificial intelligence technology, especially reinforcement learning and its related methods, has complied with the requirements of intelligent ship collision avoidance and intelligent navigation [1,2]. Reinforcement learning emphasizes the agent's learning from the environment to behavior mapping, seeks the correct action decision by maximizing the income, and combines the profit and loss considerations in the operation process, which is a more appropriate method system in the intelligent collision avoidance of ships. Therefore, the essence of intelligent collision avoidance under the reinforcement learning system is to allow the agent to make the sequential decision to maximize the benefits under the balance of profit and loss.

The function of ship path planning is to complete the navigation index of the current and target position of the ship under a restricted navigation area and among obstacles, combined with the external environment and ship conditions. Obstacle avoidance is to avoid obstacles in an unknown dynamic environment during navigation. During driving, the operator will adjust the local path of the ship. The purpose of the traditional method is to provide a complete deterministic solution at the decision-making level, including manual rules and automatic execution, by analyzing the fusion information obtained by the ship at each decision-making time. At the core of the traditional method is the decision rules that people make according to the problem situation. Liu et al. [3] used the Fast Marching Method (FMM) to generate collision avoidance paths for the USV in a dynamic environment to ensure the safety of the USV. Wang et al. [4] combined A * and B spline methods to generate a feasible path without obstacles. Zereik et al. [5] described a navigation guidance and control system based on LOS and applied it to uncrewed ships considering only static

obstacles. Chen et al. [6] proposed a collision avoidance method for USV based on Rapidly-exploring Random Tree (RRT) and conducted a simulation verification on the MATLAB platform. Szymak et al. [7] proposed an improved neuro-fuzzy algorithm to construct the collision avoidance system of USV. When the constraints become more complex and the variable dimension increases, the traditional path planning algorithm has some defects, such as high time complexity, and quickly falls into local optimization.

Reinforcement learning [8,9] has developed rapidly in intelligent navigation because of its simple structure and strong adaptability. It was first proposed by Sutton, whose essence is to learn from the interaction. In recent years, there has been a boom in theoretical research and technical implementation of reinforcement learning, especially in the Deep Q-Network (DQN) algorithm [10,11] proposed by the Google team. In the shipping industry, reinforcement learning is widely used in collision avoidance. By integrating the navigation prior knowledge to design the ship's perception state and reward function, the action space can be designed based on the course control, and the DQN method can be applied to the ship collision avoidance. Deep learning has a strong ability for data analysis and perception. It can obtain helpful knowledge from multimodal information to strengthen learning without prior knowledge. Through the interaction between the agent and the environment, it can learn the best strategy to maximize the accumulated income. Ru et al. [12] proposed an intelligent path planning method based on deep reinforcement learning, which solved the problem of vehicle tracking error and over-dependence in traditional intelligent driving vehicle path planning. Saito et al. [13] proposed and evaluated a Tabu list-based DQN (TLS-DQN) for AAV mobility control; Yang et al. [14] studied the application of the DQN algorithm in deep reinforcement learning algorithms. Combining the Q-learning algorithm with the experience playback mechanism, the target Q value is generated to solve the multi-robot path planning problem; Yi et al. [15] improved DQN as a typical deep reinforcement learning method. This article decouples the choice of action and the target value calculation. At the same time, the influence of a static environment is also considered to improve the generalization ability.

The remaining structure of this paper is as follows: the second section describes the ship route planning problem; the third section introduces the DQN of reinforcement learning in detail; the fourth section displays the test results; the fifth section provides the conclusion of the paper.

2. Related Work

Research on intelligent collision avoidance of ships can be divided into actual ship research and virtual simulation. As a modern means of transportation, ships have high costs and high energy consumption. Therefore, conducting intelligent collision avoidance experiments with accurate ships in natural channels is costly and inefficient. It also faces possible safety problems caused by uncontrollable experimental conditions. If the navigation operation is slightly careless, it can cause serious safety concerns and substantial economic losses. Moreover, the actual ship is under the complex interference of external wind and waves, and the test scenarios and results have strong randomness and poor repeatability [16]. Therefore, this paper uses the virtual simulation software to simulate the ship navigation scene, build the shipping agent, restore the ship navigation state, and train the ship collision avoidance intelligence through the simulation experiment.

The ship's motion is based on navigation control, including positioning. Nevertheless, path planning is an essential aspect of navigation. Path planning aims to find an optimal path that meets the requirement of not intersecting with any obstacle from beginning to end in a specific environment. The path generated by ship path planning plays a navigation role in its movement, guiding the ship to start from the current point, avoid obstacles and reach the target point. The navigation control module includes global path planning and local path planning. Sang et al. [17] proposed a deterministic method called multiple sub-target artificial potential field (MTAPF). MTAPF belongs to the local path planning algorithm and refers to the globally optimal path generated through an improved heuristic A_* algorithm. This algorithm divides the optimal path into multiple sub-target points, forming a sequence

of sub-target points, using the improved A_* algorithm to obtain the optimal path, using the MTAPF algorithm for local path planning, and adopting a priority strategy to prevent collisions in local path planning. Lyu et al. [18] proposed a real-time and deterministic path-planning method for autonomous ships in complex and dynamic navigation environments. The improved artificial potential field method includes improved repulsive potential field function and corresponding virtual forces to solve the collision avoidance problem of dynamic targets and static obstacles, including emergencies. He et al. [19] proposed a dynamic path-planning algorithm based on the A-star algorithm and ship navigation rules. The improved algorithm's dynamic search mechanism, considering time factors, aims to avoid collisions when moving obstacles are known. The function of global path planning is to establish a viable path from the starting point to the target point according to the existing electronic map. Local path planning, also known as local obstacle avoidance, refers to perceiving unknown obstacles in ship operation and redefining the local path to bypass the obstacles and move towards the target point. When the global and local path planning algorithms work together, the ship can complete the autonomous motion from the starting point to the target point. As a core function, autonomous navigation is the key to realizing ship perception and motion. Path planning and positioning are essential to the ship navigation control system and constitute effective ways to achieve ship autonomy and intelligence. The simulation of ship collision avoidance path planning is divided into two aspects. The first is to simulate the environment model of areas with moving obstacles and accessible moving areas. The second step is to select the appropriate path search algorithm according to the established environment model to achieve fast and real-time path planning.

2.1. Ship Navigation Environment Modeling

For ship collision avoidance, simulation modeling of the navigation environment is essential. Compared with the road, the ship route contains more complex factors. Specifically, these factors will lead to fundamental differences in static collision avoidance, dynamic collision avoidance, and collaborative collision avoidance decision-making, so it is necessary to conduct modeling and analysis in advance [20]. The navigation environment includes the following aspects: (1) channel boundary and other boundary elements, including national boundaries and regional boundaries. (2) Water geographical elements, including intertidal zones, seabed topography, and navigation obstacles. (3) Various specific regional elements, including open water areas, restricted areas, port boundaries, and area boundaries. (4) Marine environmental elements include ocean currents, icebergs, undercurrents, and other elements in the ocean region. In order to solve the problem of ship collision avoidance, the channel is simplified according to the task requirements. During channel simulation modeling, the above channel elements are simplified as navigable and non-navigable zones.

When planning the path, the first consideration is not the specific location of obstacles in the environment but to divide the environment into several sub-regions with consistent topological characteristics and connectivity. The topology method builds the network and finds the topological path in the connection area. However, it can only reflect the interconnection between points in the natural environment and cannot establish an environment consistent with the geometric shape of the natural environment. If specific geometric details and paths need to be considered, the path planning method based on free space is essential. The commonly used methods include the visual graph method [21] and the grid method [22].

The visual graph method assumes that the agent and obstacle are particles and approximate polygons, respectively. It combines and connects particles, target points, and polygon vertices using visual line segments. This segment cannot pass through obstacles simulated by polygons. The path planning problem is transformed into a group of shortest line segments from the search starting point to the target point. Because the search path is a segment connecting the vertices of the obstacle, the proxy can obtain collision information with the obstacle. Many environmental barriers and feature information exist, extending the path search time. At the same time, the visual graph method has poor flexibility, lacks

adaptability to changing environments and has real-time constraints. In addition, it cannot guarantee that the path sought is the most optimal path globally.

The grid method uses a free grid and an obstacle grid, two grids with different attributes and sequence numbers to describe the environment. There are no obstacles in the free grid area, obstacles are divided into the obstacle grid area. The ship collision avoidance problem is transformed into a set of orderly feasible grids in the search workspace. This method takes the grid as the basic unit to record the agent's environment information. If the grid size is large, the description of the feature map will be rough. Thus, the quality of the planned path could be better, and the error could be more considerable. On the contrary, the more accurate the description of environmental obstacles, the better the quality of the planned path. It makes the storage space and search algorithm more complex. Therefore, the grid method should select an appropriate grid-scale and corresponding algorithm according to the specific application environment to optimize the ordered set of search grids. Grid graphs can describe many characteristics of the navigation environment and achieve the best consumption of time and space. Therefore, a grid map is widely used to describe the environment.

