Controlling a Cargo Ship without Human Experience Based on Deep Q-Network

Chen Chen[a], Feng Ma[b,c,*], Jialun Liu[b,c], Rudy R. Negenborn[d,c], Yuanchang Liu[e] and Xinping Yan[b,c]

[a]*School of Computer Science and Technology, Wuhan University of Technology, Wuhan, PR China*
[b]*Intelligent Transportation System Centre, Wuhan University of Technology, Wuhan, PR China*
[c]*National Engineering Research Centre for Water Transport Safety, Wuhan, PR China*
[d]*Department of Maritime and Transport Technology, Delft University of Technology, Delft, the Netherlands*
[e]*Department of Mechanical Engineering, University College London, Torrington Place, London, UK*

Abstract. Human experience is regarded as an indispensable part of artificial intelligence in the process of controlling or decision making for autonomous cargo ships. In this paper, a novel Deep Q-Network-based (DQN) approach is proposed, which performs satisfactorily in controlling a cargo ship automatically without any human experience. At the very beginning, we use the model of KRISO Very Large Crude Carrier (KVLCC2) to describe a cargo ship. To manipulate this ship has to conquer great inertia and relatively insufficient driving force. Subsequently, customary waterways, regulations, conventions are described with Artificial Potential Field and value-functions in DQN. Based on this, the artificial intelligence of planning and controlling a cargo ship can be obtained by undertaking sufficient training, which can control the ship directly, while avoiding collisions, keeping its position in the middle of the route as much as possible. In simulation experiments, it is demonstrated that such an approach performs better than manual works and other traditional methods in most conditions, which makes the proposed method a promising solution in improving the autonomy level of cargo ships.

Keywords: Deep Q-Network, Reinforcement Learning, Artificial Intelligence, Autonomous Ships

1. Introduction

Unmanned Surface Vessels (USVs), including autonomous boats and ships, have become hot spots of research in recent years. Especially for autonomous cargo ships, they are considered as a promising area in the shipping industry. Many researchers believe that developing autonomous cargo or general-purpose ships might be a wise way to address the problems of manpower shortage, safety, emission and pollution [1]. However, the progress of autonomous ships is slightly slower than that of autonomous boast, unmanned automobiles and aircrafts. Cargo ships, or general-purpose ships are quite special machineries, which must conquer huge inertia, great uncertainties of perception, recognition and manipulation in the navigation. Even for a human operator, such a job requires substantial knowledge and experience which could be obtained over years. Therefore, a practical artificial intelligence used for controlling a cargo ship for general purposes has not been invented. In general, the autonomy of a cargo ship is widely acknowledged to be consisting of several sequential processes or components, which includes sensing, recognizing, decision making and controlling [2]. Hence, many researchers are dedicated to imitating such components with various approaches. Furthermore, decision making is operating on two levels that include a long-distance planning and a short-distance, or short-term decision making. The short-

* Corresponding author. Feng Ma, Intelligent Transportation System Centre, Wuhan University of Technology, Wuhan, 430063, China. Email: martin7wind@whut.edu.cn.

term decision making in this area is also known as collision avoidance or path planning.

In the field of autonomous ships, the path planning and the motion controlling are usually treated as separate issues. This inevitably brings about that the planned paths often cannot be completed by cargo ships due to their characteristics in manoeuvrability. For example, the paths planned by A* and Rapid Random Trees (RRT) generally contain many continuous turns, which do not comply with the motion characteristics of ships. Furthermore, the path planning of a cargo ship has to take navigational knowledge into considerations, which brings many associated difficulties. In principle, navigational knowledge includes regulations, customary routes, and human experience, which are generally qualitative descriptions or guidance. To translate this knowledge into specific paths in changeable waterways is a challenge.

With the development of reinforcement learning (RL), a novel perspective has been put forward to address this problem. From the view of cognitive and learning science, all the creatures have the capability to navigate in complicated and changing environments. It is worth noting that they proceed with path planning and self-controlling simultaneously. Similarly, a captain or a helmsman of a ship might only have a rough plan of what to do under specific circumstances in advance, and he or she does not plan a precise path or a trajectory to follow. In fact, in the short-distance manipulation, the sensing, decision making, and controlling are always accomplished at the same time. The truth is that the decisions on paths are changing all the time.

Under this perspective, it is not unreasonable to use the artificial Neural Networks and RL-based algorithm to imitate manual processes in navigation, which is capable of regarding the short-range path planning and the motion controlling as a merged issue.

The traditional RL-based algorithms can only be used in solving simple problems such grid-like games. The available status and the candidate actions of an agent has to be discrete and limited, otherwise the training time will be infinite. In recent years, deep learning technologies has been introduced into RL-based algorithms with the feature of being capable of fast learning and environment cognizing. After a short period of self-study and training, the DQN-based algorithm achieved a high level of intelligence across a set of 49 classic Atari 2600 games. In these challenging tasks, they used the same algorithm, network architecture and hyperparameters [3]. In 2016, DeepMind developed AlphaGo Zero based on the Monte Carlo Decision tree search, which completely accomplished the training without any human experience and used a strategy of self-play to defeat a human master player in the Go game [4]. It can be seen that several relevant algorithms have proved that human experience might not be indispensable when the environments can be described appropriately, and the corresponding RL-based algorithms are designed properly.

Inspired by this, a novel short-term path planning and motion controlling approach based on DQN is proposed for autonomous cargo ships in this paper. The navigation environment is analysed with a Conventional Neural Network (CNN), and a modular type mathematical model (the so-called MMG model) is used to describe motion constraints of ships, the corresponding waterways and conventions are modelled based on an Artificial Potential Field (APF) model. The paper is organised as follows. Relevant references are reviewed in Section 2. A novel DQN-based approach is proposed in Section 3. By establishing a simulation world as a case study, the approach is validated in Section 4. Section 5 concludes this paper and provides directions for future research.

2. Literature review

The autonomy of general-purpose ships is inspired by the USV. The *Springer* USV and other projects started by colleges are the pioneers. The development of the *Springer* USV had originated from the Marine & Industrial Dynamic Analysis Research Group (MIDAS) at the University of Plymouth. The *Springer* had been developed for conducting environmental and hydrographic surveys in coastal waters [5]. Besides, the *DELFIM* is another USV for automatic marine data acquisition and served as an acoustic relay between a submerged craft and a supporting vessel, designed by Instituto Superior Técnico (IST), Lisbon [6]. In these projects, the navigation and control of the ship's path were studied as two subsystems. Fossen [7] built a basic framework for USVs and unmanned ships, which was divided into four control subsystems: engine system, communication system, sensors and navigation, guidance and control (NGC) system [8-10].

2.1. Path planning of autonomous ships

Generally, path planning is considered to be the most important function of USV. In relevant researches, the A* algorithm is widely used, which is

designed to find a short path between start and destination based on cost functions within a grid map [11]. For instance, Casalinot et al. [12] addressed an obstacle avoidance problem and computed a real-time path in harbour field based on the A*. Campbell and Naeem [13] presented a modified A* algorithm for path planning considering the International Regulations for Preventing Collisions at Sea (COLREGs). Generally, USVs always navigate in a mapping environment with obstacles information as prior knowledge when using A*-based algorithms.

However, cargo ships always have to face dynamic obstacles and various navigation regulations. Hence, the conventional form of A*-based algorithms might not be applicable. APF and other methods are more appropriate. Ma and Chen [14] adopted APF to describe the collision potentials caused by buoys, piers and encountered vessels and then estimated the collision probabilities. Wang et al. [15] suggested a multi-ship collision avoidance and path planning solution based on APF, which analysed give-way and stand-on ships situation. This method improved the ship domain model by taking speed and course into consideration, which makes it more accurate. Lazarowska [16] introduced an APF-based path planning method for ships at open sea. However, the APF-based algorithms must face the problem of local minima [17]. Line-of-Sight (LOS) guidance is another commonly used approach in path planning, which is considered as the most popular guidance law in use for surface vessels. Zereik et al. [18] described a navigation guidance and control system based on LOS and applied it to unmanned marine vehicles considering only stationary obstacles. Moe and Pettersen [19] extended set-based guidance theory to an underactuated USV and presented a switched guidance and control system. This guidance law can ensure the safety by creating a safe radius around moving obstacles and complying with COLREGs. Liu et al. [20] made use of the fast marching method (FMM) algorithm to search for an optimal collision-free trajectory and a new waypoint-generator based on the LOS to facilitate the trajectory tracking of the *Springer* USV, which showed that it can be seamlessly integrated with the *Springer*'s exiting autopilot to achieve full autonomy.