Due to the huge cost in the real environment, in order to verify the autonomous navigation function of the ship, we will simulate the whole ship navigation space. The whole navigation is carried out in feasible waters with obstacles. Obstacles do not represent fixed islands, but also shallow waters, wave areas and inaccessible locations [23]. In the whole navigation area, every position experienced by the ship is affected by wind and waves. Waves more than 3 m are dangerous areas [24]. The influence of wind and waves on the ship is also considered in the article. Marine meteorological data are from European Centre for Medium-Range Weather Forecasts(ECMWF) [25]. Figure 1 shows the path planning environment modeling of this paper. The width and height set by the test simulation are 12×12, and the cell is set to 40 pixels. The position of the robot represents the starting point, the green square represents the obstacle, and the flag represents the target point. The agent needs to find a path from the starting point to the target point in the shortest cost, which should avoid obstacles perfectly. In this paper, dynamic means that in addition to the starting point, obstacles and target point are randomly generated in the process of agent travel. Dynamic obstacle modeling is shown in Figure 2.

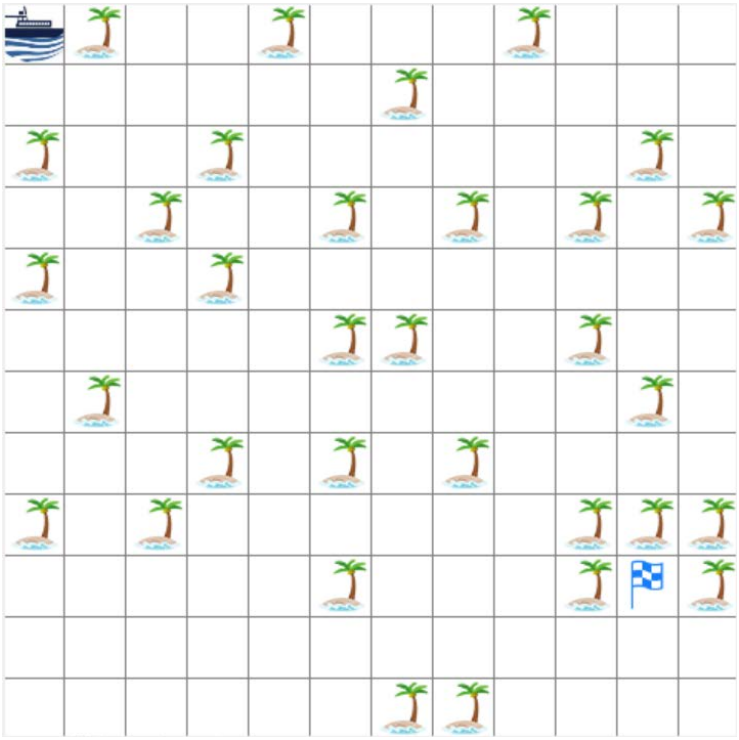

Figure 1. Environment modeling of path planning.

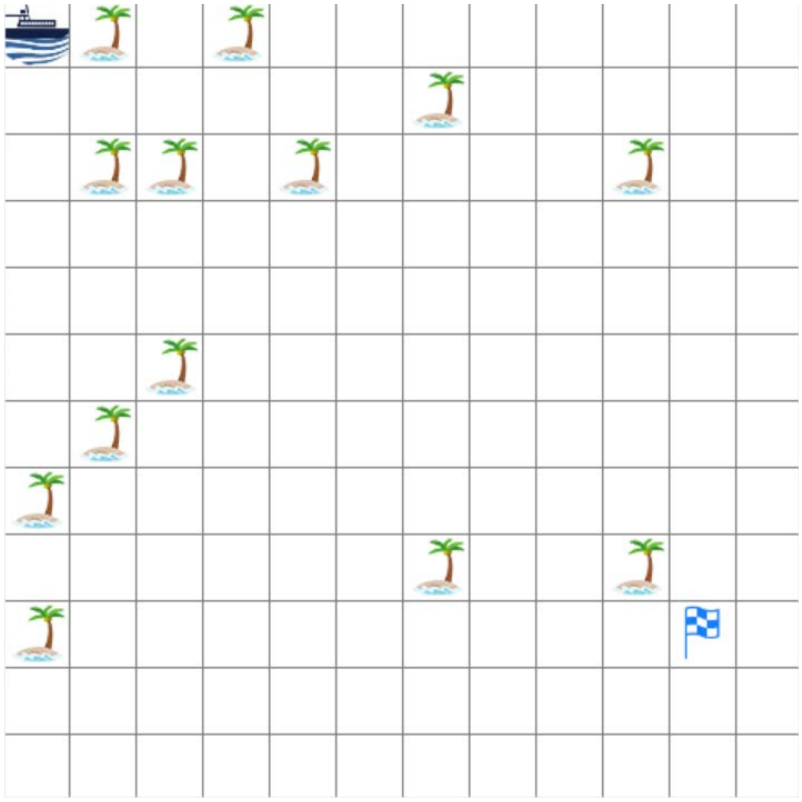

Figure 2. Dynamic environment modeling of path planning.

2.2. *Algorithm of Path Planning*

The goal of ship collision avoidance is to complete the navigation planning of the current and target position of the ship in combination with the external environment and the ship situation in the case of restricted navigation and obstacles. The obstacle avoidance problem of ships is that the ship perceives the unknown environment during the navigation process to avoid adverse factors and achieve the goal of safe navigation. During navigation, the operator will adjust the local path of the ship. Traditional ship collision avoidance algorithms, such as A^* [19], artificial potential field [26], RRT [27], and other models have limitations such as being prone to fall into local optimal solutions, low search efficiency, or slow calculation speed.

Although the research on intelligent collision avoidance of ships has made significant progress in recent years, the above algorithms have their advantages. However, no intelligent collision avoidance method that can replace human pilots has been found with the extensive application of deep reinforcement learning in robots, games, image processing, medicine, and other fields [28–31]. In particular, deep reinforcement learning has made a breakthrough in intelligent driving and uncrewed vehicles. The successful application of reinforcement learning in the field of intelligent vehicles has quickly extended its experience to collision avoidance algorithms in the field of intelligent ships. The reinforcement learning method has developed rapidly in intelligent navigation because of its simple structure and strong adaptability. It was first proposed by Sutton, whose essence is to learn from the interaction. In recent years, especially the Deep Q-Network (DQN) algorithm proposed by the Google team has caused a boom in theoretical research and technical implementation of reinforcement learning. In the shipping industry, reinforcement learning is widely used in collision avoidance to integrate the navigation prior knowledge to design the ship's perception state and reward function, design the action space based on the course control, and apply the DQN method to the ship collision avoidance. Wang et al. [32] developed a new intelligent collision avoidance algorithm based on approximate representation reinforcement learning (AR-RL) and applied it to ship collision avoidance in continuous space.

Zhai et al. [33] proposed a multi-ship automatic collision avoidance method based on a double deep Q network (DDQN) with prioritized experience replay.

3. The Proposed Deep Q Network(DQN)

The development of AI technology, especially reinforcement learning and its related methods, has adapted well to the needs of intelligent collision avoidance of ships. Reinforcement learning learns strategies through the interaction between agents and the environment to maximize returns or achieve specific goals. It can obtain helpful knowledge from multimodal information without prior knowledge. Through the interaction between the agent and the environment, it can learn the best strategy to maximize the accumulated income. Therefore, the essence of intelligent collision avoidance under the reinforcement learning method is to let the agent make the sequential decision to maximize the benefits under the balance of profit and loss, which is closer to human driving intelligence. DQN (Deep Q-network) refers to a Q-learning algorithm [34,35] based on deep learning. It mainly combines the value function approximation and neural network technology and uses the target network and experience playback to train the network. DQN is a value-based algorithm. In the value-based algorithm, what they learn is not the strategy but the evaluation of the action. In DQN, the state is entered. The neural network extracts features, and the Q value of the action is back-propagated to the evaluation network through the loss function.

Among many methods, the reinforcement learning algorithm is more suitable [36–38]. It can allow the ship agent to interact with the environment in real-time and improve the collision avoidance ability by constantly trying to accumulate experience, which is very similar to the experience accumulation process of people. DQN can directly output the action decisions agents should take in different states, thus highly simulating human decisions and actions in real situations. In addition, DQN can also fully consider the constraints of navigation rules and conventions when designing the value function. It is a collision avoidance method that can be used in actual navigation. The agents in DQN implement new actions according to specific strategies according to the rewards and environmental feedback of the new status. This learning method lets agents know which actions they should take in a specific state to maximize their rewards.

3.1. The Original DQN

Reinforcement learning is a typical Markov decision process (MDP) [39]. The environment information obtained in the ship traveling process is insufficient, so the Markov decision process modeled in this paper is a partially observable Markov decision process (POMDP). POMDP can use $(S, A, P_{sa}, R, \omega, O, \gamma)$ description, where $S = s_1, s_2, \ldots, s_n$ is a set of state space, and s_i represents the state of step i. $A = a_1, a_2, \ldots, a_n$ is the action space set, and a_i indicates the action of the step i. P_{sa} is the state transition probability matrix. R is the reward function. The probability of observation is ω. O is the observation space. γ is the discount factor. The value function iteration method of DQN and Q-learning is very similar. Q-learning directly uses the subsequent state data of the iteration for learning. Q-learning is based on the optimal action-value function Q_*. Both algorithms adopt the optimal Behrman equation to make the agent learn some data, as shown Equation (1) .

$$Q_*(s_t, a_t) = \mathbb{E}_{s_{t+1} \sim p(\cdot|s_t, a_t)}[R_t + \gamma \underset{A \in \mathcal{A}}{max} Q_*(S_{t+1}, A) | S_t = s_t, A_t = a_t] \tag{1}$$

Equation (1) is approximated. $Q_*(s_t, a_t)$ on the left of the Equation (1) can be approximated as $\tilde{Q}(s_t, a_t)$. $\tilde{Q}(s_t, a_t)$ is made at step t. On the right of the Equation (1), the mathematical expectation is about the state S_{t+1} at the next step. If the current state s_t is known, the agent executes the action a_t according to strategy, and the environment will return the reward r_t and obtain the new state s_{t+1}. Based on r_t and s_{t+1}, the expectation is calculated by Monte Carlo approximation, as shown in Equation (2).

$$r_t + \gamma \underset{a \in \mathcal{A}}{max} Q_*(s_{t+1}, a) \tag{2}$$

Further, Q_* in Equation (2) is approximated by $\tilde{Q}$. Equation (3) is used to estimate $Q_*(s_t, a_t)$ at step $t+1$, it is called Temporary Difference (TD). $\tilde{Q}(s_t, a_t)$ and $\hat{y}_t$ is the estimation with regard to optimal action value $Q_*(s_t, a_t)$. Because $\hat{y}_t$ includes reward r_t partially based on real observations, it is believed that $\hat{y}_t$ is the more reliable. So, it is encouraged $\tilde{Q}(s_t, a_t))$ to be closed to $\hat{y}_t$. Therefore, Equation (4) is used to update $\tilde{Q}$.

$$\hat{y}_t \triangleq r_t + \gamma \underset{a \in \mathcal{A}}{max} \tilde{Q}(s_{t+1}, a) \tag{3}$$

$$\tilde{Q}(s_t, a_t) \leftarrow (1-\alpha)\tilde{Q}(s_t, a_t) + \alpha \hat{y}_t \tag{4}$$

However, there are some differences between DQN and Q-learning. The DQN consists of two networks–network Q and target network $\tilde{Q}$. At initialization, the structure of the target network $\tilde{Q}$ and Q are the same. In each episode, the agent will obtain a state s_t and take an action a_t when interacting with the environment. Network Q based on ε-greedy is used to evaluate the Q-value function. Then, the agent obtains the reward r_t and enters the state s_{t+1}. Therefore, some data (s_t, a_t, r_t, s_{t+1}) will be collected and input to the experience replay. If the experience replay is full, some old data will be discarded. Next, the sampled data in the playback buffer is calculated by the target network $\tilde{Q}$, as shown in Equation (3).

The detailed operation of experience playback is as follows: first, randomly select n batches of data from the experience playback pool and record it as (s_t, a_t, r_t, s_{t+1}), $\tilde{Q}_{now}$, $\tilde{Q}_{new}$; secondly, the $\tilde{Q}_{now}$ at position (s_j, a_j) is expressed as Equation (5); thirdly, the maximum value of $\tilde{Q}_{now}$ at s_{j+1} is calculated as Equation (6); fourthly, Equation (7) is used to calculate TD target; fifth, Equation (8) is adopted to update the element at (s_j, a_j).

$$\hat{q}_j = \tilde{Q}_{now}(s_j, a_j) \tag{5}$$

$$\hat{q_{j+1}} = \underset{a}{max} \tilde{Q}_{now}(s_{j+1}, a) \tag{6}$$

$$\hat{y}_j = r_j + \gamma \hat{q_{j+1}}, \delta_j = \hat{q}_j - \hat{y}_j \tag{7}$$

$$\tilde{Q}_{new}(s_j, a_j) \leftarrow (1-\alpha)\tilde{Q}_{now}(s_j, a_j) + \alpha \delta_j \tag{8}$$

3.2. The Improved DQN

DQN uses prioritized experience replay (PER). When sampling data train the Q network, the data will be uniformly sampled from the playback buffer. This is not necessarily the best method because some data may be more critical. Assuming that some data have been sampled before, it was found that the timing difference error of these data is enormous. The timing difference error is the difference between the output of the network and the target, which means that these data are relatively challenging to train when training the network. Since it is not easy to train, we should give them a high probability of being sampled, that is, give them a priority. In this way, we will consider the data that are not good for training. When using PER, we will change the sampling process and the method of updating parameters because of changing the sampling process. So PER not only changes the distribution of sampling data but also changes the training process.

Ordinary experience playback obtains a sample every time and uses it to update DQN parameters. Prioritized experience replay assigns a weight to each sample and performs non-uniform random sampling according to the weight. If DQN has an inaccurate judgment on the value of (s_j, a_j), that is, Q is far from Q_* , the sample should have a higher weight. The data generated during navigation have different levels of importance. In path planning, agents will generate many such samples. However, the ship will avoid obstacles during the navigation process, and these small samples will significantly impact the path. Therefore, the data on avoiding obstacles should have a higher weight and receive more attention.

Samples should have different degrees of attention. We can determine which samples are more important according to the absolute value of TD error. If the absolute value of TD error δ_j is large, which indicates that the current evaluation of the real value of (s_j, a_j) is inaccurate. Then the higher weight should be set for (s_j, a_j, r_j, s_{j+1}). In this paper, Equation (9) is used to calculate the sampling probability. If uniform sampling is used, all samples have the same learning rate of $alpha$. If non-uniform sampling is adopted, the learning rate $alpha$ must be adjusted according to the sampling probability. Set the learning rate as Equation (10).

$$p_j \propto |\delta_j| + \epsilon \tag{9}$$

$$p_j = \frac{p_j^a}{\sum_k p_k^a} \tag{10}$$

where, ϵ is a small number to prevent the sampling probability from approaching 0. β is a super parameter that needs to be adjusted. We take multi-step updating to make the Q network as close as possible to the target value. The multi-step method combines the Monte Carlo method and the time series difference method, so it has not only the advantages and disadvantages of the Monte Carlo method but also the advantages and disadvantages of the time series difference method. Previously, only a particular step was sampled, so the data obtained is accurate, and the Q value is estimated. Now there are more steps to sample and only N steps to estimate the value, so the impact of the estimated part will be relatively small. Of course, the disadvantages of the multi-step method are the same as those of the Monte Carlo method. Because there are many items, the total variance of N items will be more significant. However, we can adjust the value of N to balance the variance and the inaccurate Q value. The whole algorithm flow is shown in Algorithm 1.

Algorithm 1 DQN based on prioritized experience replay and multi-step updating

Initialize Q and $\tilde{Q}$, let Q = $\tilde{Q}$, b, β, γ ;
for e = 1 to episodes
  repeat;
  Initialization status s_t, perform action a_t based on Q(ε -greedy strategy);
  obtain r_t and the new state s_{t+1};
  Store N samples (s_i, a_i, r_i, s_{i+1}) to experience playback pool;
  Prioritized experience replay and multi-step update Q;
  Non-uniform sampling from experience playback pool with Equation (9) ;
  Calculate the target value with Equation (7);
  Update the parameters of $Q_{(s_i,a_i)}$ so that Q is as close to y as possible;
  Reset Q = $\tilde{Q}$ every m update;
end for

4. Experiment

4.1. Environment and Parameter Settings

The setting of the test running environment of the paper is shown in Table 1. The experiments in this paper are based on the following equipment.

Table 1. Experimental environment.

Configuration	Parameters
Operating system	CentOS Linux release 7.6.1810 (Core)
Memory	754 G
CPU	Intel(R) Xeon(R) Platinum 8260 CPU
Basic frequency	2.40 GHZ
Programing language	Python 3.8.3
Graphics card	NVIDIA Corporation TU102GL (rev a1)

Note that $\gamma = 0$ means that the agent only cares about the recent rewards, while $\gamma = 1$ makes it committed to higher rewards in the long term. If the discount factor reaches or exceeds 1, the action value Q may diverge. In this case, the learning efficiency can be improved by gradually increasing the discount factor from a lower value to the final value. As the number of iterations increases, the value of γ increases from 0.1 to 0.9. the total number of samples in the experience replay b = 5000. β = 0.3. In the experimental simulation, the actions taken by the agent are only up, down, left and right.