2.2. Motion controlling of autonomous ships

In most researches, after path planning, the following issue is to maintain the velocity and to control the ship on the planned path, namely motion controlling. The corresponding work can be traced back to an autopilot which appeared in the 1920s [21]. The autopilot is a kind of facility to keep the heading on a setting value by directly controlling the rudder, which is still an essential means of assistant driving. The typical research is listed as follows. Miao et al. [22] designed an adaptive PID algorithm for the heading control system of USV. Liu et al. [23] proposed a model predictive control (MPC) approach based on adaptive LOS guidance for path following control of autonomous surface vehicles and verified by simulation experiments without disturbances and with disturbances. Sharma and Sutton [24] developed a modified nonlinear MPC algorithm based on a genetic algorithm for the USV. Such researches are used to improve the adaptive capacity of different controllers in changeable environments, which can be regarded as improvements of autopilot. Although relevant research has made much progress in these years, an autopilot that is capable of replacing a helmsman has not been invented yet. The conventional autopilots are designed to steer the wheel and keep the course to reduce human labour while more advanced skills of a helmsman like path planning and decision making are not fulfilled yet. As discussed previously, the helmsman must make rapid and continuous decisions in ever-changing waterways. To divide the controlling behaviours into two parts, path planning and motion controlling might be open to question.

2.3. The application of RL in autonomous ships

RL takes path planning and motion controlling as a whole system and trains such a system by interacting with environments, which provides a new idea for navigating autonomously and adaptively in an unknown environment. RL provides agents with the capability of interacting with the environment in real-time and tries constantly to obtain an appropriate strategy. The outputs of the agents are actions of rudder or engine, which can be used to make continuous decisions for smart ships in real situation. Blekas and Vlachos [25] investigated RL for the path planning of an autonomous triangular marine platform in unknown environments under various environmental disturbances. They also simulated system and sensor failures, and the results showed that the algorithm performed well, proving its robustness. Considering the dynamic characteristics of a vehicle and disturbance effects in ocean environments, Yoo and Kim [26] presented an RL based algorithm generates a near-optimal path and compared it with A*, RRT and dynamic programming

algorithms when addressing the Zermelo's problem with the same simulation setting. The results proved that the path obtained by the RL based method is more satisfied, which has been further validated by a field experiment in the western sea of Korea. Chen et al. [27] proposed a path planning and motion control approach based on Q-learning, and compared this approach with A* and RRT methods. The RL algorithm has achieved promising results in autonomous driving, but the huge magnitude of the state space and the action space is still a problem for a ship.

To address the similar problem, DeepMind put forward DQN [28], which attracted widespread attention. DQN applied the achievements of cognitive neuroscience to the training of deep neural networks and solved the problem of state space explosion using experience replay and a separate target network. In 2018, a British company, Wayve [29] realized that a car without any prior knowledge can learn lane tracking skills in only 30 minutes using a model-free deep RL algorithm. DeepMind [30] presented a dual-pathway agent architecture, which can learn to navigate in a city-scale, real environment using visual navigation scheme. This method uses end-to-end RL for training and Google Street View for its photographic content. NVIDIA [31] proposed an end-to-end approach that is capable of steering a car without human knowledge. The successful application experience of DQN in unmanned vehicle field was quickly applied to unmanned ships. Cheng and Zhang [32] made use of DQN to learn obstacle avoidance for underactuated unmanned vessels. In their research, they designed a reward function based on target approaching, speed modification, and attitude correction.

Based on these references, it can be concluded that a DQN-based agent can achieve autonomous end-to-end learning from perception to action like humans. DQN inputs raw sensory data such as vision. Then, CNN can catch the hidden features by training, making the algorithm's versatility and mobility satisfactory. Moreover, with the help of RL, these researches also proved that the planning and controlling can be treated as a merged issue, while human knowledge is not indispensable. Since the traditional ways of developing ship autonomy has meet some bottlenecks, the DQN-based method might be worth a try.

3. A proposed approach

As discussed previously, the long-term route planning will produce a rough lane or a rough route for a ship to follow. The following task is to follow such a lane while avoiding collisions. In this section, a DQN-based approach is proposed to accomplish this task. At the beginning, rigorous ship mathematical models are derived to establish the foundation for training and simulations, making the simulated ship's manoeuvring characteristics consistent with a real ship. Then, static and dynamic obstacles (encountering vessels) are placed in the simulation world. The proposed DQN-based algorithm is then introduced to build the artificial intelligence of controlling.

3.1. Mathematical modelling of ship motions

The success of RL-based algorithms requires sufficient training episodes. Since such trainings generally take thousands of rounds, it can only proceed in a simulation world or with a lot of real-like data. Therefore, it is essential to make such a simulation world consistent with the real world. In a simulation world, the problem of how to infer the coming status after a certain action should be addressed at first.

The ship manoeuvring model is used to forecast the state changing of a ship when it takes a specific action, making the training environment consistent with the real world. In this area, ship manoeuvring motions are generally presented with a standard three degree-of-freedom MMG model [33] that considers surge, sway, and yaw for simplification. Fig. 1 illustrates the static earth-fixed $o_0 - x_0y_0z_0$ and the dynamic body-fixed $o - xyz$ coordinate systems. The origin of $o - xyz$ locates at the middle of the ship o. x-, y- and z- axes are positive to the bow of a ship, the starboard of the ship, and downwards of the water surface xy respectively. Assuming that the ship presented in Fig. 1 is manoeuvring at surge speed u and sway speed v, the ship speed is $V = \sqrt{u^2 + v^2}$. The heading angle is ψ. The ship is turning with a rudder angle δ at yaw rate $r = \dot{\psi}$.

The MMG model used in this research describes the hydrodynamic force and the moment in three aspects: hull, propeller and rudder. The motion equations are expressed as the following [34]:

$$\begin{cases} (m + m_x)\dot{u} - (m + m_y)vr - x_G m r^2 = X_H + X_P + X_R \\ (m + m_y)\dot{v} + (m + m_x)ur + x_G m\dot{r} = Y_H + Y_R \\ (I_Z + x_G^2 m + J_Z)\dot{r} + x_G m(\dot{v} + ur) = N_H + N_R \end{cases} \quad (1)$$

where subscripts H, P, and R denote hull, propeller, and rudder, respectively, with force (X and Y) and moment (N). m is the ship mass, m_x and m_y are added mass due to motions in surge and sway directions. $\dot{u}$, $\dot{v}$

and $\dot{r}$ are surge, sway and yaw acceleration, and I_z, J_z, are the moments of inertia, where $I_Z \approx (0.25L_{pp})^2$m. If not particularly specified, the parameters, such as velocity (u, v, r, and V), acceleration ($\dot{u}$, $\dot{v}$, and $\dot{r}$), force (X and Y), and moment (N) are defined on or around midship. According to the MMG model, the trajectory and status of a ship can be predicted under different initial conditions (positions, speeds, rudder angles and different angular velocities).

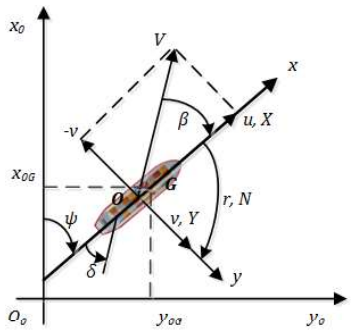


Fig.1. Applied earth-fixed and body-fixed coordinate systems

3.2. Deep Q-Network

Based on the ship model supported by MMG, DQN is adopted in this research to build the artificial intelligence for navigation and collision avoidance along a planned route or a lane. As discussed, RL-based methods consider tasks in which the agent interacts with an environment through a sequence of observations, actions and rewards. In RL, rewards represents the gains or losses after specific actions. The function of estimating rewards is called action-value function. The goal of RL is to learn an optimal policy for sequential decision problems by maximizing a cumulative future reward [35].

To estimate the action-value function, the Bellman equation is used as an iterative update. In practice, most problems are difficult to learn action values in all states separately. Instead, DQN uses a deep CNN to approximate the optimal action-value function.

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a) = \max_{\pi} E[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots | s_t = s, a_t = a, \pi] \quad (2)$$

where r_t is the reward at each time-step t, achievable by an action policy $\pi = P(a|s)$, after making an observation (s) and taking an action (a).

DQN is a multi-layered neural network that $Q(s,a;\theta_i)$ is used to evaluate action values, where θ_i are the parameters of the Q-network at iteration i. The agent experience $e_t = (s_t, a_t, r_t, s_{t+1})$ are stored at each time-step t in a data set $\mathrm{D}_t = \{e_1, \cdots, e_t\}$. During learning, we apply Q-learning updates, on samples (or minibatches) of experience $(\mathrm{s}, \mathrm{a}, \mathrm{r}, \mathrm{s}') \sim U(D)$, drawn uniformly randomly from the pool of stored samples. The loss function is [36]:

$$L_i(\theta_i) = E_{(s,a,r,s')\sim U(D)}\left[\left(r + \gamma \max_{a'} Q(s', a'; \theta_i^-) - Q(s,a;\theta_i)\right)\right] \quad (3)$$

in which γ is the discount factor determining the agent's horizon, θ_i are the parameters of the Q-network at iteration i and θ_i^- are the network parameters used to compute the target at iteration i.