4.2. Comparison of Test Results

According to the above experimental simulation environment and parameter settings, the ship path drawn by improved DQN, original DQN, Q-learning algorithm in the case of obstacles is shown in Figure 3–5. It can be seen from the figure that the ship draws the shortest navigation path with the improved DQN. Although all three algorithms finally find the end point, their costs are different. Figures 4 and 5 show the capabilities of the original DQN and Q-learning in ship path planning, and the two algorithms produce repeated exploration in some sections. The original DQN performs better than the Q-learning algorithm. The costs of the three algorithms at given starting point and end point are shown in Table 2.

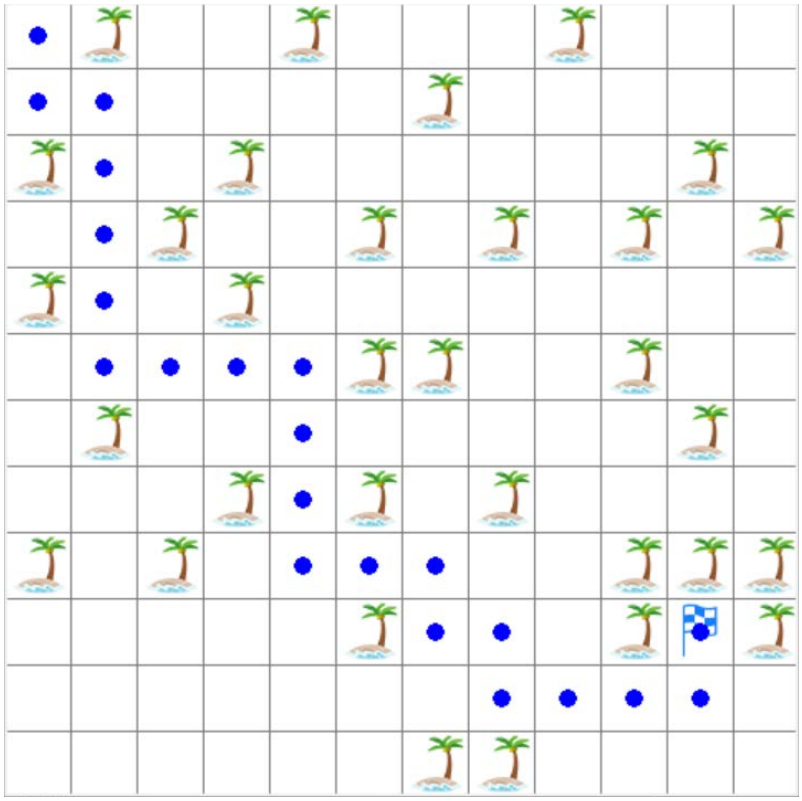

Figure 3. The improved DQN for path planning.

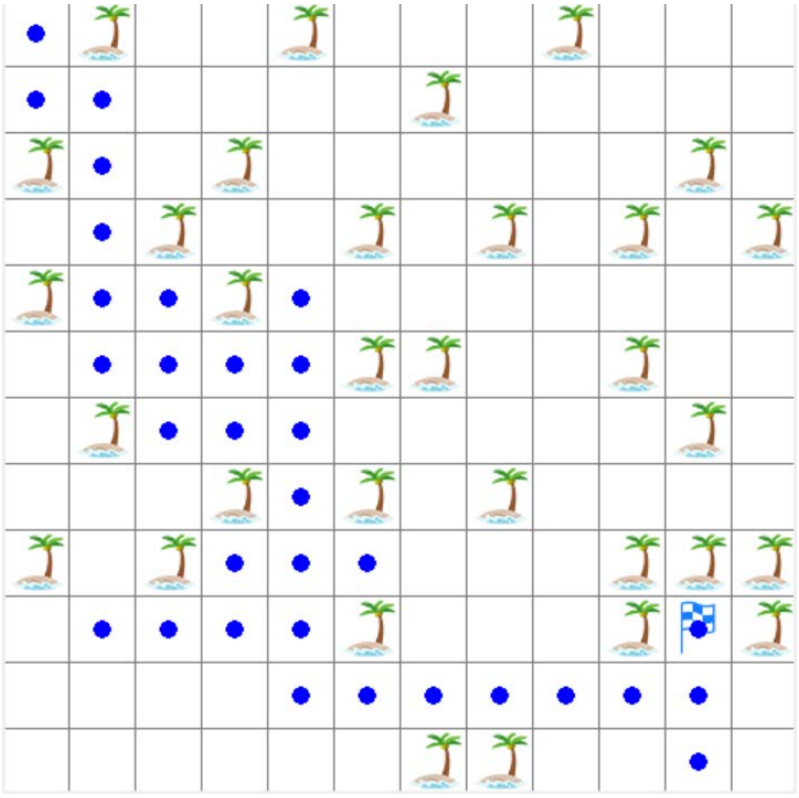

Figure 4. The original DQN for path planning.

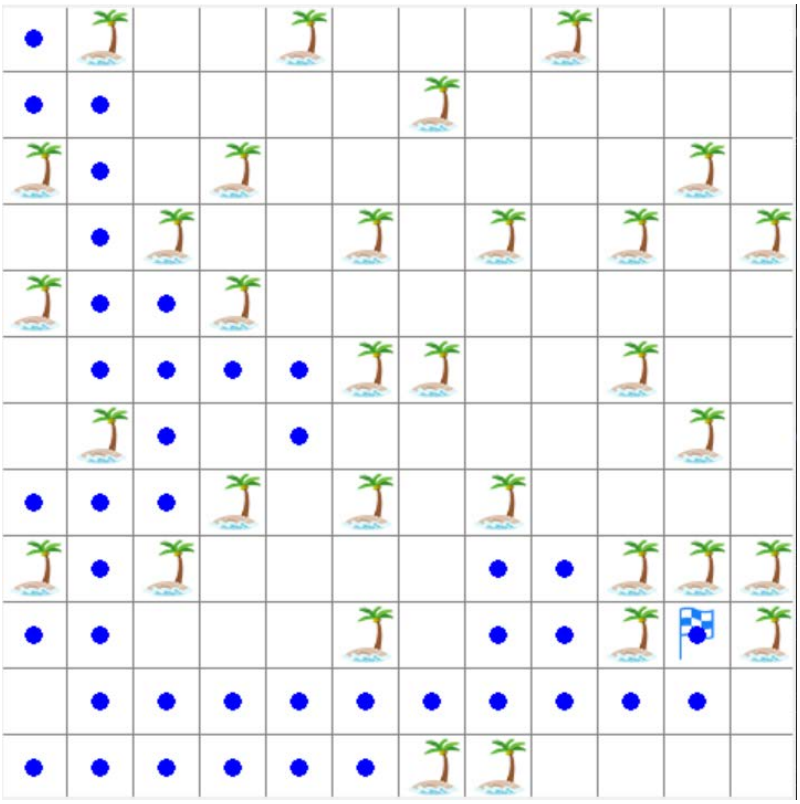

Figure 5. The Q-learning for path planning.

Table 2. Experimental environment

Algorithm	The Length of Path
The improved DQN	21
The original DQN	32
The Q-learning	124

In the simulation, when the ship arrives at the destination, the reward value is 1, and when the ship moves towards the obstacle, the reward value is −1. Moreover, the reward value is 0. The maximum episode setting in the paper is 2000. With the execution of the episode, the steps and cost of each episode of the ship are shown in Figures 6–11. In Figure 6, the ship based on the improved DQN algorithm is constantly explored, and the steps will change dramatically. When the algorithm is executed to the 1750th episode, it reaches stability. The value of steps is 21. The steps of the original DQN algorithm are compared, as shown in Figure 7. In the whole episode process, the original DQN has been in dynamic exploration and can not move towards the established goal due to uniform sampling. Figure 8 shows the steps performance of Q-learning algorithm. Q-learning has explored the longest step, but the algorithm is not stable. Figures 9–11 shows the cost of the whole episode. In this paper, cost is the reward obtained by the ship when it travels to the target point. It can be seen from the figure that the reward of the improved DQN algorithm has always been at the maximum in the 1750th episode. The performance of the other two algorithms is weaker than that of the improved DQN.