In this research, DQN lays the foundation of artificial intelligence for autonomous ships by selecting actions in a way that maximizes future rewards. In order to remove correlations of data, a Dueling Network Architecture was established [37], which includes two CNN. One is called the Q network, and the other is called the target Q network. Moreover, the architecture and initial neural network parameters of the two neural networks are the same. Hence, this section only introduces the architecture of the target Q Network for simplicity. This CNN is made up of three convolutional layers and then followed by two fully connected layers and one pooling layer, as shown in Fig. 2.

The agent observes the sailing situation from the top view as shown on the left-top of Fig. 2. Then, the sailing situation including the waterways, encountered vessels are then simplified as sequential images or frames. Subsequently, every four fixed-interval frames will be used as the input of the CNN. This research simplifies the input as four 80×80 pixels resolution images to lower the burden of calculation. Then, the first hidden layer convolves 32 filters of 8×8 pixels with stride 4 with the input images and applies a rectifier nonlinearity. Followed by a pooling layer, it adopts MaxPooling, with sliding window 2×2 and stride 2. The second hidden layer convolves 64 filters of 4×4 with stride 2, again followed by a rectifier nonlinearity. This is followed by a third convolutional layer that convolves 64 filters of 3×3 with stride 1 followed by a rectifier. The final hidden layer is fully connected and consists of 512 rectifier units [3]. Since the Dueling network is a two-stream architecture, these units are divided into two parts and sent to separate streams. Each stream contains a fully connected layer. The final layer combines the output of the two streams, and the output of this network is a set of Q-

values for each valid action [4]. Moreover, the output is the Q-value from executing a certain action.

In this model, CNN is used as an unsupervised tool that is capable of learning features in the ship controlling and collision avoidance based on sequential images. With the help of DQN, the ship will build a memory or experience, which instructs itself the situation that might happen after a specific action, represented in simplified and sequential figures.

3.3. Reward functions

The following task is to teach artificial intelligence which consequence or situation is appreciated in a navigation. For example, collisions should be avoided and punished, whereas approaching a destination should be rewarded. Based on the sequential figures given by CNN, this procedure is translated as a reward function. An appropriate reward function will make the manoeuvring behaviours of a ship similar to a helmsman's commands. Generally, each factor that affects the choices of a helmsman should be described as appropriate rewards or punishments. Since many factors have a direct or indirect influence on the decisions of the helmsman, it might take enormous effort to enumerate these factors perfectly in a reward function, which should be a huge engineering problem. Hence, this research only selects four typical factors from different perspectives to design a reward function for simplicity, aiming to prove the applicability of the proposed approach. More factors can be discussed in the future. A helmsman might take more factors into consideration in the navigation.

(1) Heading (Course) deviation. Since a route has already been set in advance, the ship should keep its heading consistent with the direction of this route as much as possible. Hence, the included angle between the ship's heading and the route can be considered as the quantification of the consistency. In this research, the reward of this factor is denoted as,

$$R_{angle} = \begin{cases} \lambda_{angle}, & -\Delta < angle < \Delta \\ -\lambda_{angle}, & else \end{cases} \quad (4)$$

where $angle$ represents the included angle between the ship heading and the route or the lane. λ_{angle} is a constant that is greater than 0. When $angle$ is between - Δ and Δ, this ship's heading can be regarded as normal, then the reward is set to λ_{angle}. When $angle$ is another value, this ship's heading can be regarded as abnormal, then the reward is set to $-\lambda_{angle}$. This policy will encourage the ship to keep its heading parallel with the direction of the route.

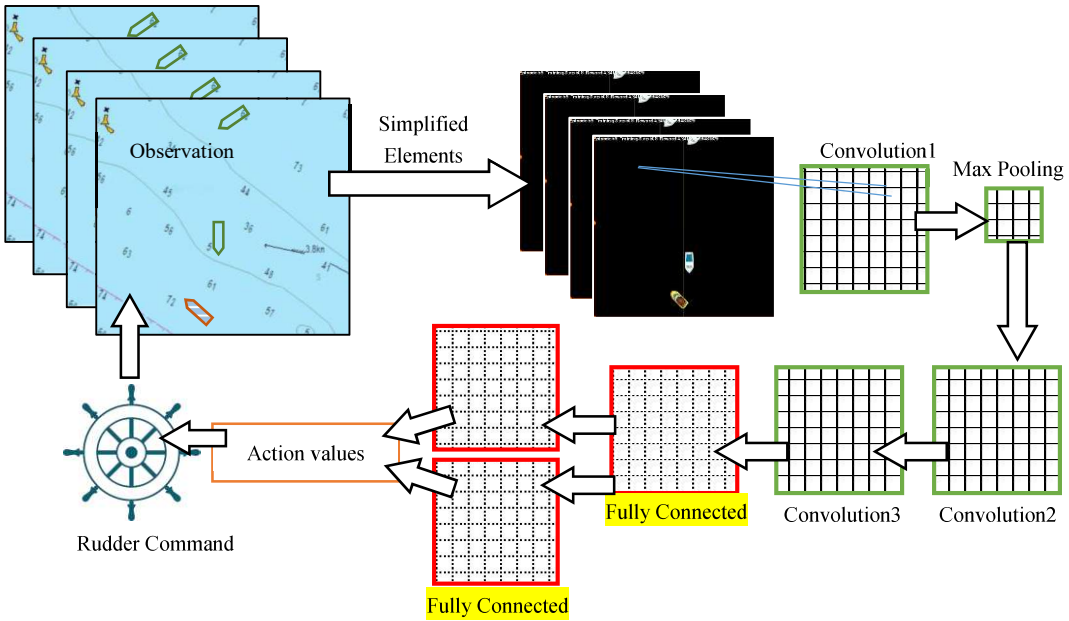


Fig.2. The architecture of the DQN in this research

(2) Lane deviation. Besides the included angle, the deviation to the planned route is another essential factor while navigating. As it is well known, vehicle usually travels within a lane. Similarly, ships always travel along a specific or customary route. The only difference is that the route of a ship is much wider than a lane of an automobile. To model this factor, an APF-based model is introduced to quantify the deviation to the middle of the waterway or lane. Refer to the research conducted in the driverless car, a lane potential model is used [38], which is shown in Fig. 3.

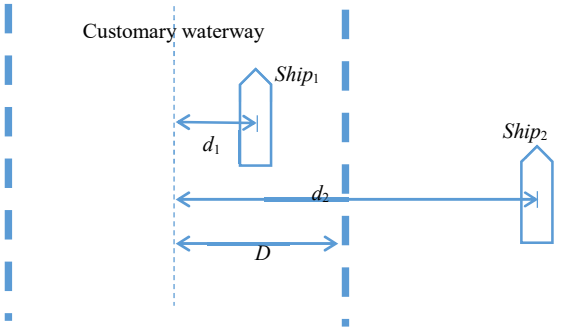


Fig.3. The modelling of lane deviation

D is a constant, which denotes the 1/2 width of this lane. d, d_1 and d_2 denote the displacement to the middle of the route. In Fig.3, Ship1 is still in the route, but there is a small deviation to the middle. Ship2 is already out of the route. Based on the lane potential, the degree of the displacement deviation can be denoted as $(d - D)/D$. In our research, we simply do not encourage the ship to sail out of the lane or the route, making the training easier. Hence, this reward is denoted as,

$$R_{lane} = \begin{cases} (1 - \frac{d-D}{D}) \cdot \lambda_{lanein}, d < D \\ -\lambda_{laneout}, d \geq D \end{cases} \quad (5)$$

where λ_{lanein} denotes the reward value when the ship is sailing in the middle of the route, $-\lambda_{laneout}$ denotes the punishment when the ship is out of the route. Moreover, if the target is out of the lane, the current episode of training can be considered as failed and the training process will restart.

(3) Collision. To avoid collision is the first priority for ships. When colliding with some objects, such as encountering with other ships, shallow water, rocks, or a coastline, the ship should be punished. This reward is denoted as,

$$R_{collision} = \begin{cases} -\lambda_{collision}, \ if \ collision \\ 0, \ else \end{cases} \quad (6)$$

where $\lambda_{collision}$ denotes the punishment value. Moreover, if the target collides with something, the present episode of training can be considered as failed and the training process will restart too. In particular, $\lambda_{collision}$ should be assigned with a relatively large value, since avoiding collisions should always be the priority.

(4) Ship domain. Ship domain is a concept invented by traditional marine technologies [39]. In practice, collision avoidance is very difficult for a cargo ship due to its large tonnage, huge inertia, and relatively weak driving forces. Therefore, an imaginary region, namely a ship domain, should be defined in advance which is generally 7 times longer than its length and 3 times wider than its width. When an obstacle has entered this area, caution warnings will be triggered, which is a tense situation for all the crews. An experienced helmsman should try to avoid this situation. This reward can be denoted as,

$$R_{danger} = \begin{cases} -\lambda_{danger}, \ in \ ship \ domain \\ 0, \ else \end{cases} \quad (7)$$

where $-\lambda_{danger}$ denotes the punishment when some other object enters the ship's domain.