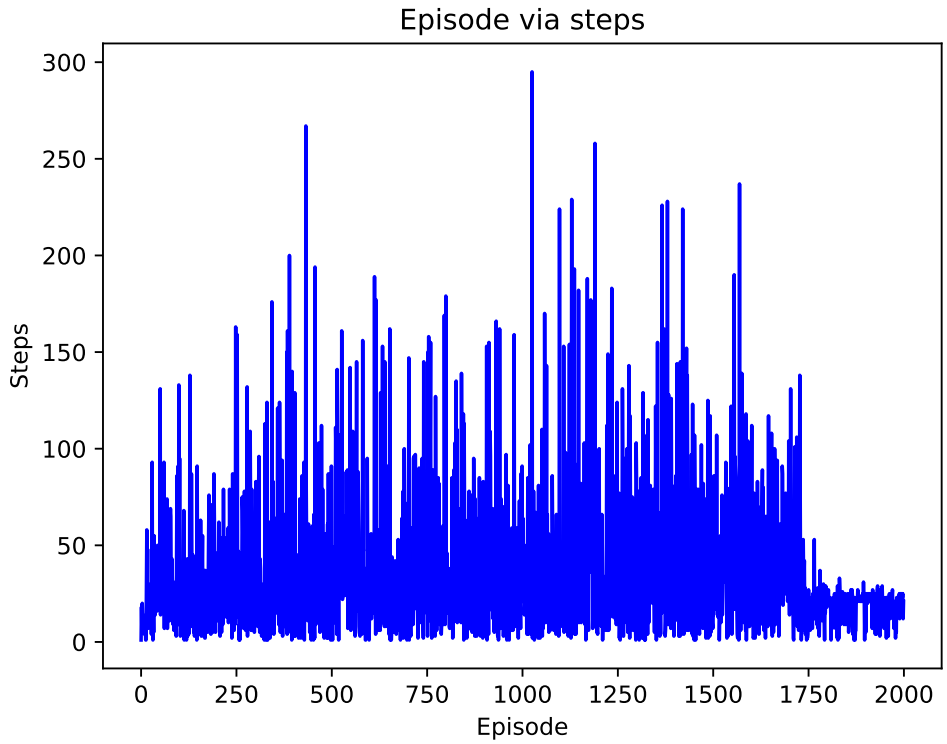

Figure 6. Episode via steps of the improved DQN.

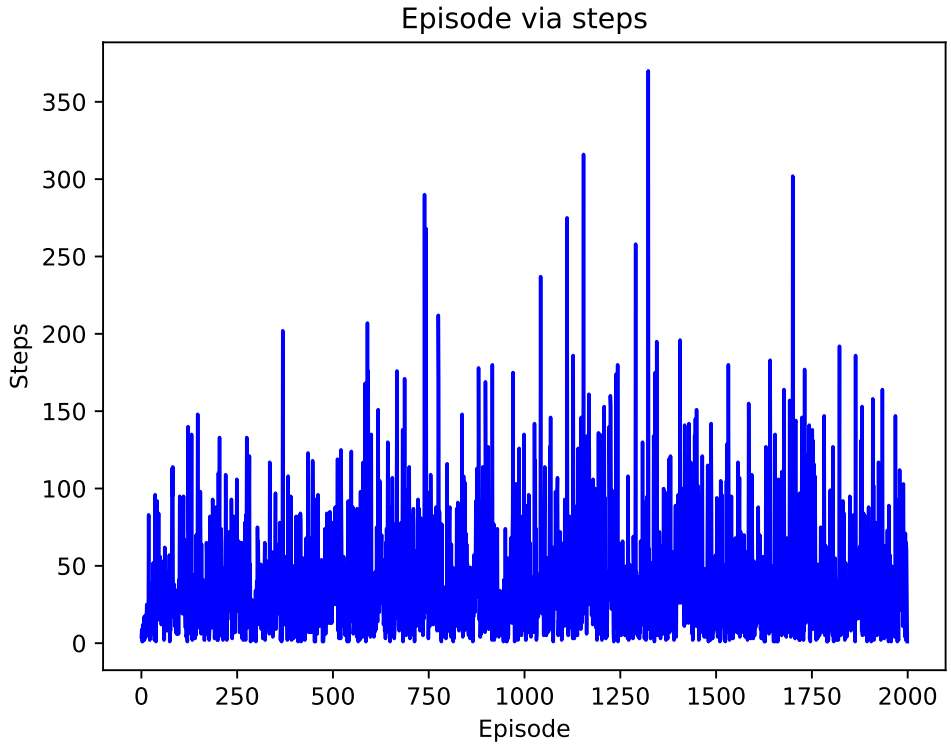

Figure 7. Episode via steps of the original DQN.

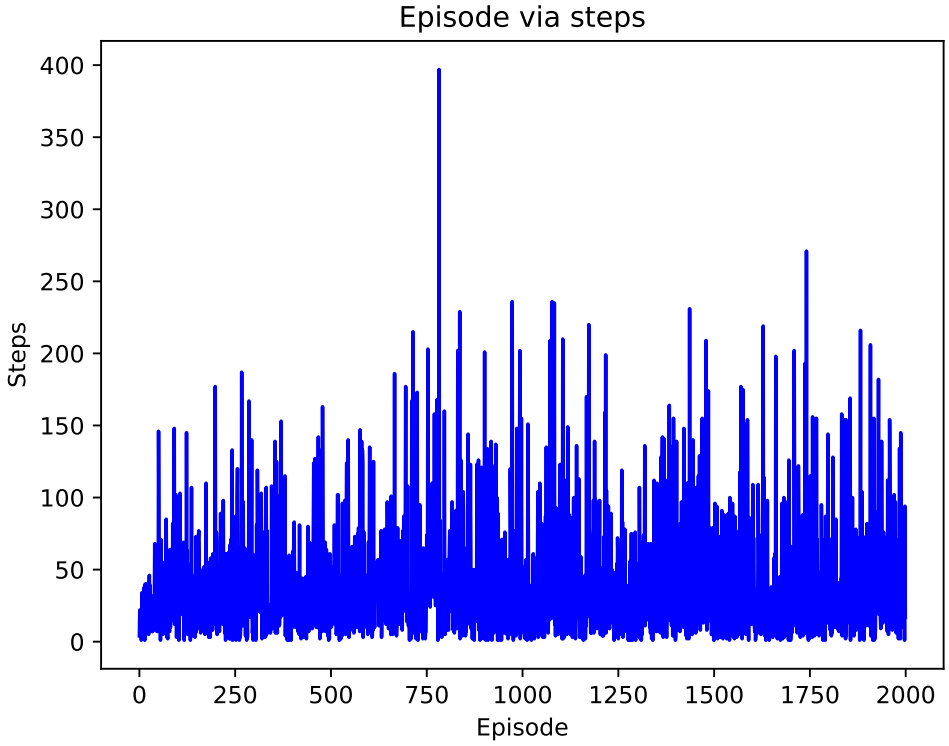

Figure 8. Episode via steps of the Q-learning.

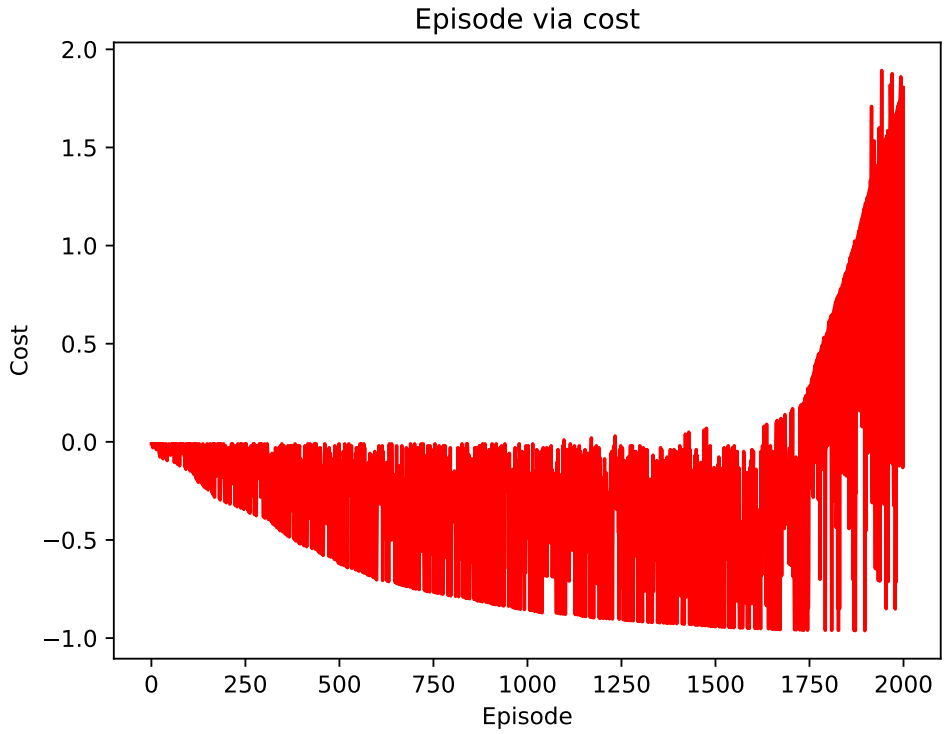

Figure 9. Episode via cost of the improved DQN.

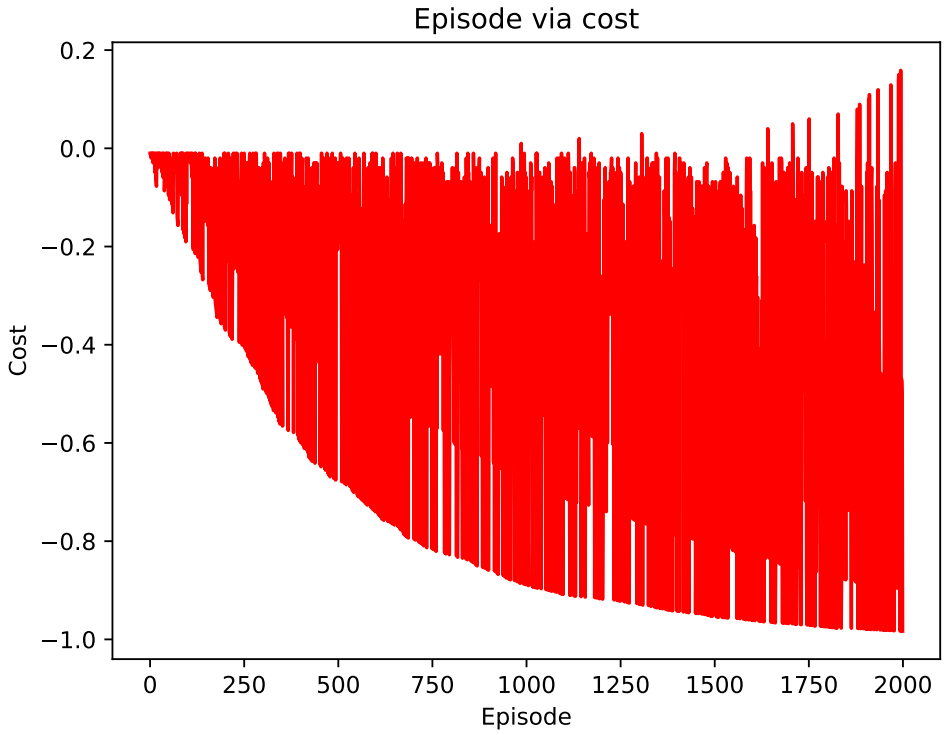

Figure 10. Episode via cost of the original DQN.

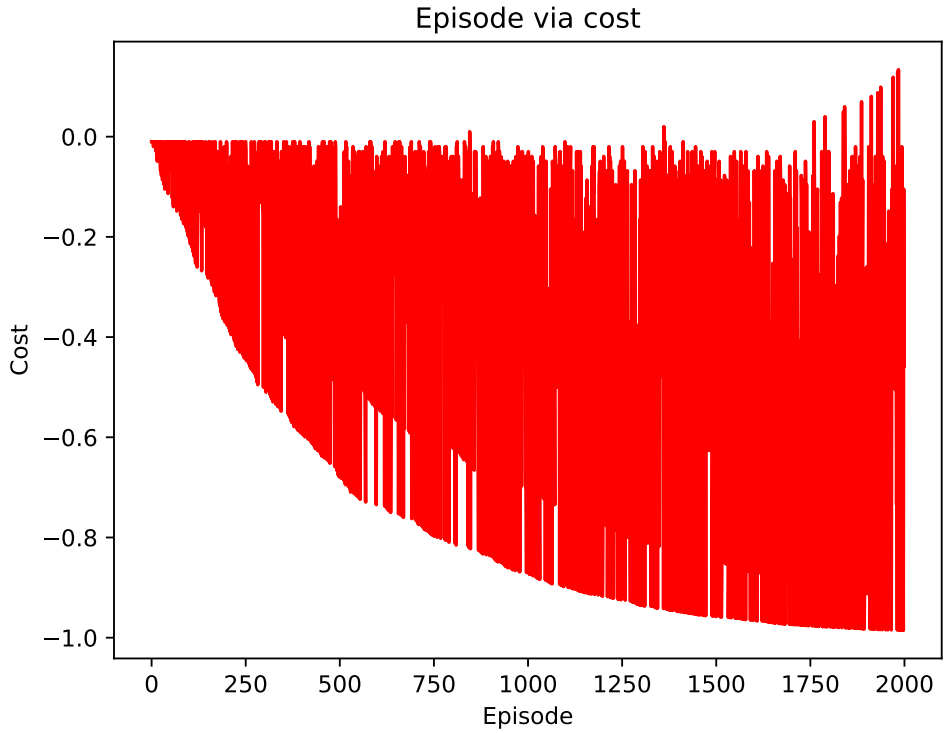

Figure 11. Episode via cost of the Q-learning.

In order to further verify the performance of the improved DQN algorithm, this paper dynamically generates obstacles in the simulation environment except that the meteorological environment remains unchanged. The simulation results are shown in Figures 12–17. The improved DQN algorithm also performs well in the ship path planning problem with randomly generated obstacles. Under the complex environment, the improved DQN can also find a safe and reliable path. As shown in Figures 13 and 16, after little exploration, the improved DQN algorithm can find a feasible path and reach a steady state. Since the goal of the algorithm is to obtain the maximum reward, the improved DQN algorithm aims to pursue the maximum reward. The cost of each episode is shown in Figures 13 and 16.

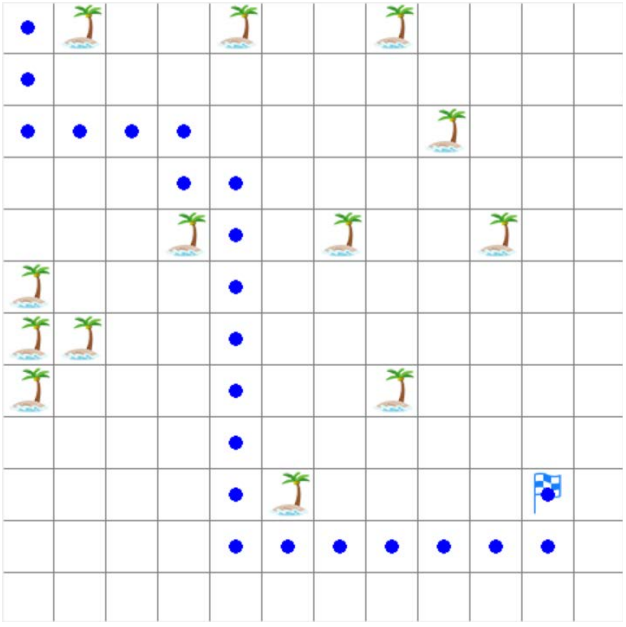

Figure 12. The improved DQN for the path planning of dynamic obstacle scenario 1.

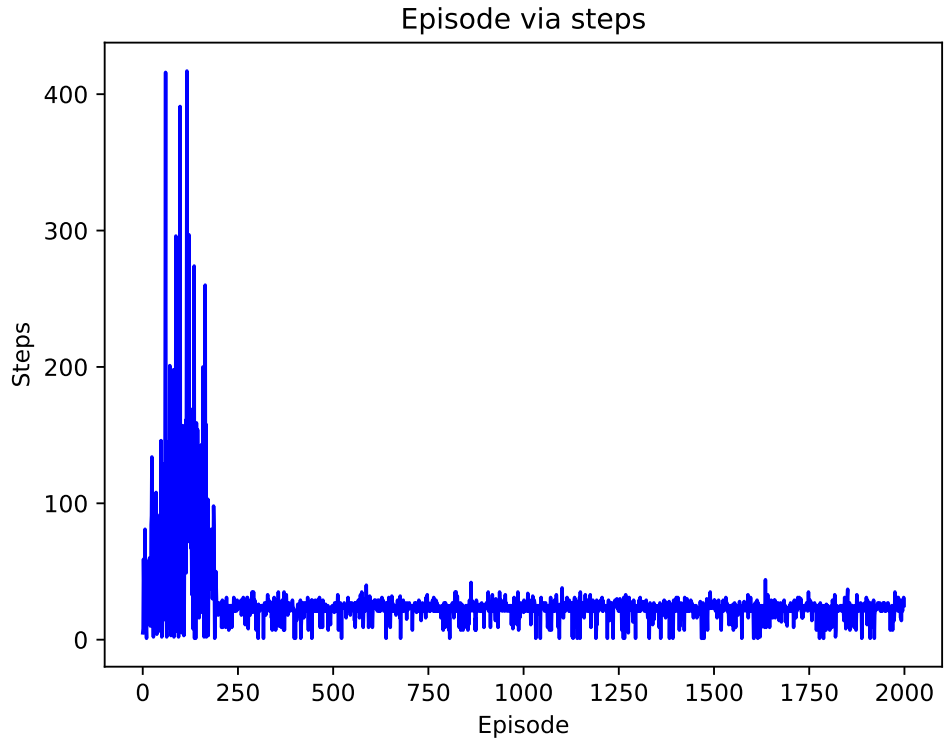

Figure 13. Episode via steps of the improved DQN for scenario 1.