Based on these four factors, the global reward function can be defined as,

$$R = R_{angle} + R_{collision} + R_{danger} + R_{channel} \quad (8)$$

As elaborated previously, the factors affect the ship are more than these four discussed in this section. The reason for choosing these four lies in that they are coming from different perspectives. In the past, it was difficult to consider them all in one framework. More factors based on another perspective can be modelled similarly. It is worth noting that the coefficient values of Eq. (4) to Eq. (7) will be discussed in the case study.

Based on Eq. (8), the training process can be run in accordance with Fig. 8. After sufficient training, the artificial intelligence of manipulating this ship can be obtained. The details of training will be given in the following section.

4. A case study

To demonstrate and to validate the proposed approach, a case study is given in this section

4.1. Experimental platform

The simulation, training and validation environment are built with PyCharm, which uses Python 3.6.8 as the coding language, TensorFlow-GPU 1.14.0 as

the machine learning library and Gym 0.14 as the RL library. The computing and training are accomplished on a CORE i7 9700K PC, a GTX 1660 Ti as the GPU, Windows 10 Professional as the OS.

As elaborated on Section 3.1, in this environment, a ship is described by a rigorous hydrodynamic model of MMG. Meanwhile, this research chooses a KVLCC2 tanker as the target ship. The free-running model tests of this ship carried out by the Maritime Research Institute Netherlands (MARIN) are referred for the validation of the mathematical model [40-41]. The ship model is scaled to 7 metres with a scale factor of 45.7. Simulations are performed with the model-scale ship parameters as presented in Table 1. More detailed information can be found in the reference [33]. With the help of MMG, the resulting status after a certain operation can be predicted.

Table 1

Basic parameters of the KVLCC2 model within the MMG model

Attributes	Value
Length (m)	7
Width (m)	1.17
Draught (m)	0.46
Block coefficient (-)	0.81
Propeller revolution per second (1/s)	10.4
Range of rudder angles (deg)	-35~35

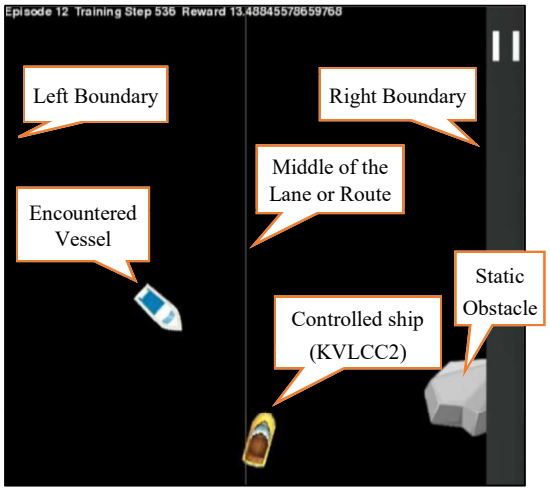


Fig.4. Main UI of the experiment platform

The navigation environment and the corresponding software program have been presented in Fig. 4. As represented in this figure, the yellow ship at the bottom is the controlled ship, which is based on the KVLCC2 model. In particular, the main layout represents a route or a lane that has already been planned by the long-term route planning. The observed area is 600×600 pixels, which represents a 128-metre-width planned waterway. The left border represents the left boundary of this route, and the right border represents the right boundary of this waterway. The shallow green straight line in the middle represents the central line of this route or lane. The white vessel represents the encountering vessels, and the grey rock icons represent the static obstacles. The controlled ship should try to follow the central line of the route, while sailing forward and avoiding all the collisions.

Moreover, the inference interval of this platform is one second per frame or step. Every single frame in this platform stands for one second in the real world.

4.2. Basic modelling of DQN

As elaborated in Section 3.2, in the learning process, the CNN of DQN selects 32 groups of 4 frames of continuous images as the input in each time step. However, the size of the input frame or picture is 32 bits × 600 pixels × 600 pixels × 4 frames, which is too huge causing the parameter update slowly. Therefore, the original sequential images have been scaled to 80 × 80 pixels before greyed and binarized as elaborated in Section 3.2. As a result, the size of input data is reduced to 32 bits × 80 pixels × 80 pixels × 4 frames. Moreover, the background has been set to plain black, and the grey value of the controlled ship and obstacles is set to 255, making the binarization easier. After such preprocessing, the size of the input is compressed to 1 bits × 80 pixels × 80 pixels × 4 frames, which is demonstrated in Fig. 5.

Subsequently, the input to the CNN consists of 32×80×80×4 image. Meanwhile, the output is the Q-value after executing a certain action. In other words, the output of the CNN in each state is equal to the size of the action space, corresponding to the action-value function of each action in a certain state.

According to the CNN output, a rudder angle can be selected. We also use a simple frame-skipping technique that the agent sees and selects actions on each 4th frame instead of all frames, and its last action is repeated on skipped frames [42].Then, the selected rudder angle is input into the hydrodynamic model of the ship to obtain the trajectory of the ship. Subsequently, 4 consecutive frames of navigation environment images are input into the CNN to continue the above process. After repetitive training processes, the agent ship will have the intelligence of avoiding obstacles.

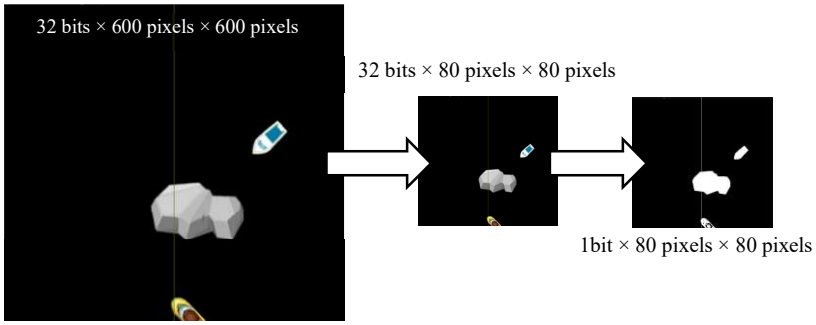


Fig.5. Pre-processing of the input images

In this research, the training is carried out in a way that gradually increases the complexity and difficulty of the experimental environment that includes two steps: static obstacles in a static scenario; static and dynamic obstacles in a continuous route.

4.3. Basic training in a static environment

Static obstacles in a static environment are used to build a basic consciousness of collision avoidance for the controlled ship in the experiment introduced in Section 4.1.

In this training, there are two static obstacles, which are located at (0,300) and (350,100) with sizes 240×145 and 180×109, as shown in Fig. 6. The ship is designed to sail from the bottom to the top with the help of DQN proposed in Section 3. In practice, the available actions of a ship are very large, including different rudder angle and different propeller speed. However, a cargo ship merely changes its propeller speed at open sea due to the features of engines. Hence, it is fair to set the propeller speed as a static value, which is 10.4 round per seconds in this research. Particularly, 10.4 round per seconds is also a normative testing value in the testing of ship's manoeuvrability. To make the convergence faster, the rudder angle of the ship is set to only 3 options, [-3°, 0°, 3°] to reduce the training time. In fact, other values of rudder angle, +1°, -1°, +2°, -2°, +4°, -4°, +5°, -5°, +7°, -7°, are also applicable. However, the training of DQN will take an unbearable long time when the actions are too many. Additionally, the undetermined coefficients in Section 3 are assigned preliminarily, where $\Delta = 7$, $\lambda_{angle} = 1$, $\lambda_{collision} = 100$ and $\lambda_{danger} = 10$. In the latest research, these coefficients can be further analysed and learned by imitate reinforcement learning, which will be discussed in the future.

According to the distance between the start and the destination and the speed of the agent ship, the ship can reach the destination in less than 100 steps as long as it does not turn in circles or deviate from its course. For this reason, a maximum of 100 steps is specified for one exploration in order to accelerate the convergence of this algorithm referring to other similar research. It means that if the controlled ship does not reach the destination in 100 steps, the exploration is considered to be a failure, and the ship will return to its start point to re-explore.

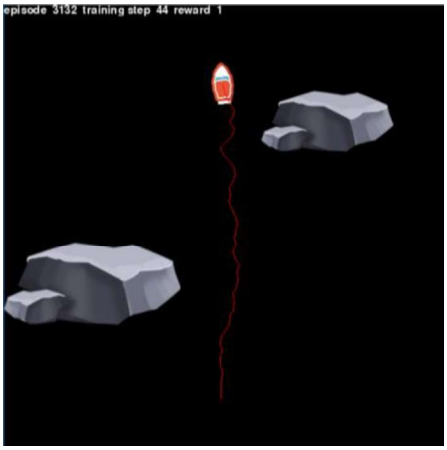


Fig.6. Basic training of the DQN

The convergence process of the proposed approach in this static environment is shown in Fig. 7. The X-axis represents the number of training episodes, and the Y-axis represents the number of successful arrivals in 100 episodes. It can be inferred from Fig. 7 that the agent began to have the ability to avoid static obstacles after about 2,400 times of exploring. Such an ability became stronger in accumulated training. When the ship explored more than 3,400 times, it can reach the destination faithfully, and the parameters of the neural networks were stored.