Figure 14. Episode via cost of the improved DQN for scenario 1.

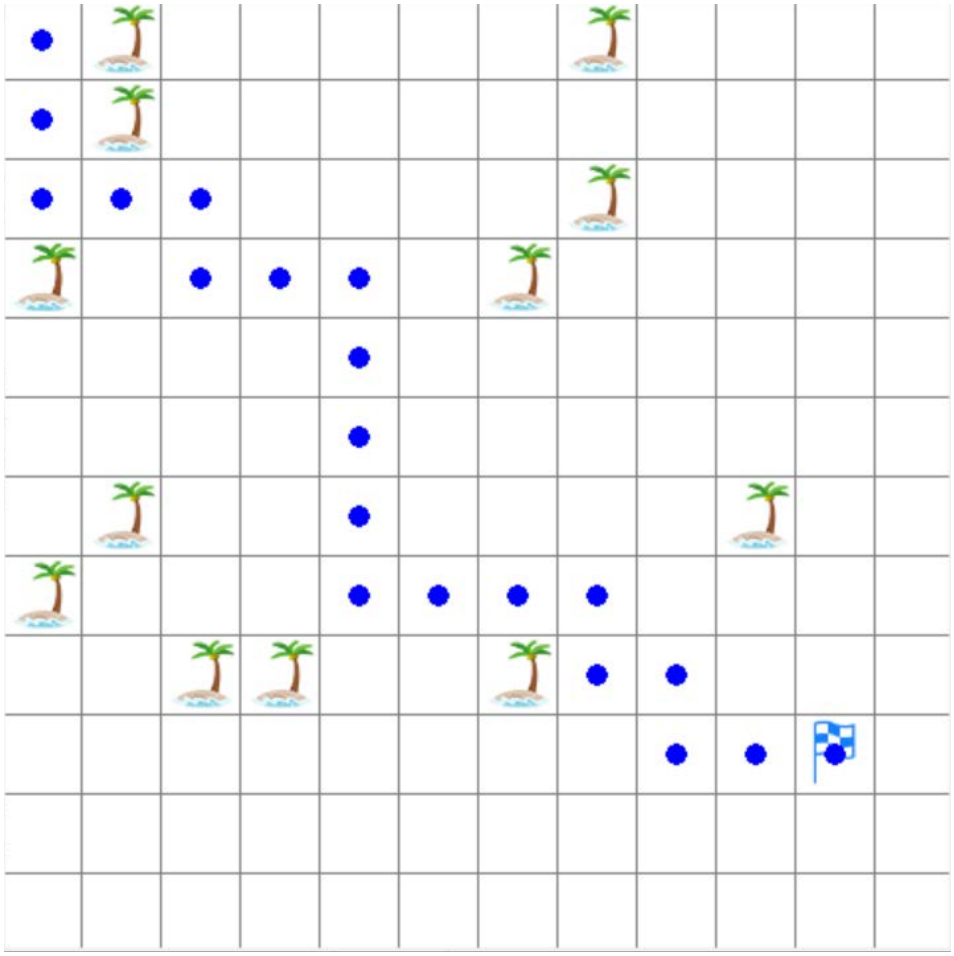

Figure 15. The improved DQN for the path planning of dynamic obstacle scenario 2.

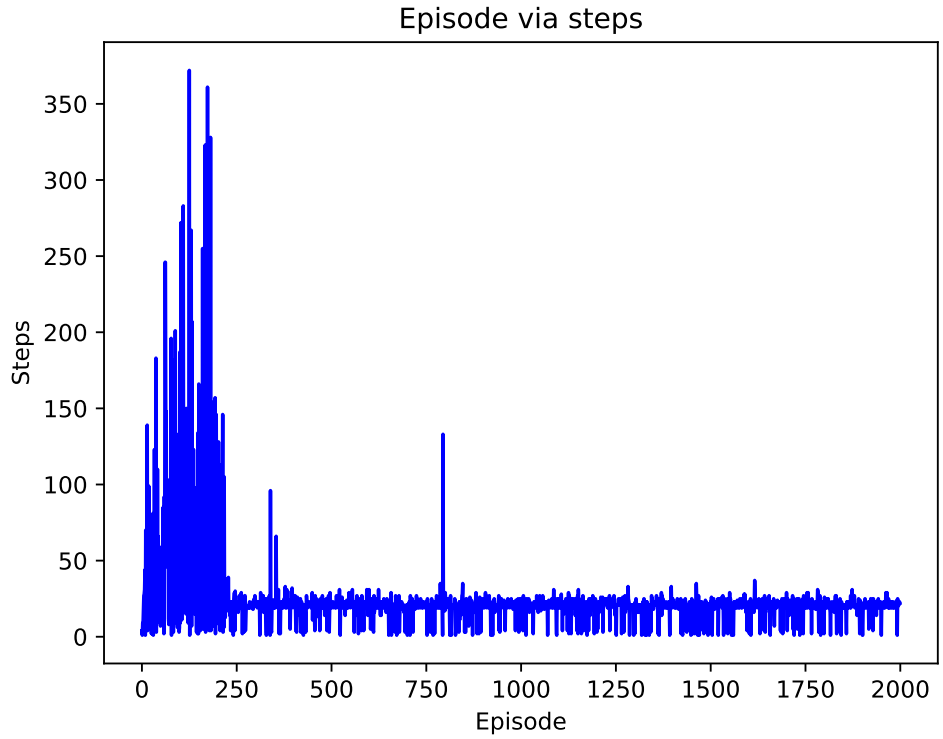

Figure 16. Episode via steps of the improved DQN for scenario 2.

Figure 17. Episode via cost of the improved DQN for scenario 2.

5. Discussion

The complexity of the environment puts forward higher requirements for the rapidity and flexibility of path planning. The rapid development of artificial intelligence provides an effective way for path planning of complex problems. This paper improves the experience replay of DQN algorithm, and it is applied to path planning in complex marine environments. By sorting and sampling the samples (s_j, a_j, r_j, s_{j+1}) in the experience playback pool, the higher the value of the samples, the greater the probability of the sampling motivating the agent to learn correctly. The multi-step update reduces the inaccurate estimation of Q value and accelerates training. The improved DQN enhances the algorithm's performance by training network parameters based on non-uniform sampling by setting sample priority. In the simulation environment, the unit length of the shortest collision avoidance path planned by the improved DQN is 21, which is superior to other comparative algorithms. In the simulation environment of dynamically generating random obstacles, the shortest collision avoidance path planned by the improved DQN outperforms other comparative algorithms regarding exploration steps and rewards. The improved DQN algorithm is applied to ship path simulation and can plan reliable paths in dynamic and static obstacle environments. Experiments show that this improved DQN reinforcement learning algorithm can be used to solve the path planning problem in complex environments. The ship simulation environment designed in this article is relatively simple, and exploring DQN planning collision avoidance paths in more complex environments is more meaningful. Exploring other technologies to improve DQN performance in solving collision avoidance path problems is worthy of future research.

Author Contributions: Conceptualization, X.Y. and Q.H.; methodology, X.Y.; software, X.Y.; validation, X.Y.; formal analysis, Q.H.; investigation, X.Y.; resources, Q.H.; data curation, X.Y.; writing—original draft preparation, X.Y.; writing—review and editing, X.Y.; visualization, X.Y.; supervision, Q.H.; project administration, Q.H.; funding acquisition, Q.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the National Key R&D Program of China under Grant No. 2020YFB1710200.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data available on request due to restrictions, e.g., privacy or ethical.

Acknowledgments: Thanks for the support of the laboratory for the simulation experiment.