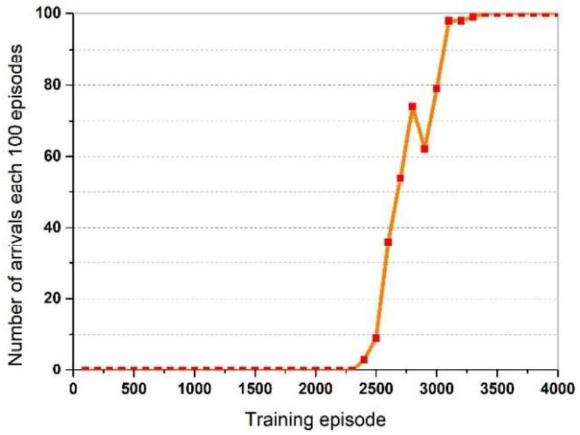


Fig.7. The convergence of DQN algorithm in static scenarios

4.4. Training in an environment with dynamic obstacles

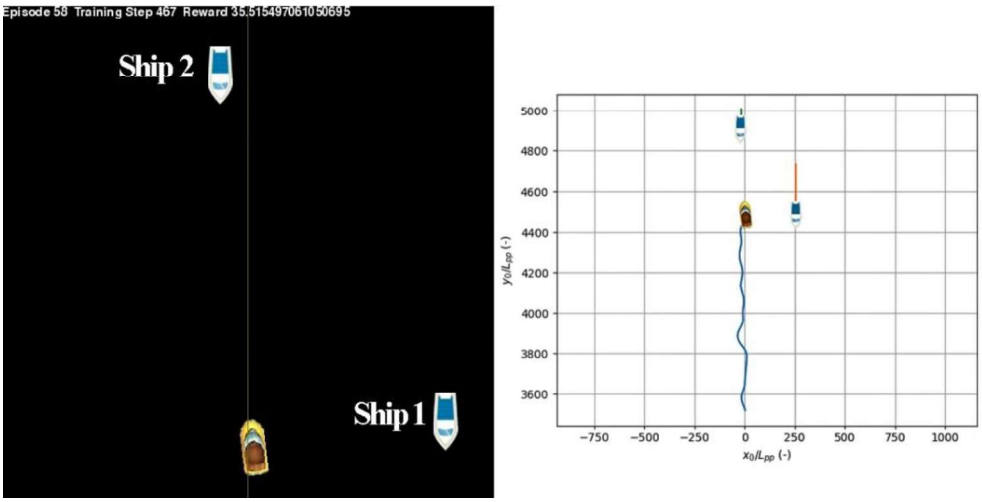
After the basic training has been accomplished in the previous section, the DQN-based ship can be further trained in a more complicated environment, which better resembles practical navigation situations. First of all, the route is endless, the controlled ship should always move upwards. Moreover, the encountered vessels and static obstacles will be refreshed randomly at any place. Therefore, the controlled ship has to face various kinds of situations.

All the constants used in this section share the same values as the previous section. Such a training took 13 hours, much longer than the first step. Then, the training CNN and the target CNN had converged eventually. According to the CNN output, the appropriate actions, which is actually the choice of rudder angle can always be selected. The output artificial intelligence provides a satisfying performance in avoiding collisions while keeping itself in the middle of the planned route. It is worth noting that this ship is set to be imbalanced on the rudder on purpose making the control of the ship difficult.

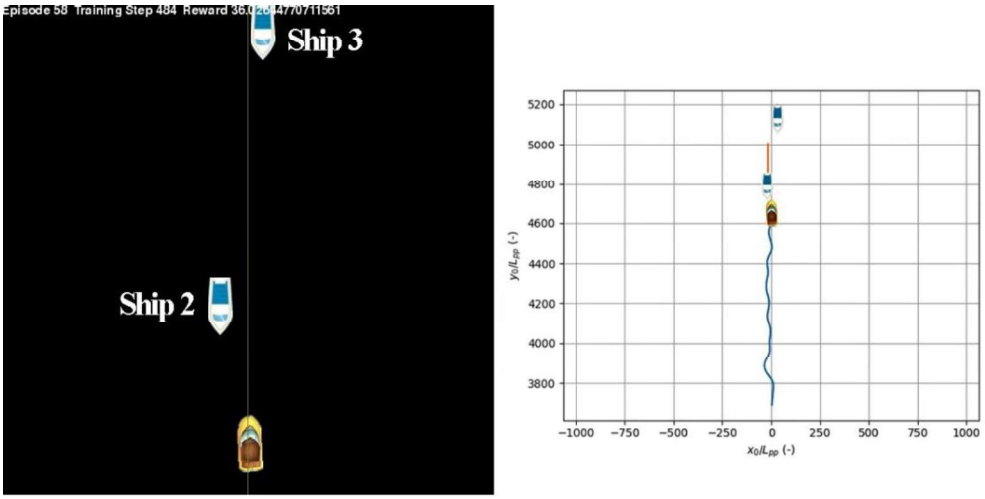
Fig. 8 is a typical collision avoidance of the controlled ship after about 80 hours of training, which contains 12 sub-figures listed according to the time sequence. Figures on the left represent the controlled ship and its surroundings on a specific time. The corresponding trajectories of the controlled ships and the encountered vessels are represented in the sub-figures on the right side. In particular, the trajectories of the controlled ship are denoted as blue curves, while the trajectories of the encountered vessels are denoted as red curves.

Based on Fig. 8, it can be inferred that the controlled ship had encountered with four encountering ships, named Ship 1, Ship 2, Ship 3 and Ship 4. Ship 1 is far away from the smart ship as shown in Fig. 8(a), and the controlled ship kept its heading with no further action. In Fig. 8(b), a typical head-on encountering happened. The controlled ship turned starboard and passed through on the port side of Ship 2, as shown in Fig. 8(c). Afterwards, the controlled ship returned to the middle of the planned route and encountered with Ship 3 was on the left, as illustrated in Fig. 8(d) and Fig. 8(e). Fig. 8(f) shows that the controlled ship turned to the left to avoid Ship 3. Then, the controlled ship succeeded in avoiding the collisions with Ship 4 using a very sharp turn. The video of the trained ship sailing in the dynamic waterway can be found online (https://youtu.be/KddYixKSn-4).

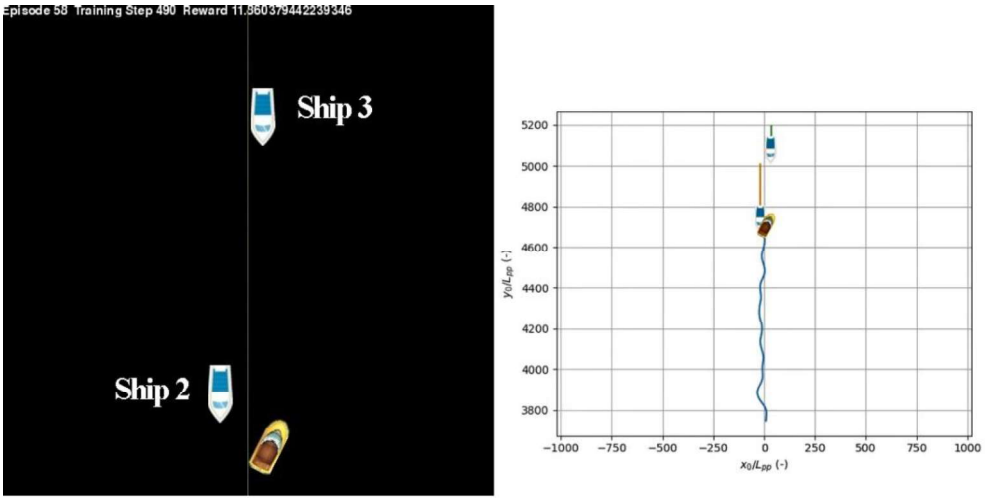
With this artificial intelligence, the controlled ship is capable of surviving in these difficult simulation scenarios for more than 2,000 steps in general. In particular, the collision happens only when the collision situation seems evitable since the encountered vessels and rocks are refreshed frequently and randomly.


Episode 58 Training Step 467 Reward 35.515497061050695
Ship 2
Ship 1
5000
4800
4600
4400
4200
4000
3800
3600
−750
−500
−250
0
250
500
750
1000
y_0/L_{pp} (-)
x_0/L_{pp} (-)

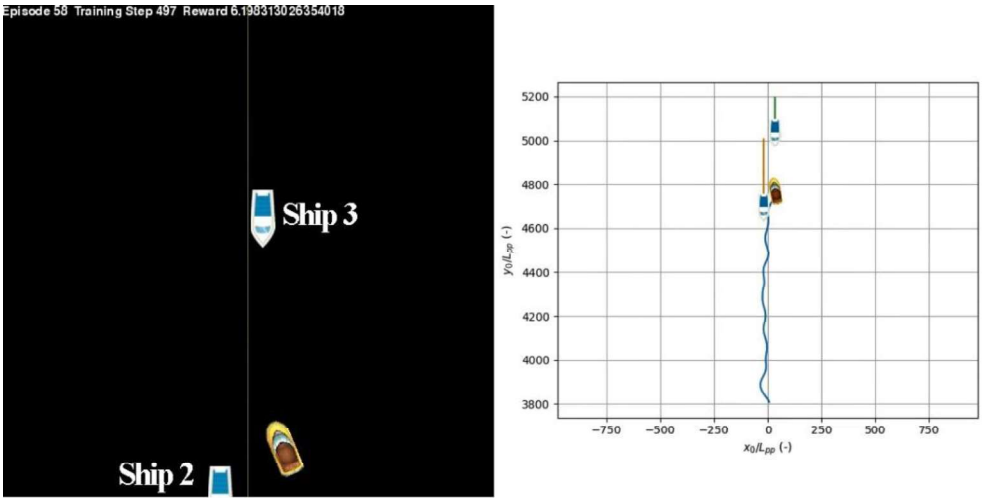
(a)


Episode 58 Training Step 484 Reward 36.026047707711561
Ship 3
Ship 2
5200
5000
4800
4600
4400
4200
4000
3800
−1000
−750
−500
−250
0
250
500
750
1000
y_0/L_{pp} (-)
x_0/L_{pp} (-)

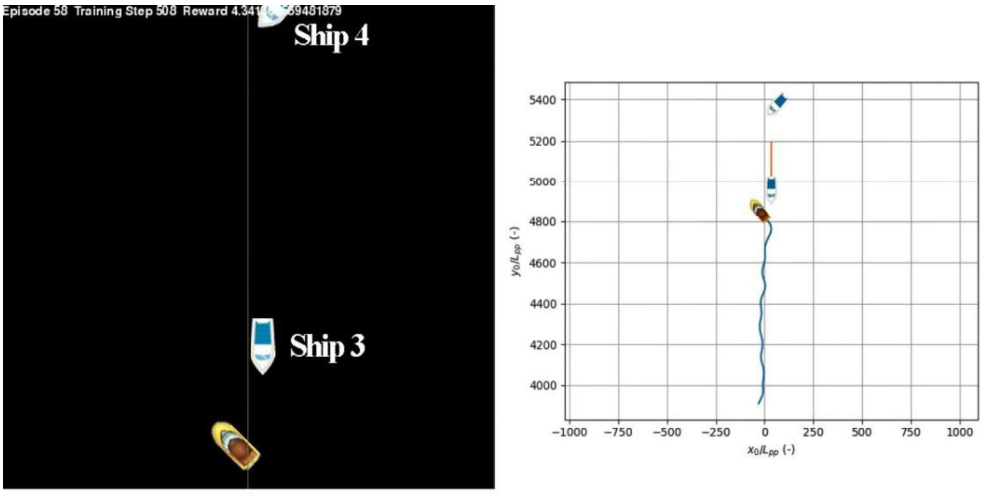
(b)


Episode 58 Training Step 490 Reward 11.960379442239346
Ship 3
Ship 2
5200
5000
4800
4600
4400
4200
4000
3800
−1000
−750
−500
−250
0
250
500
750
1000
y_0/L_{pp} (-)
x_0/L_{pp} (-)

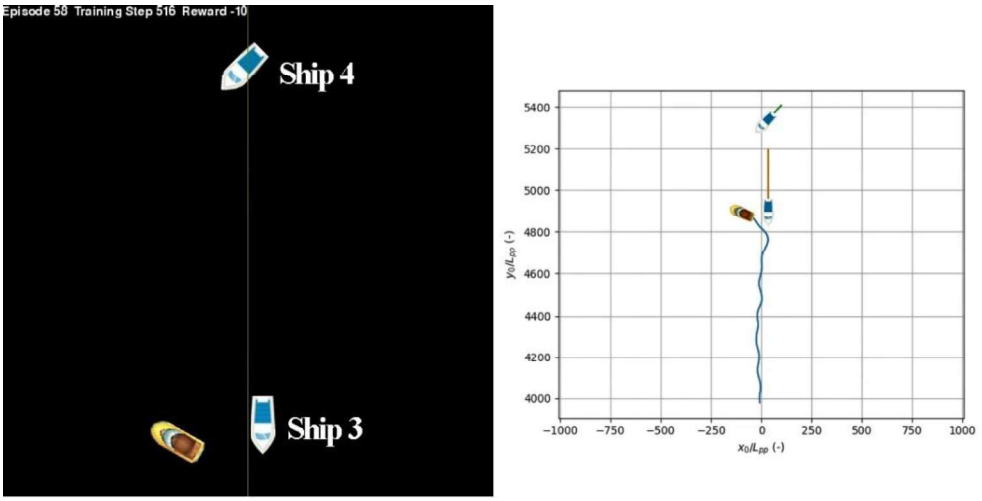
(c)


Episode 58 Training Step 497 Reward 6.198313026354018
Ship 3
Ship 2
5200
5000
4800
4600
4400
4200
4000
3800
−750
−500
−250
0
250
500
750
x_0/L_{pp} (-)
y_0/L_{pp} (-)

(d)

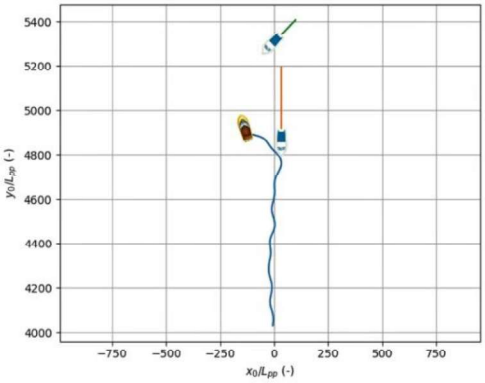

Episode 58 Training Step 508 Reward 4.341
59481879
Ship 4
Ship 3
5400
5200
5000
4800
4600
4400
4200
4000
−1000
−750
−500
−250
0
250
500
750
1000
x_0/L_{pp} (-)
y_0/L_{pp} (-)

(e)


Episode 58 Training Step 516 Reward -10
Ship 4
Ship 3
5400
5200
5000
4800
4600
4400
4200
4000
−1000
−750
−500
−250
0
250
500
750
1000
x_0/L_{pp} (-)
y_0/L_{pp} (-)

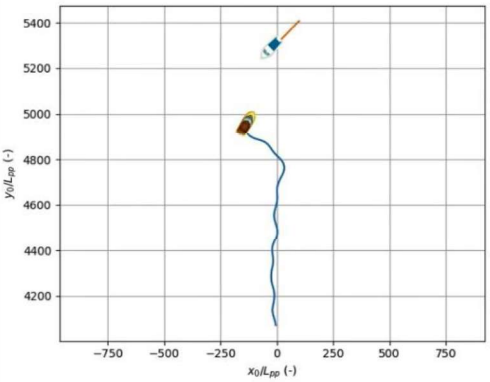
(f)


Episode 58 Training Step 521 Reward 10
Ship 4
Ship 3


5400
5200
5000
4800
4600
4400
4200
4000
−750
−500
−250
0
250
500
750
y_0/L_{pp} (-)
x_0/L_{pp} (-)

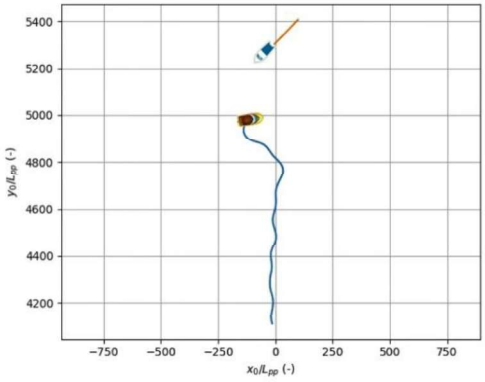
(g)


Episode 58 Training Step 525 Reward -10
Ship 4

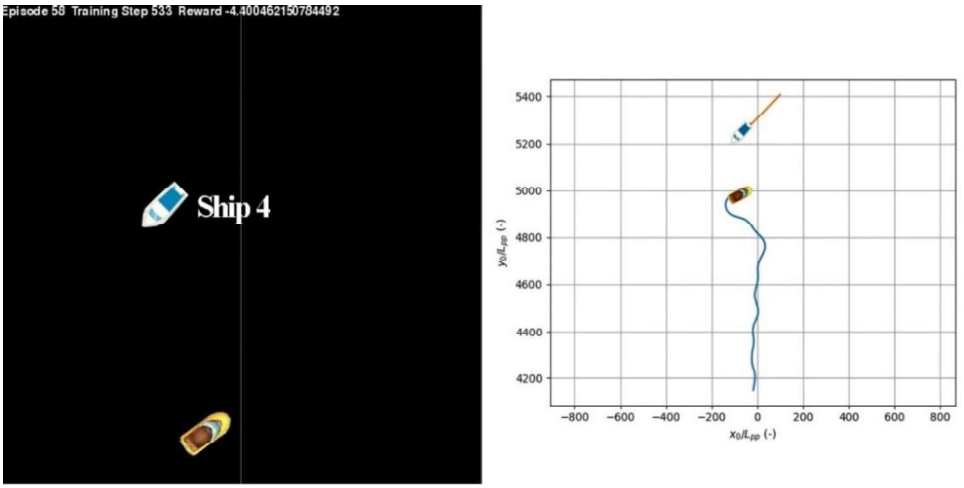

5400
5200
5000
4800
4600
4400
4200
−750
−500
−250
0
250
500
750
y_0/L_{pp} (-)
x_0/L_{pp} (-)