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

References

1. Zhang, X.; Wang, C.; Jiang, L.; An, L.; Yang, R. Collision-avoidance navigation systems for Maritime Autonomous Surface Ships: A state of the art survey. *Ocean Eng.* **2021**, *235*, 109380. [CrossRef]
2. Jiang, L.; An, L.; Zhang, X.; Wang, C.; Wang, X. A human-like collision avoidance method for autonomous ship with attention-based deep reinforcement learning. *Ocean Eng.* **2022**, *264*, 112378. [CrossRef]
3. Liu, Y.; Bucknall, R. Efficient multi-task allocation and path planning for unmanned surface vehicle in support of ocean operations. *Neurocomputing* **2018**, *275*, 1550–1566. [CrossRef]
4. Wang, Z.; Xiang, X.; Yang, J.; Yang, S. Composite Astar and B-spline algorithm for path planning of autonomous underwater vehicle. In Proceedings of the 2017 IEEE 7th International Conference on Underwater System Technology: Theory and Applications (USYS), Kuala Lumpur, Malaysia, 18–20 December 2017; pp. 1–6.
5. Zereik, E.; Sorbara, A.; Bibuli, M.; Bruzzone, G.; Caccia, M. Priority task approach for usvs' path following missions with obstacle avoidance and speed regulation. *IFAC-PapersOnLine* **2015**, *48*, 25–30. [CrossRef]
6. Chen, Z.; Yu, J.; Zhao, Z.; Wang, X.; Chen, Y. A Path-Planning Method Considering Environmental Disturbance Based on VPF-RRT. *Drones* **2023**, *7*, 145. [CrossRef]
7. Szymak, P.; Praczyk, T. Using neural-evolutionary-fuzzy algorithm for anti-collision system of Unmanned Surface Vehicle. In Proceedings of the 2012 17th International Conference on Methods & Models in Automation & Robotics (MMAR), Miedzyzdroje, Poland, 27–30 August 2012; pp. 286–290.
8. Chen, P.; Pei, J.; Lu, W.; Li, M. A deep reinforcement learning based method for real-time path planning and dynamic obstacle avoidance. *Neurocomputing* **2022**, *497*, 64–75. [CrossRef]
9. Li, J.; Chen, Y.; Zhao, X.; Huang, J. An improved DQN path planning algorithm. *J. Supercomput.* **2022**, *78*, 616–639. [CrossRef]
10. Xu, X.; Cai, P.; Ahmed, Z.; Yellapu, V.S.; Zhang, W. Path planning and dynamic collision avoidance algorithm under COLREGs via deep reinforcement learning. *Neurocomputing* **2022**, *468*, 181–197. [CrossRef]
11. Xiaofei, Y.; Yilun, S.; Wei, L.; Hui, Y.; Weibo, Z.; Zhengrong, X. Global path planning algorithm based on double DQN for multi-tasks amphibious unmanned surface vehicle. *Ocean Eng.* **2022**, *266*, 112809. [CrossRef]
12. Ru, J.; Yu, H.; Liu, H.; Liu, J.; Zhang, X.; Xu, H. A Bounded Near-Bottom Cruise Trajectory Planning Algorithm for Underwater Vehicles. *J. Mar. Sci. Eng.* **2022**, *11*, 7. [CrossRef]
13. Saito, N.; Oda, T.; Hirata, A.; Nagai, Y.; Hirota, M.; Katayama, K.; Barolli, L. A Tabu list strategy based DQN for AAV mobility in indoor single-path environment: Implementation and performance evaluation. *Internet Things* **2021**, *14*, 100394. [CrossRef]
14. Yang, Y.; Juntao, L.; Lingling, P. Multi-robot path planning based on a deep reinforcement learning DQN algorithm. *CAAI Trans. Intell. Technol.* **2020**, *5*, 177–183. [CrossRef]
15. Yi, C.; Qi, M. Research on virtual path planning based on improved DQN. In Proceedings of the 2020 IEEE International Conference on Real-time Computing and Robotics (RCAR), Asahikawa, Japan, 28–29 September 2020; pp. 387–392.
16. Bakdi, A.; Vanem, E. Fullest COLREGs evaluation using fuzzy logic for collaborative decision-making analysis of autonomous ships in complex situations. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 18433–18445. [CrossRef]
17. Sang, H.; You, Y.; Sun, X.; Zhou, Y.; Liu, F. The hybrid path planning algorithm based on improved A* and artificial potential field for unmanned surface vehicle formations. *Ocean Eng.* **2021**, *223*, 108709. [CrossRef]
18. 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]
19. He, Z.; Liu, C.; Chu, X.; Negenborn, R.R.; Wu, Q. Dynamic anti-collision A-star algorithm for multi-ship encounter situations. *Appl. Ocean Res.* **2022**, *118*, 102995. [CrossRef]
20. Zhou, S.; Wu, Z.; Ren, L. Ship Path Planning Based on Buoy Offset Historical Trajectory Data. *J. Mar. Sci. Eng.* **2022**, *10*, 674. [CrossRef]
21. Bi, J.; Gao, M.; Zhang, W.; Zhang, X.; Bao, K.; Xin, Q. Research on Navigation Safety Evaluation of Coastal Waters Based on Dynamic Irregular Grid. *J. Mar. Sci. Eng.* **2022**, *10*, 733. [CrossRef]
22. Yang, B.; Yan, J.; Cai, Z.; Ding, Z.; Li, D.; Cao, Y.; Guo, L. A novel heuristic emergency path planning method based on vector grid map. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 370. [CrossRef]
23. Chen, J.; Bian, W.; Wan, Z.; Yang, Z.; Zheng, H.; Wang, P. Identifying factors influencing total-loss marine accidents in the world: Analysis and evaluation based on ship types and sea regions. *Ocean Eng.* **2019**, *191*, 106495. [CrossRef]

24. Prpić-Oršić, J.; Vettor, R.; Faltinsen, O.M.; Guedes Soares, C. The influence of route choice and operating conditions on fuel consumption and CO_2 emission of ships. *J. Mar. Sci. Technol.* **2016**, *21*, 434–457. [CrossRef]
25. Harikumar, R.; Balakrishnan Nair, T.; Bhat, G.; Nayak, S.; Reddem, V.S.; Shenoi, S. Ship-mounted real-time surface observational system on board Indian vessels for validation and refinement of model forcing fields. *J. Atmos. Ocean. Technol.* **2013**, *30*, 626–637. [CrossRef]
26. Zhen, Q.; Wan, L.; Li, Y.; Jiang, D. Formation control of a multi-AUVs system based on virtual structure and artificial potential field on SE (3). *Ocean Eng.* **2022**, *253*, 111148. [CrossRef]
27. Shi, P.; Liu, Z.; Liu, G. Local Path Planning of Unmanned Vehicles Based on Improved RRT Algorithm. In Proceedings of the 2022 4th Asia Pacific Information Technology Conference, Bangkok, Thailand, 14–16 January 2022; pp. 231–239.
28. Rudin, N.; Hoeller, D.; Reist, P.; Hutter, M. Learning to walk in minutes using massively parallel deep reinforcement learning. In Proceedings of the Conference on Robot Learning, PMLR, Auckland, NZ, USA, 14–18 December 2022; pp. 91–100.
29. Chen, M.; Liu, W.; Wang, T.; Zhang, S.; Liu, A. A game-based deep reinforcement learning approach for energy-efficient computation in MEC systems. *Knowl.-Based Syst.* **2022**, *235*, 107660. [CrossRef]
30. Boute, R.N.; Gijsbrechts, J.; Van Jaarsveld, W.; Vanvuchelen, N. Deep reinforcement learning for inventory control: A roadmap. *Eur. J. Oper. Res.* **2022**, *298*, 401–412. [CrossRef]
31. Yuan, D.; Liu, Y.; Xu, Z.; Zhan, Y.; Chen, J.; Lukasiewicz, T. Painless and accurate medical image analysis using deep reinforcement learning with task-oriented homogenized automatic pre-processing. *Comput. Biol. Med.* **2023**, *153*, 106487. [CrossRef]
32. Wang, C.; Zhang, X.; Yang, Z.; Bashir, M.; Lee, K. Collision avoidance for autonomous ship using deep reinforcement learning and prior-knowledge-based approximate representation. *Front. Mar. Sci.* **2023**, *9*, 1084763. [CrossRef]
33. Zhai, P.; Zhang, Y.; Shaobo, W. Intelligent Ship Collision Avoidance Algorithm Based on DDQN with Prioritized Experience Replay under COLREGs. *J. Mar. Sci. Eng.* **2022**, *10*, 585. [CrossRef]
34. Goswami, I.; Das, P.K.; Konar, A.; Janarthanan, R. Extended Q-learning algorithm for path-planning of a mobile robot. In Proceedings of the Simulated Evolution and Learning: 8th International Conference, SEAL 2010, Kanpur, India, 1–4 December 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 379–383.
35. Konar, A.; Chakraborty, I.G.; Singh, S.J.; Jain, L.C.; Nagar, A.K. A deterministic improved Q-learning for path planning of a mobile robot. *IEEE Trans. Syst. Man Cybern. Syst.* **2013**, *43*, 1141–1153. [CrossRef]
36. Li, C.; Zheng, P.; Yin, Y.; Wang, B.; Wang, L. Deep reinforcement learning in smart manufacturing: A review and prospects. *CIRP J. Manuf. Sci. Technol.* **2023**, *40*, 75–101. [CrossRef]
37. Song, D.; Gan, W.; Yao, P. Search and tracking strategy of autonomous surface underwater vehicle in oceanic eddies based on deep reinforcement learning. *Appl. Soft Comput.* **2023**, *132*, 109902. [CrossRef]
38. Hu, H.; Yang, X.; Xiao, S.; Wang, F. Anti-conflict AGV path planning in automated container terminals based on multi-agent reinforcement learning. *Int. J. Prod. Res.* **2023**, *61*, 210. [CrossRef]
39. 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. [CrossRef]

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