(h)

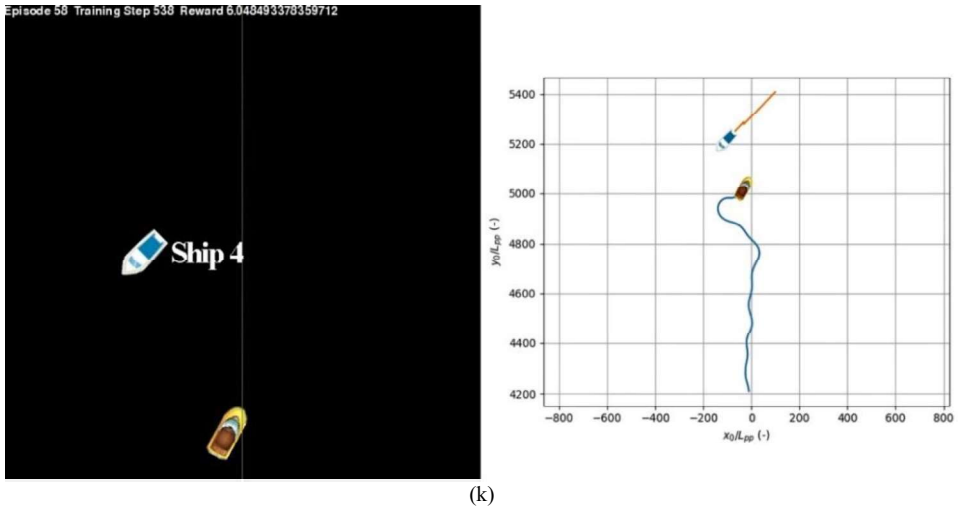

Episode 58 Training Step 528 Reward -10
Ship 4


5400
5200
5000
4800
4600
4400
4200
−750
−500
−250
0
250
500
750
y_0/L_{pp} (-)
x_0/L_{pp} (-)

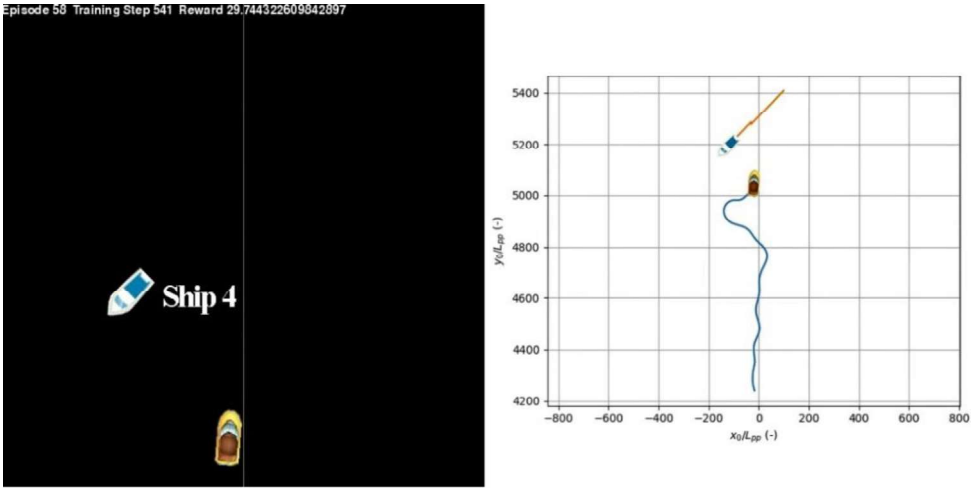
(i)



(j)



(k)



(l)

Fig.8. Dynamic ship collision avoidance

To validate this approach, manual works (human players), a traditional LOS/PID-based algorithm [22] and an APF-Vector-based algorithm [16] are used to compare in the same simulation world. In particular, the manual works is used a special baseline. As described above, high-speed encountered vessels might appear continuously in this narrow waterway. Hence, the task of controlling this ship to avoid collisions in this world is quite difficult. In this occasion, it is reasonable to use how many steps (actually seconds in the simulation world) the manipulated ship survived in average as an indicator to quantize its ability when using different methods. Such an evaluation method is common in RL research [43]. The result is given in Fig. 9. In this figure, X-axis represents the training time (hours), and Y-axis represents the average surviving steps in 10 rounds of trails.

As elaborated previously, there is a bias or an offset placed into the rudder of this ship on purpose, and such disturbance makes the robust control of the ship more difficult. Three human volunteers who have never control a ship before were asked to control this ship for 100 rounds. Therefore, it takes hours for them to get used to the imbalance of rudder. Meanwhile, the results have been recorded by the simulation platform mentioned in Section 4.1. However, even so, he or she generally cannot make the ship survive more than 350 steps, since the dynamic obstacles appeared on any palaces, requiring many intense manipulations. The LOS/PID-based approach is capable of maintaining the heading of this ship perfectly. However, this method only performs satisfactorily when facing static obstacles. In the testing, such method generally lasts in this test for 112 steps on average. The vector/APF-based method seems to be completely unable to control this ship, which can only last for 51 steps on average. In fact, many improvements had been made on the LOS/PID-based methods, APF-based methods. For simplification, only the classical forms of these methods were compared here.

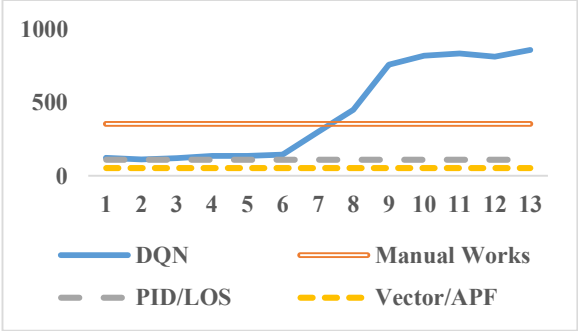


Fig.9. Comparisons with manual works, PID and APF-based methods

In the training, the performance of the proposed approach became better after 6 hours of training. After 13 hours, the performance of the approach had converged. Then, this controlled ship can always find a way to evade from threats. It collided with obstacles only when facing hopeless complexions, since rocks and encountered vessels refreshed randomly. Meanwhile, this controlled ship always knew to get back to the middle of the planned route. In this simulation world, we can give a conclusion that the proposed approach performs better than human players without any human experience input. All the techniques of evading from encountered vessels and static obstacles, controlling such an imbalanced ship are learned through un-supervised training. Moreover, the short-term path planning and motion controlling are accomplished simultaneously.

5. Conclusions and future work

In the area of autonomous ship, researchers are more willing to divide the collision avoidance problem into two sub-systems, path planning and motion controlling. However, facing a general-purpose cargo ship, more and more factors should be taken into consideration, including the relatively insufficient driving power, customary routes, navigational regulations and others. In this situation, neuron networks might be a solution. In this paper, a novel perspective was put forward in which multiple factors in collision avoidance can be described and modelled as the constraints of RL-based methods. To prove the effectiveness, a DQN-based approach was proposed, which took the ship manoeuvring, navigation regulations, routes, collision avoidance as a merged training problem. In the simulations, it is proved to be superior to manual works after 13 hours of training. Hence, the DQN and other Deep RL-based methods might be a promising way to build artificial intelligence of controlling a cargo ship.

In the future, the proposed DQN-based approach can be further improved.

1) Limited actions of the cargo ship had been used to reduce the training time and the calculation amount in this research, making the behaviours of the trained cargo weird in a normal sense. More actions should be introduced in the future.

2) More factors should be further discussed, so as to make the behaviours of the cargo closer to a helmsman. The corresponding reward function should be improved accordingly.

3) The constant coefficients of the reward function in DQN should be determined with rigorous and reasonable methods, such as imitate RL-based method.

4) The proposed approach and the model will be further validated on a real cargo ship.

Acknowledgements

This work is supported by National Key R&D Program of China [2018YFB1601503] and Ministry of Industry and Information Technology of the People's Republic of China [2018473].

References

[1] L. Kretschmann, H.C. Burmeister, C. Jahn, Analyzing the economic benefit of unmanned autonomous ships: An exploratory cost-comparison between an autonomous and a conventional bulk carrier, Research in Transportation Business & Management. 25 (2017) 76-86.

[2] P. Švec, A. Thakur, E. Raboin, B.C. Shah, S.K. Gupta, Target following with motion prediction for unmanned surface vehicle operating in cluttered environments, Autonomous Robots. 36 (2014) 383-405.

[3] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, Human-level control through deep reinforcement learning, Nature. 518 (2015) 529.

[4] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, Mastering the game of Go with deep neural networks and tree search, Nature. 529 (2016) 484.

[5] W. Naeem, T. Xu, R. Sutton, A. Tiano, The design of a navigation, guidance, and control system for an unmanned surface vehicle for environmental monitoring, Proceedings of the Institution of Mechanical Engineers Part M: Journal of Engineering for the Maritime Environment. 222 (2008) 67-79.

[6] J. Alves, P. Oliveira, R. Oliveira, A. Pascoal, M. Rufino, L. Sebastiao, C. Silvestre, Vehicle and mission control of the DELFIM autonomous surface craft, in: 2006 14th Mediterranean Conference on Control and Automation, IEEE, 2006, pp. 1-6.

[7] T.I. Fossen, Handbook of marine craft hydrodynamics and motion control, John Wiley & Sons, 2011.

[8] L.E. van Cappelle, L. Chen, R.R. Negenborn, Survey on short-term technology developments and readiness levels for autonomous shipping, in: Proceedings of the 9th International Conference on Computational Logistics (ICCL 2018), Vietri sul Mare, Italy, 2018, pp. 106-123.

[9] M. Schiaretti, L. Chen, R.R. Negenborn, Survey on autonomous surface vessels: Part I - A new detailed definition of autonomy levels, in: Proceedings of the 8th International Conference on Computational Logistics (ICCL 2017), Southampton, UK, 2017, pp. 219-233.

[10] M. Schiaretti, L. Chen, R.R. Negenborn, Survey on autonomous surface vessels: Part II - Categorization of 60 prototypes and future applications, in: Proceedings of the 8th International Conference on Computational Logistics (ICCL 2017), Southampton, UK, 2017, pp. 234-252.

[11] L. Chen, R.R. Negenborn, G. Lodewijks, Path Planning for Autonomous Inland Vessels Using A*BG, in: Proceedings of the 7th International Conference on Computational Logistics (ICCL 2016), Lisbon, Portugal, 2016, pp. 65-79.

[12] G. Casalino, A. Turetta, E. Simetti, A three-layered architecture for real time path planning and obstacle avoidance for surveillance USVs operating in harbour fields, in: Oceans 2009-Europe, IEEE, 2009, pp. 1-8.

[13] S. Campbell, W. Naeem, A rule-based heuristic method for colregs-compliant collision avoidance for an unmanned surface vehicle, IFAC Proceedings Volumes. 45 (2012) 386-391.

[14] F. Ma, Y. Chen, Probabilistic Assessment of Vessel Collision Risk: An Evidential Reasoning and Artificial Potential Field-Based Method, in: Multi-Criteria Decision Making in Maritime Studies and Logistics, Springer, 2018: pp. 123-149.

[15] T. Wang, X. Yan, Y. Wang, Q. Wu, Ship domain model for multi-ship collision avoidance decision-making with COLREGs based on artificial potential field, TransNav: International Journal on Marine Navigation and Safety of Sea Transportation. 11 (2017) 85-92.

[16] A. Lazarowska, A New Potential Field Inspired Path Planning Algorithm for Ships, in: 2018 23rd International Conference on Methods & Models in Automation & Robotics (MMAR), IEEE, 2018, pp. 166-170.

[17] S. Campbell, W. Naeem, G.W. Irwin, A review on improving the autonomy of unmanned surface vehicles through intelligent collision avoidance manoeuvres, Annual Reviews in Control. 36 (2012) 267-283.

[18] E. Zereik, A. Sorbara, M. Bibuli, G. Bruzzone, M. Caccia, Priority Task Approach for USVs' Path Following Missions with Obstacle Avoidance and Speed Regulation, IFAC-Papers On Line. 48 (2015) 25-30.

[19] S. Moe, K.Y. Pettersen, Set-based Line-of-Sight (LOS) path following with collision avoidance for underactuated unmanned surface vessel, in: 2016 24th Mediterranean Conference on Control and Automation, IEEE, 2016, pp. 402-409.

[20] Y. Liu, R. Song, R. Bucknall, A practical path planning and navigation algorithm for an unmanned surface vehicle using the fast marching algorithm, in: OCEANS 2015-Genova, IEEE, 2015, pp. 1-7.

[21] H. Zheng, R.R. Negenborn, G. Lodewijks, Predictive path following with arrival time awareness for waterborne AGVs, Transportation Research Part C: Emerging Technologies. 70 (2016) 214-237.

[22] R. Miao, Z. Dong, L. Wan, & J. Zeng, Heading control system design for a Micro-USV based on an adaptive expert S-PID algorithm. Polish Maritime Research. 25 (2018) 6-13.

[23] C. Liu, R.R. Negenborn, X. Chu, H. Zheng, Predictive path following based on adaptive line-of-sight for underactuated autonomous surface vessels, Journal of Marine Science and Technology. 23 (2018) 483-494.

[24] S.K. Sharma, R. Sutton, An optimised nonlinear model predictive control based autopilot for an uninhabited surface vehicle, IFAC Proceedings Volumes. 46 (2013) 73-78.

[25] K. Blekas, K. Vlachos, RL-based path planning for an over-actuated floating vehicle under disturbances, Robotics and Autonomous Systems. 101 (2018) 93-102.

[26] B. Yoo, J. Kim, Path optimization for marine vehicles in ocean currents using reinforcement learning, Journal of Marine Science and Technology. 21 (2016) 334-343.

[27] C. Chen, X. Chen, F. Ma, X. Zeng, J. Wang, A knowledge-free path planning approach for smart ships based on reinforcement learning, Ocean Engineering. 189 (2019) 106299.

[28] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing Atari with deep reinforcement learning, in: Neural Information Processing Systems Deep Learning Workshop, 2013.
[29] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J.M. Allen, V.D. Lam, A. Bewley, A. Shah, Learning to drive in a day, in: 2019 International Conference on Robotics and Automation, IEEE, 2019, pp. 8248-8254.
[30] P. Mirowski, M. Grimes, M. Malinowski, K.M. Hermann, K. Anderson, D. Teplyashin, K. Simonyan, A. Zisserman, R. Hadsell, Learning to navigate in cities without a map, in: Advances in Neural Information Processing Systems, 2018, pp. 2419-2430.
[31] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L.D. Jackel, M. Monfort, U. Muller, J. Zhang, End to end learning for self-driving cars, (2016). arXiv preprint arXiv:1604.07316 .
[32] Y. Cheng, W. Zhang, Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels, Neurocomputing. 272 (2018) 63-73.
[33] J. Liu, F. Quadvlieg, R. Hekkenberg, Impacts of the rudder profile on manoeuvring performance of ships, Ocean Engineering. 124 (2016) 226-240.
[34] J. Liu, R. Hekkenberg, F. Quadvlieg, H. Hopman, B. Zhao, An integrated empirical manoeuvring model for inland vessels, Ocean Engineering, 137 (2017) 287-308.
[35] G. Zhao, Y. Tao, H. Liu, X. Deng, Y. Chen, H. Xiong, X. Xie, A robot demonstration method based on LWR and Q-learning algorithm, Journal of Interlligent & Fuzzy Systems, 35 (2018) 35-46.
[36] I. Carlucho, M. De Paula, S. Wang, Y. Petillot, G.G. Acosta, Adaptive low-level control of autonomous underwater vehicles using deep reinforcement learning, Robotics & Autonomous Systems. 107 (2018) 71-86.
[37] Z. Wang, N. De Freitas, M. Lanctot, Dueling Network Architectures for Deep Reinforcement Learning, in: 33rd International Conference on Machine Learning (ICML), 2016.
[38] Y. Rasekhipour, A. Khajepour, S.K. Chen, B. Litkouhi, A Potential Field-Based Model Predictive Path-Planning Controller for Autonomous Road Vehicles, IEEE Transactions on Intelligent Transportation Systems. 18 (2017) 1255-1267.
[39] F. Yahei, T. Kenichi, Traffic Capacity, Journal of Navigation. 24 (1971) 543–552.
[40] S.W. Lee, S.L. Toxopeus, F.H.H.A. Quadvlieg, Free Sailing Manoeuvring Tests on KVLCC 1 and KVLCC 2. Technical Report. Maritime Research Institute Netherlands, (2007).
[41] Quadvlieg, F.H.H.A. Brouwer, KVLCC2 Benchmark Data Including Uncertainty Analysis To Support Manoeuvring Predictions, (2011).
[42] M.G. Bellemare, J. Veness, M. Bowling, Investigating Contingency Awareness Using Atari 2600 Games, in: 2012.
[43] A. Banino, C. Barry, B. Uria, et al., Vector-based navigation using grid-like representations in artificial agents. Nature. 557(2018) 429-447.